

HANDBOOK OF STATISTICAL ANALYSIS

HANDBOOK OF STATISTICAL ANALYSIS: AI AND ML APPLICATIONS

THIRD EDITION

Authors

ROBERT NISBET, PH.D.

University of California, Irvine, Predictive Analytics Certificate Program (Retired), Santa Barbara, Goleta, California

KEITH MCCORMICK

*Data Science Principal, Further; Instructor, Curriculum Developer, and Advisory Board Member, Predictive Analytics Certificate Program,
University of California, Irvine Division of Continuing Education, Durham, North Carolina*

GARY MINER, PH.D.

University of California, Irvine, Predictive Analytics Certificate Program (Retired), Tulsa, Oklahoma, and Rome, Georgia



Academic Press is an imprint of Elsevier
125 London Wall, London EC2Y 5AS, United Kingdom
525 B Street, Suite 1650, San Diego, CA 92101, United States
50 Hampshire Street, 5th Floor, Cambridge, MA 02139, United States

Copyright © 2025 Elsevier Inc. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

Publisher's note: Elsevier takes a neutral position with respect to territorial disputes or jurisdictional claims in its published content, including in maps and institutional affiliations.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: www.elsevier.com/permissions.

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary.

Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

ISBN: 978-0-443-15845-2

For Information on all Academic Press publications
visit our website at <https://www.elsevier.com/books-and-journals>

This book has a companion website hosting complementary materials.
Visit this URL to access it: <https://www.elsevier.com/books-and-journals/book-companion/9780443158452>.

Publisher: Peter Linsley
Acquisitions Editor: Stephanie Cohen
Editorial Project Manager: Teddy A. Lewis
Production Project Manager: Gayathri S
Cover Designer: Mark Rogers

Typeset by MPS Limited, Chennai, India

Printed in India

Last digit is the print number: 9 8 7 6 5 4 3 2 1



Working together
to grow libraries in
developing countries

www.elsevier.com • www.bookaid.org

Contents

Foreword 1 for first edition	xi	Data understanding phase (mostly science)	38
Foreword 2 for first edition	xiii	Data preparation phase (a mixture of art and science)	40
Foreword—to this third edition	xv	Modeling (a mixture of art and science)	41
Preface	xvii	Deployment (mostly art)	43
Praise for the first edition of this book	xix	Closing the information loop (art)	43
Biographies of the primary authors		The art of data analytics	43
of this book	xxi	Postscript	44
		References	44

I

Introduction

1. The background and history of predictive analytics

Preamble	3	Prelude	45
Phases in the history of statistics and predictive analytics	4	Accessory tools	45
Current trends of development in predictive analytics	21	Pure coding languages	46
Postscript	21	Menu-based procedural platforms	47
References	21	Multipersona data science and machine learning (DSML)	

2. Theoretical considerations for data analytics

Preamble	23	platforms with graphical programming canvases	50
The scientific method	23	and AutoML capabilities	52
What is the process of data analytics?	24	Advice on tool selection and professional development	53
A theoretical framework for the data analytics process	25	Postscript	53
Strengths of the data analytics process	26	Reference	53
Customer centric versus account centric: a new way to			
look at your data	26		
The data paradigm shift	28		
Creation of the CAR	28		
Major activities of predictive analytics	28		
Major challenges of data analytics and predictive analytics	30		
General examples of predictive analytics applications	30		
Major issues in data analytics	30		
General requirements for success in a data analytics project	31		
Example of a data analytics project: classify a bat's			
species by its sound	31		
The importance of domain knowledge	32		
Postscript	33		
References	33		

3. The data analytics process

Preamble	35		
Two approaches to understanding and problem solving	35		
The role of process in data analytics	35		
Business understanding phase (mostly art)	37		

4. Tools

Prelude	45
Accessory tools	45
Pure coding languages	46
Menu-based procedural platforms	47
Multipersona data science and machine learning (DSML)	
platforms with graphical programming canvases	50
and AutoML capabilities	52
Advice on tool selection and professional development	53
Postscript	53
Reference	53

II

Data preparation

5. Data access

Preamble	57
Knowledge from data: predictive analytics	57
Overall structure of the database – data modeling	57
Account-centric format of data in early	
computer databases	57
The customer-centric analytical data mart	58
The ETL process	59
Issues in data mart design	66
Current trends in data access and data	
structure design	66
References	67

6. Data Understanding

Prelude	69
Four Data Understanding tasks in CRISP-DM	69
KNIME examples in Data Understanding	74
IBM SPSS modeler examples	74
Reference	74

7. Visualizing data for understanding

Foundations of data visualization: John Tukey	75
Importance of the audience of visualizations	75
Seeing is believing	76
Case Study: visualizations of the Ames Housing data set	76
Postscript	84
References	84

8. Data cleaning

Preamble	85
Data cleaning operation classification	85
Types of data cleaning operations	85
Technical correction of data	85
String manipulation	89
Data filtering	90
Time-series filtering	90
Outlier handling	90
Postscript	93
References	93

9. Data conditioning

Preamble	95
Data conditioning operations that generate consistent data sets	95
Three types of data conditioning operations are discussed in this chapter	95
Postscript	101
References	101

10. Data and feature engineering

Prelude	103
Definitions	103
Topics of discussion in Chapter 10	103
References	115

11. Feature selection

Modeling and sculpting	117
Some introductory concepts	117
Feature selection versus dimensionality reduction	119
Classification of dimensionality reduction methods	119
Decision tree processing elements useful as feature reduction filters	120
KNIME examples of feature selection methods	121
Correlation filtering (a filter method)	122
References	123
Additional bibliography for feature selection	123

12. A data preparation cookbook

Preamble	125
Introduction	125
1. CRISP-DM phase I: business understanding	125
2. CRISP-DM phase II: data understanding	127
3. CRISP-DM phase III: data preparation	129
4. CRISP-DM phase IV: modeling	131
18 common mistakes in data preparation in predictive analytics projects	132
References	134

III

Modeling

13. A taxonomy of algorithms

Preamble	137
Introduction	137
The top five modeling algorithms	137
More advanced algorithms	138
A brief introduction to the top five algorithms	140
Keywords and phrases	147
References	147

14. Modeling methods

Prelude	149
Scope of modeling operations	149
General modeling issues	149
Types of analytical models	150
Two approaches to building analytical models	152
Two basic types of predictive models	153
The parameter optimization loop in KNIME	167
Keywords and phrases	168
References	168

15. Model evaluation and enhancement

Preamble	169
Evaluation and enhancement: part of the modeling process	169
Types of errors in analytical models	169
Assessment of random error	173
Model enhancement techniques	177
Model enhancement checklist	180
References	181

16. Ensembles and complexity: a practical approach

Preamble	183
Modeling: How to choose the best modeling option?	183
Do ensembles violate Occam's razor?	184
The power of ensembles	184
Types of ensemble modeling techniques	184
General structure of a fused ensemble model	185
References	191

17. Deep learning versus traditional machine learning

Preamble	193
Introduction	193
Differences between deep learning and traditional machine learning	193
The journey to deep learning	194
Why is feature engineering so important	196
Interactions	197
Interaction representation in artificial neural networks	199
Postscript	200
References	200

18. Interpretable machine learning and explainable AI

Preamble	201
Introduction	201
Comparing interpretable machine learning and XAI	202
Comparing two types of XAI: global and local explanations	204
Techniques for global explanations	205
Techniques for local explanations	207
LIME	208
Regulatory considerations	208
The proposed AI Act in the EU	209
Postscript	209
References	209

19. Human in the loop machine learning and data annotation

Preamble	211
Introduction	211
Defining human in the loop	212
Concurrent rise of human in the loop and deep learning	212
Common human in the loop use cases	213
The data annotation ecosystem	215
The ethics of data annotation	216
Issues associated with managing a data annotation project	217
Becoming “data centric”	217
Postscript	218
References	218

IV

Applications

20. Overview of healthcare delivery and medical informatics

Prelude	223
Introduction	223
Chapter 1: The purpose of the book on medical data analytics	223
Chapter 2: History of predictive analytics in medicine and health care	223
Chapter 3: Bioinformatics	223
Chapter 4: Data and process models in medical informatics	224
Chapter 5: Access to data for analytics – the initial issue in medical and healthcare predictive analytics	224
Chapter 6: Precision (personalized) medicine	224
Chapter 7: Patient-directed health care	224
Chapter 8: Regulatory measures – agencies and data issues in medicine and health care	224
Chapter 9: Predictive analytics with multiomics data	224
Chapter 10: Artificial intelligence and genomics	225
Chapter 11: Glaucoma (eye disease): a case study with suggested predictive analytics modeling for identifying an individual patient’s best diagnosis and best treatment	225
Chapter 12: Practical application example: using data science algorithms in predicting ICU patient urine output in response to diuretics to aid clinicians and healthcare providers in clinical decision making	225

Chapter 13: Practical application example: prediction tool development	225
Chapter 14: Modeling precancerous colon polyps with OMOP data	225
Chapter 15: Prediction of pancreatic and lung cancer from metabolomics data	226
Chapter 16: COVID-19 descriptive analytics visualization of pandemic and hospitalization data	227
Chapter 17: Disseminated intravascular coagulation predictive analytics with pediatric ICU admissions	227
Chapter 18: Challenges for healthcare administration and delivery: integrating predictive and prescriptive modeling into personalized precision health care	227
Chapter 19: Challenges of medical research in incorporating modern data analytics in studies	227
Chapter 20: The nature of insight from data and implications for automated decisioning	227
Chapter 21: Model management and ModelOps: managing an artificial intelligence-driven enterprise	228
Chapter 22: Forecasts for advances in predictive and prescriptive analytics	228
Chapter 23: Sampling and data analysis	228
Chapter 24: Analytics architectures for the 21st century	228
Chapter 25: Predictive models versus prescriptive models causal inference and Bayesian networks	228
Chapter 26: The future: 21st-century health care and wellness in the digital age	228
Postscript	229
References	229

21. Customer response modeling with temporal abstractions

Preamble	231
Early CRM issues in business	231
Knowing how customers behaved before they acted	231
Transforming corporations into business ecosystems: The path to customer fulfillment	232
CRM modeling in business ecosystems	232
Modeling with different sets of predictor variables	235
Discussion	237
Conclusions	238
Postscript	238
References	239

22. Academic Analytics: a case study in student retention

Preamble	241
Introduction	241
Academic Analytics	241
Student retention: a case study	243
Methodology	243
Results	245
Key word list	246
Discussion	247
Postscript	247
References	247

23. Medical informatics case study: predicting type 1 precancerous colon polyps

Prelude	249
Introduction	249
The University of California, Irvine Colon Polyp Risk Modeling Project	249
Modeling	254
Results	256
Discussion	258
Postscript	258
References	258

24. Credit risk case study

Prelude	259
Introduction	259
Credit scoring	259
Credit risk models	260
Case study: credit risk score for a loan application	260
Postscript	263
Reference	263

25. The intelligence revolution

Preamble	265
Introduction	265
Large language models (ChatGPT, BARD, etc.)	265
TinyML	268
HUMANE	268
Tiny to Large AI-FAB LABS	269
Data and Predictive Analytics as a “Personal AI” or “Company AI” that functions automatically with AI and AGI – the future is only 3–7 years away!!!	270
Postscript	271
References	271

V

Right models – luck & ethics of analytics

26 The “Right Model” for the “Right Purpose” – when less is good enough

Preamble	275
More is not necessarily better: Lessons from nature and engineering	275
References	280

27. Ethics and data analytics

Preamble	281
The birthday party – a practical example for ethical action	281
Secular ethics	281
Michael Sandel on “Doing the Right Thing” with data analytics	282
Alignment of ethical perspectives: The ethical road map	283
What is generative artificial intelligence?	284
Postscript	285
References	285

28. The significance of luck

Preamble	287
Introduction	287
The problem of significance in traditional p -value statistical analysis	287
The response of the American Statistical Association	288
Postscript	290
References	290

VI

Tutorials and case studies

Tutorial A Example of data mining recipes using Windows 10 and STATISTICA 13	293
LINDA A. MINER	

Tutorial B Analysis of Hurricane data (Hurrdata.sta) using the STATISTICA Data Miner 13	303
JEFF WONG	

Tutorial C Case study – Predicting student success at high-stakes nursing examinations (NCLEX) using IBM SPSS Modeler and STATISTICA Data Miner 13	317
GALINA BELOKUROVA AND CHIARINA PIAZZA	

Tutorial D Constructing a Histogram using MidWest Company Personality Data with KNIME	339
LINDA A. MINER AND ROBERT NISBET	

Tutorial E Feature selection using KNIME	353
ROBERT NISBET	

Tutorial F Medical/business tutorial using STATISTICA Data Miner 13 determining possible predictors for days with hospice service for patients with dementia	367
LINDA A. MINER, JAMES ROSS, AND KAREN JAMES	

Tutorial G Modeling hospice residence stay with KNIME	393
LINDA A. MINER AND ROBERT NISBET	

Tutorial H Data preparation: Merging data sources	401
ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	

Tutorial I Data description ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	413	Tutorial N Filling missing values with formulas ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	445
Tutorial J Data cleaning and recoding ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	419	Tutorial O Filling missing values with a model ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	449
Tutorial K Dummy coding category variables ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	427	Appendix A Listing of tutorials and other resources on this book's companion web page	459
Tutorial L Outlier handling ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	435	Appendix B	463
Tutorial M Filling missing values with constants ROBERTA BORTOLOTTI, SENIOR DIRECTOR, NASDAQ	441	Index	465

Foreword 1 for first edition

This book will help the novice user become familiar with data mining. Basically, data mining is performing data analysis (or statistics) on data sets (often large) that have been obtained from potentially many sources. As such, the miner may not have control of the input data but must rely on sources that have gathered the data. As such, there are problems that every data miner must be aware of as they begin (or complete) a mining operation. I strongly resonated with the material on “The Top 10 Data Mining Mistakes,” which gives a worthwhile checklist:

- Ensure you have a response variable and predictor variables—and that they are correctly measured.
- Beware of overfitting. With scads of variables, it is easy with most statistical programs to fit incredibly complex models, but they cannot be reproduced. It is good to save part of the sample to use to test the model. Various methods are offered in this book.
- Don’t use only one method. Using only linear regression can be a problem.
 - Try dichotomizing the response or categorizing it to remove nonlinearities in the response variable. Often, there are clusters of values at zero, which messes up any normality assumption. This, of course, loses information, so you may want to categorize a continuous response variable and use an alternative to regression. Similarly, predictor variables may need to be treated as factors rather than linear predictors. A classic example is using marital status or race as a linear predictor when there is no order.
- Asking the wrong question—when looking for a rare phenomenon, it may be helpful to identify the most common pattern. These may lead to complex analyses, as in item 3, but they may also be conceptually simple. Again, you may need to take care that you don’t overfit the data.
- Don’t become enamored with the data. There may be a substantial history from earlier data or from domain experts that can help with the modeling.
- Be wary of using an outcome variable (or one highly correlated with the outcome variable) and becoming excited about the result. The predictors should be “proper” predictors in the sense that they (a) are measured prior to the outcome and (b) are not a function of the outcome.
- Do not discard outliers without solid justification. Just because an observation is out of line with others, it shouldn’t be ignored. You must check the circumstances that led to the value. In any event, it is useful to conduct the analysis with the observation(s) included and excluded to determine the sensitivity of the results to the outlier.
- Extrapolating is a fine way to go broke; the best example is the stock market. Stick within your data and if you must go outside, put plenty of caveats. Better still, restrain the impulse to extrapolate. Beware that pictures are often far too simple and we can be misled. Political campaigns oversimplify complex problems (“my opponent wants to raise taxes”; “my opponent will take us to war”) when the realities may imply we have some infrastructure needs that can be handled only with new funding or we have been attacked by some bad guys. Be wary of your data sources. If you are combining several sets of data, they need to meet a few standards:
- The definitions of variables that are being merged should be identical. Often, they are close but not exact (especially in metaanalysis, where clinical studies may have somewhat different definitions due to different medical institutions or laboratories).
- Be careful about missing values. Often, when multiple data sets are merged, missing values can be induced: one variable isn’t present in another data set; what you thought was a unique variable name was slightly different in the two sets, so you end up with two variables that both have a lot of missing values.
- How you handle missing values can be crucial. In one example, I used complete cases and lost half of my sample; all variables had at least 85% completeness, but when put together, the sample lost half of the data. The residual sum of squares from a stepwise regression was about 8. When I included more variables using mean replacement,

almost the same set of predictor variables surfaced, but the residual sum of squares was 20. I then used multiple imputations and found approximately the same set of predictors but had a residual sum of squares (median of 20 imputations) of 25. I find that mean replacement is rather optimistic but surely better than relying on only complete cases. Using stepwise regression, I find it useful to replicate it with a bootstrap or with multiple imputations. However, with large data sets, this approach may be expensive computationally.

To conclude, there is a wealth of material in this handbook that will repay study.

Peter A. Lachenbruch, Ph.D.

Oregon State University, Corvallis, OR

United States American Statistical Association,

Alexandria, VA, United States

Johns Hopkins University, Baltimore, MD, United States

UCLA, Los Angeles, CA, United States

University of Iowa, Iowa City, IA, United States

University of North Carolina, Chapel Hill, NC,

United States

Foreword 2 for first edition

A November 2008 search on www.amazon.com for “data mining” books yielded over 15,000 hits—including 72 to be published in 2009. Most of these books either describe data mining in very technical and mathematical terms beyond the reach of most individuals or approach data mining at an introductory level without sufficient detail to be useful to the practitioner. The *Handbook of Statistical Analysis and Data Mining Applications* is the book that strikes the right balance between these two treatments of data mining.

This volume is not a theoretical treatment of the subject—the authors themselves recommend other books for this—but rather contains a description of data mining principles and techniques in a series of “knowledge-transfer” sessions, where examples from real data mining projects illustrate the main ideas. This aspect of the book makes it most valuable for practitioners, whether novice or more experienced.

While it would be easier for everyone if data mining were merely a matter of finding and applying the correct mathematical equation or approach for any given problem, the reality is that both “art” and “science” are necessary. The “art” in data mining requires experience: when one has seen and overcome the difficulties in finding solutions from among the many possible approaches, one can apply newfound wisdom to the next project. However, this process takes considerable time, and particularly for data mining novices, the iterative process inevitable in data mining can lead to discouragement when a “textbook” approach doesn’t yield a good solution.

This book is different; it is organized with the practitioner in mind. The volume is divided into four parts. Part I provides an overview of analytics from a historical perspective and frameworks from which to approach data mining, including CRISP-DM and SEMMA. These chapters will provide a novice analyst an excellent overview by defining terms and methods to use and will provide program managers with a framework from which to approach a wide variety of data mining problems. Part II describes algorithms, though without extensive mathematics. These will appeal to practitioners who

are or will be involved with day-to-day analytics and need to understand the qualitative aspects of the algorithms. The inclusion of a chapter on text mining is particularly timely, as text mining has shown tremendous growth in recent years.

Part III provides a series of tutorials that are both domain specific and software specific. Any instructor knows that examples make the abstract concept more concrete, and these tutorials accomplish exactly that. In addition, each tutorial shows how the solutions were developed using popular data mining software tools, such as Clementine, Enterprise Miner, Weka, and STATISTICA. The step-by-step specifics will assist practitioners in learning not only how to approach a wide variety of problems but also how to use these software products effectively. Part IV presents a look at the future of data mining, including a treatment of model ensembles and “The Top 10 Data Mining Mistakes,” from the popular presentation by Dr. Elder.

However, the book is best read a few chapters at a time while actively doing the data mining rather than read cover to cover (a daunting task for a book this size). Practitioners will appreciate tutorials that match their business objectives and choose to ignore other tutorials. They may choose to read sections on a particular algorithm to increase insight into that algorithm and then decide to add a second algorithm after the first is mastered. For those new to a particular software tool highlighted in the tutorials section, the step-by-step approach will operate much like a user’s manual. Many chapters stand well on their own, such as the excellent “History of Statistics and Data Mining” chapter and [Chapters 16, 17, and 18](#). These are broadly applicable and should be read by even the most experienced data miners.

The *Handbook of Statistical Analysis and Data Mining Applications* is an exceptional book that should be on every data miner’s bookshelf or, better yet, found lying open next to their computer.

Dean Abbott
Abbott Analytics, San Diego, CA,
United States

Foreword—to this third edition

By *Usama Fayyad and Hamit Hamutcu, the Institute for Experiential AI at Northeastern University*

As data mining evolved and started addressing additional aspects, including an emphasis on unstructured data, heterogeneous databases, and the growing complexity of schema and data lakes, the strong relevance of the more diverse activities known as “data science” became more prominent. Data science is transdisciplinary. Regardless of how one defines data science, and there have certainly been many attempts at this, the field incorporates principles, techniques, and methods from multiple existing academic disciplines and application domains. As a natural result of this, the knowledge and skills that organizations need and expect from data professionals also cover a broad range. Our research, as part of the work at the Initiative for Analytics and Data Science Standards (IADSS), has shown that they range from traditional statistical analysis to cutting-edge learning and mining algorithms and, in our framework, we list more than 60 topics and subtopics as foundational pieces of knowledge and skills to practice data science in the industry.

With this in mind, we were pleased to write the foreword to the third edition of this vital text on statistical analysis and data mining. The three authors skillfully blend the art of the practice with an injection of knowledge on how to use the relevant various techniques from multiple fields to make “Data Analytics” a reality in the context of practical applications. The book offers readers a roadmap to navigate the intricacies of data exploration and analysis, or as Dean Abbott referred to in his foreword to the first edition of this book, the approach addresses the “art” and “science” of data analysis and mining.

In a world shaped by the revolutionary performance of technologies like AlexNet in 2012 and the cultural influence of ChatGPT, we find ourselves amid an AI summer (as opposed to the infamous AI winters a few decades ago). In a rush toward cutting-edge technologies, there is a risk of graduating a generation of data scientists fueled by the excitement about LLMs, generative AI, and deep learning, yet lacking exposure to proven techniques that might be better suited to a particular problem at hand. This book’s alignment with time-tested approaches to structured data and traditional use cases

resonates deeply with our belief in the ever-evolving and diverse skills required in the field of data analytics.

The authors draw upon decades of experience, tracing their techniques back to pivotal influences from the 1990s, starting with the development of CRISP-DM in the data mining community. The relevance of traditional use cases on structured data, from fraud detection to predictive maintenance, persists in today’s data landscape. This book serves as a guide to the skills needed for such scenarios—problem definition, scientific method, inferential statistics, descriptive statistics, data exploration, feature engineering, and more. It draws a lineage from the past to the present, acknowledging the foundational skills that remain crucial in solving real-world challenges. The book does not target explaining the details of how algorithms work but rather, brings awareness of such algorithms and more importantly, a guide for how to be an effective user of such algorithms if you are trying to perform data analytics work.

We are firm believers that data science is an applied field, and hands-on experience in the real world is essential to any formal training. You could consider this similar to why a medical residency is an integral part of a physician’s training after medical school. This book provides a unique blend of practical solutions to day-to-day business problems through the inclusion of dozens of case studies from multiple domains. More importantly, it aims to be an accessible “user guide” to the various methods available to the practitioner. The work of the authors not only preserves the rich traditions of the field but also helps to grow the practice into the future of data analytics.

Biographical information on the foreword writers

Usama Fayyad, Ph.D. short bio: Ph.D. in engineering in AI/machine learning from the University of Michigan, Ann Arbor; two bachelor of science degrees in engineering, a master of science degree in engineering in computer engineering, and an M.Sc. in mathematics.

Most noteworthy:

- Usama heads up the Institute for Experiential AI at Northeastern University.
- **He was the first person with the CDO (Chief Data Officer) title when Yahoo! acquired his second startup.**
- **He held the Global CDO role at Barclays Bank for 3 years.**

Additional accomplishments of Usama Fayyad:

- Professor of the practice at Khoury College of Computer Sciences.
- Chairman of Open Insights, a technology and consulting firm he founded in 2008 after leaving Yahoo!.
- At Yahoo!, he served as the company's first Chief Data Officer and founded Yahoo! Research Labs as EVP.
- Founding Executive Chairman at Oasis500.
- Cofounder and CTO of OODA Health, a health tech company to streamline payments and claims.
- Held leadership roles at Microsoft and ran research and product teams there.
- Founded the Machine Learning Systems group at NASA's Jet Propulsion Laboratory (*where his work on machine learning resulted in the top Excellence in Research award from Caltech and a US Government medal from NASA*)
- Fayyad has published over 100 technical articles on data mining, data science, AI/ML, and databases.
- He holds over 20 patents.
- He is a fellow of both the Association for the Advancement of Artificial Intelligence (AAAI) and the Association for Computing Machinery (ACM).
- He has edited two influential books on data mining/data science.
- He served as founding editor-in-chief of two key journals.
- Cofounder at the *Initiative for Analytics and Data Science Standards* (IADSS).

Hamit Hamutcu, BS, MBA short bio: BS in electronics engineering at Bogazici University in Istanbul; MBA from the University of Florida.

- Senior director at the Institute for Experiential AI at Northeastern University, where he leads in strategy development.

- He brings 25 years of industry and consulting experience in the areas of analytics and data-driven business strategy.
- Hamit is the cofounder of the *Initiative for Analytics and Data Science Standards*, IADSS, identifying and standardizing the skills necessary for high-performing data science, machine learning, and AI teams (<https://www.iadss.org/>).
- He is also cofounder at *Elements*, an innovative data skills assessment platform.

Previously, Hamit's career involved:

- Being a partner for Peppers & Rogers Group in Stamford, Connecticut; heading the Global Analytics Group; being founding partner for the Europe, Middle East, and Africa offices and growing the firm in those regions.
- Marketing analytics and technology positions at FedEx in Memphis, Tennessee.

Hamit is a frequent speaker, writer, and board member at various startups and nonprofit organizations.

Contact information:

Usama Fayyad, Ph.D. (u.fayyad@northeastern.edu)
Professor of the practice, Northeastern University
Boston, MA, USA

Chairman and founder, Open Insights (usama@open-insights.com)

Executive director, Institute for Experiential AI—
Northeastern University

EA: Rana Shami—+1-865-396-9500 (rana@open-insights.com or r.alshami@northeastern.edu)

Preface

Much has happened in the professional discipline known previously as data mining since the second edition of this book was written in 2016. This discipline has broadened and deepened to a very large extent, requiring a major reorganization of its elements. A new parent discipline was formed, data science, which includes previous subjects and activities in data mining, as well as many new elements of the scientific study of data, including storage structures optimized for analytical use, data ethics, and the performance of many activities in business, industry, and education. Analytical aspects that used to be included in data mining have broadened considerably to include image analysis, facial recognition, industrial performance and control, threat detection, fraud detection, astronomy, national security, weather forecasting, financial forensics, and many others. Consequently, several subdisciplines have been erected to contain various specialized data analytical applications. Vincent Granville lists these subdisciplines of data science:

- Machine learning – analytical algorithm design and optimization
- Data mining – now generally restricted in scope to pattern recognition apart from causes and interpretation
- Predictive analytics – using algorithms to predict things rather than describe them or manage them
- Statistical analysis – use of parametric statistical algorithms for analysis and prediction
- Industrial statistical analysis – analytical techniques to control and direct industrial operations
- Operations research – decision science and optimization of business processes
- Stock market quants – focused on stock market trading and portfolio optimization
- Data engineering – focused on optimizing data flow through memories and storage structures
- Business intelligence – focused primarily on descriptive aspects of data, but predictive aspects are coming
- Business analytics – focused primarily on the predictive aspects of data but is merging with descriptives

(<http://www.datasciencecentral.com/profiles/blogs/17-analytic-disciplines-compared>)

In this book, we will use the term data analytics to include aspects of data science for data access, data preparation, building analytical models, and methods to evaluate models. The book includes an extended discussion of the most popular data analytics processing plan (CRISP-DM). It will lead ordinary business analysts to follow a path through the CRISP-DM processing plan to create models that are “good enough.” These less-than-optimal models are far better in their ability to leverage faint patterns in databases to solve problems than the ways it used to be done. These tools provide default configurations and automatic operations, which shield the user from the technical complexity underneath. They provide one part in the crude analogy to the automobile interface. You don’t have to be a chemical engineer or physicist who understands moments of force to be able to operate a car. All you have to do is learn to turn the key in the ignition, step on the gas and the brake at the right times, turn the wheel to change direction in a safe manner, and voila, you are an expert user of the very complex technology under the hood. The other half of the story is the instruction manual and the driver’s education course that help you to learn how to drive.

This book provides that instruction manual and a series of case studies to show you how to perform many analytical operations and build predictive models in many subject areas. We provide intuitive explanations (rather than formal mathematical definitions) of the data analytics process and algorithms, which will enable even beginners to understand the basic concepts necessary to understand what they are doing. The case studies show how to implement the data analytics process in many different industries and businesses.

Overall organization of this book

We have divided the chapters in this book into four parts to guide you through the aspects of predictive analytics.

Part I covers the history, the CRISP-DM processing plan, and an overview of different types of data analytics tools. Part II discusses the many aspects of data preparation necessary for building analytical models. Part III includes an overview of algorithms used in analytical models and describes the general process of modeling

plus model evaluation and enhancement methods. Part IV presents some case studies describing examples of modeling in various data domains, plus discussions of several overall aspects of how models are used and abused.

The central theme of this book is the education of beginning data analytics practitioners, not the rigorous academic preparation of algorithm scientists. The approach chosen for this book is “a mile wide and an inch deep” by design, but there is a lot packed into that inch. There is enough here to stimulate you to take deeper dives into theory, and there is enough here to permit you to construct “smart enough” business operations with a relatively small amount of the right information.

James Taylor developed this concept for automating operational decision making in the area of enterprise decision management (Taylor, 2007). Taylor recognized that companies need decision-making systems that are automated enough to keep up with the volume and time-critical nature of modern business operations. These decisions should be deliberate, precise, consistent across the enterprise, smart enough to serve immediate needs appropriately, and agile enough to adapt to new opportunities and challenges in the company. The same concept can be applied to nonoperational systems for customer relationship management (CRM) and marketing support. Even though a CRM model for cross sell may not be optimal, it may enable several times the response rate in product sales following a marketing campaign. Models like this are “smart enough” to drive companies to the next level of sales. When models like this are proliferated throughout the enterprise to lift all sales to the next level, more refined models can be developed to do even better. This enterprise-wide “lift” in intelligent operations can drive a company through evolutionary rather than revolutionary changes to reach long-term goals. Companies can leverage “smart enough” decision systems to do likewise in their pursuit of optimal profitability in their business.

Clearly, the use of this book and these tools will not make you expert data analytics practitioners. Nor will the explanations in the book permit you to understand the complexity of the theory behind the algorithms and methodologies so necessary for the academic student. But we will conduct you through a relatively thin slice across the wide practice of data mining in many industries and disciplines. We can show you how to create powerful predictive models in your own organization in a relatively short period of time. In addition, this book can function as a springboard to launch you into higher-level studies of the theory behind the practice of data mining. If we can accomplish those goals, we will have succeeded in taking a significant step in bringing the practice of data mining into the mainstream of business analysis.

This book has a companion website hosting complementary materials. Visit this URL to access it: <https://www.elsevier.com/books-and-journals/book-companion/9780443158452>.

Acknowledgments

The three coauthors could not have done this book completely by themselves, and we wish to thank the following individuals, with the disclaimer that we apologize if, by our neglect, we have left out of this “thank-you list” anyone who contributed.

Foremost, we would like to thank our Elsevier Acquisitions Editor, Katey Birtcher, who saw the need for a third edition and “pestered” us until we consented, as well as all the other Elsevier people influential in making this third edition happen, including: Stephanie Cohen, editor in the US Elsevier office; Teddy A. Lewis, project manager in the UK Elsevier office; and Maria Shalini, production manager in the India Elsevier office. A big thanks to all of the Elsevier – Academic Press staff!!!

Dr. Robert Nisbet would like to honor and thank his wife, Jean Nisbet, Ph.D., who blasted him off in his technical career by retyping his Ph.D. dissertation five times (before word processing) and assumed much of the family’s burdens during the writing of this book. Dr. Nisbet also thanks Dr. Daniel B. Botkin, the famous global ecologist, for introducing him to the world of modeling and exposing him to the distinction between viewing the world as a machine and viewing it as an organism. And thanks are due to Ken Reed, Ph.D., for inducting Dr. Nisbet into the practice of data mining.

Coauthor Gary Miner wishes to thank his wife, Linda A. Winters-Miner, Ph.D., who has been working with him on similar books over the past 20 years and wrote several of the tutorials included in this book, using real-world data. Gary also wishes to thank the following people from his office who helped in various ways: Angela Waner, Jon Hillis, Greg Sergeant, and Dr. Thomas Hill, who gave permission to use and also edited a group of the tutorials that had been written over the years by some of the people listed as guest authors in this book. Without all the help of the people mentioned here and maybe many others we failed to specifically mention, this book would never have been completed. Thanks to you all!

Coauthor Keith McCormick would like to thank Gary and Dr. Nisbet for inviting him to join them in writing the third edition. Keith would also like to thank Dr. Nisbet for his mentorship and his encouragement as a fellow instructor in the UC Irvine Predictive Analytics Certificate program.

*Robert Nisbet
Keith McCormick
Gary Miner
September 30, 2023*

Praise for the first edition of this book

"Great introduction to the real-world process of data mining. The overviews, practical advice, tutorials, and extra DVD material make this book an invaluable resource for both new and experienced data miners."

Karl Rexer, Ph.D. (president and founder of Rexer Analytics, Boston, MA, www.RexerAnalytics.com)

Advance Praise for the Second Edition of This Book

Dr. Eric Siegel's ADVANCE PRAISE/"BLURB/REVIEW" for the FRONTISPIECE of the 2nd edition of *Handbook of Statistical Analysis and Data Mining Applications*, by Nisbet, Miner, and Yale.

Data mining practitioners, here is your bible, the complete "driver's manual" for data mining. From starting the engine to handling the curves, this book covers the gamut of data mining techniques—including predictive analytics and text mining—illustrating how to achieve maximal value across business, scientific, engineering, and medical applications. What are the best practices through each phase of a data mining project? How can you avoid the most treacherous pitfalls? The answers are in here.

Going beyond its responsibility as a reference book, the heavily updated second edition also provides all-new, detailed tutorials with step-by-step instructions to drive established data mining software tools across real-world applications. This way, newcomers start their engines immediately and experience hands-on success.

What's more, this edition drills down on hot topics across seven new chapters, including deep learning and how to avert "bullshit"* results. If you want to roll up

your sleeves and execute on predictive analytics, this is your definite, go-to resource. To put it lightly, if this book isn't on your shelf, you're not a data miner.

—Eric Siegel, Ph.D., Founder of Predictive Analytics World and author of *Predictive Analytics: The Power to Predict Who Will Click, Buy, Lie, or Die*

*This "BS" refers to the new course introduced in March 2017 at the University of Washington on "How to Perceive 'Bullshit'" in what one reads in science journal articles, the newspaper, and elsewhere ... a course that "filled in 1 minute" upon being listed at this university in Seattle.

The authors discuss this further in [Chapter 20: Significance and Falsehoods in Data Analysis](#).

"Statistical thinking will one day be as necessary for efficient citizenship as the ability to read and write."

H. G. Wells (1866–1946)

*"Today we aren't quite to the place that H. G. Wells predicted years ago, but society is getting closer out of necessity. Global businesses and organizations are being forced to use statistical analysis and data mining applications in a format that combines **art** and **science**—**intuition** and **expertise** in collecting and understanding data in order to make **accurate models** that realistically **predict** the **future** and lead to informed strategic **decisions**, thus allowing correct **actions** and ensuring **success**, before it is too late ... today, numeracy is as essential as literacy. As John Elder likes to say: 'Go data mining!' It really does save enormous time and money. For those with the patience and faith to get through the early stages of business understanding and data transformation, the cascade of results can be extremely rewarding."*

Gary Miner, September, 2017

Biographies of the primary authors of this book

ROBERT NISBET, Ph.D.

Dr. Nisbet was trained initially in ecology and ecosystems analysis. He has over 30 years of experience in complex system analysis and modeling, most recently as a researcher in forest growth modeling at the University of California, Santa Barbara. During 25 years in business, he pioneered the design and development of configurable data analytics applications for retail sales forecasting and churn, propensity to buy, and customer acquisition in telecommunications insurance, banking, and credit industries. Until retirement in 2021, he served as an instructor in the University of California, Irvine Predictive Analytics Certificate Program, teaching courses in effective data preparation and introduction to predictive analytics. Currently, he is modeling the presence of Type 1 colon polyps (precursors of colon cancer) for the UC Irvine Medical Center, using electronic medical record data. He is the lead author of the *Handbook of Statistical Analysis and Data Mining Applications* (Academic Press, 2009), a coauthor of *Practical Text Mining* (Academic Press, 2012), and coauthor of *Practical Predictive Analytics and Decisioning Systems for Medicine* (Academic Press, 2015).



KEITH McCORMICK

Keith showed an early interest in the intersection of computer science, psychology, and statistics. Studying in the Computer Science Department of Worcester Polytechnic Institute, he ultimately earned an interdisciplinary studies degree in Natural and Artificial Intelligence. With a plan to pursue his interests further with graduate work in psychometrics, Keith took a part-time job teaching statistics software workshops for SPSS Inc. That job transformed into a career. When SPSS Inc. bought the small British firm ISL, the company behind the Clementine machine learning platform (now IBM SPSS Modeler) in 1999, he became an early adopter of both Clementine and CRISP-DM. Over many years, he taught thousands of attendees from hundreds of organizations how to apply data mining techniques. In 2009, IBM bought SPSS Inc., and Keith's career shifted toward consulting.

He's coauthored several books, notably the *IBM SPSS Modeler Cookbook* (2013), *SPSS Statistics for Dummies*, fourth edition (2020), and *SPSS Statistics Workbook for Dummies* (2023). Keith is a frequent conference speaker and a prolific contributor to the LinkedIn Learning platform, with more than 20 courses and over 800,000 learners. Keith is currently revising his courses in the Predictive Analytics Certificate program at the UC Irvine Department of Continuing Education, where he intends to use this book as the text. Keith also serves as Executive Data Scientist in Residence for the AI consultancy Pandata.



Contact information for coauthor Keith McCormick, who is happy to answer readers' questions about topics in this book:

Data Science Principal, Further; Instructor, Curriculum Developer, and Advisory Board Member, Predictive Analytics Certificate Program, University of California, Irvine Division of Continuing Education



Recognized Analytics Leader

TEL: +1.919.673.7108 | WEBSITE: keithmccormick.com

INSTRUCTION: [Linkedin.com/learning/instructors/keith-mccormick](https://www.linkedin.com/learning/instructors/keith-mccormick)

BOOKS: Bookshop.org/shop/mccormick | TWITTER: [@kmccormickblog](https://twitter.com/kmccormickblog)

[linkedin.com/in/keithmccormick/](https://www.linkedin.com/in/keithmccormick/)

keith@keithmccormick.com

about.me/keithmccormick

GARY MINER, Ph.D.

Dr. Gary Miner received a BS from Hamline University, St. Paul, Minnesota, with biology, chemistry, and education majors; an MS in zoology and population genetics from the University of Wyoming; and a PhD in biochemical genetics from the University of Kansas as the recipient of a NASA predoctoral fellowship. He pursued additional National Institutes of Health postdoctoral studies at the University of Minnesota and University of Iowa, eventually becoming immersed in the study of affective disorders and Alzheimer disease.

In 1985, he and his wife, Dr. Linda Winters-Miner, founded the Familial Alzheimer's Disease Research Foundation, which became a leading force in organizing both local and international scientific meetings, bringing together all the leaders in the field of genetics of Alzheimer's from several countries, resulting in the first major book on the genetics of Alzheimer disease. In the mid-1990s, Dr. Miner turned his data analysis interests to the business world, joining the team at StatSoft and deciding to specialize in data mining. He started developing what eventually became the *Handbook of Statistical Analysis and Data Mining Applications* (coauthored with Dr. Robert Nisbet and Dr. John Elder), which received the 2009 American Publishers Award for Professional and Scholarly Excellence (PROSE). Their follow-up collaboration, *Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications*, also received a PROSE award in February 2013. Gary was also the coauthor of *Practical Predictive Analytics and Decisioning Systems for Medicine* (Academic Press, 2015). Overall, Dr. Miner's career has focused on medicine and health issues and the use of data analytics (statistics and predictive analytics) in analyzing medical data to distinguish fact from fiction.

Gary has also served as a merit reviewer for Patient-Centered Outcomes Research Institute (PCORI), which awards grants for predictive analytics research into the comparative effectiveness and heterogeneous treatment effects of medical interventions including drugs among different genetic groups of patients; additionally, he teaches online classes in "introduction to predictive analytics," "text analytics," "risk analytics," and "healthcare predictive analytics" for the University of California, Irvine. Until his "official retirement," he spent most of his time in his primary role as senior analyst/healthcare applications specialist for Information Management Group, Dell Software (through Dell's acquisition of StatSoft [www.StatSoft.com] in April 2014). During the past few years, Gary has continued to write and has been coauthor on a three-part book series called *It's All Analytics!* published by Routledge Press, Taylor & Francis, the last of these three released on September 15, 2023, and also published the second edition of *Practical Predictive Analytics for Medicine*, released in April, 2023 by Academic Press, Elsevier.



Introduction

This book has weathered the market in two editions, published in 2009 and 2017, respectively. You might wonder why a third edition is appropriate to serve the Data Science book market. The reason is that so much has happened in Data Science since 2016, when the second edition was written.

- Rise of Deep Learning
- Recommender systems
- Sports analytics
- Development of new decision tree-based methods
- Development of metabolomics and genomics modeling applications
- Development of Large Language Models like ChatGPT-3

Often, data analysts are asked, “What are statistical analysis and data analytics?” In this book, we will define what data analytics is from a procedural standpoint. But most people have a hard time relating what we tell them to the things they know and understand. Before moving on into the book, we would like to provide a little background for data analytics that everyone can relate to. The preface describes the many changes in activities related to data analytics since the first edition of this book was published in 2009. Now it is time to dig deeper and discuss the differences between statistical analysis and data analytics (aka predictive analytics).

Statistical analysis and data analytics are two methods for simulating the unconscious operations that occur in the human brain to provide a rationale for decision making and actions. Statistical analysis is a very directed rationale that is based on norms. We all think and make decisions on the basis of norms. For example, we consider (unconsciously) what the norm is for dress in a certain situation. Also, we consider the acceptable range of variation in dress styles in our culture. Based on these two concepts, the norm and the variation around that norm, we render judgments like, “That person is inappropriately dressed.” Using similar concepts of mean and standard deviation, statistical analysis proceeds in a very logical way to make very similar judgments (in principle). On the other hand, data analytics learns case by case and does not use means or standard deviations. Data analytics algorithms build patterns, clarifying the pattern as each case is submitted for processing. These are two very different ways of arriving at the same conclusion: a decision. We will introduce some basic analytical history and theory in [Chapters 1 and 2](#). The basic process of analytical modeling is presented in [Chapter 3](#). But it may be difficult for you to relate what is happening in the process without some sort of tie to the real world that you know and enjoy. In many ways, the decisions served by analytical modeling are similar to those we make every day. While [Chapter 3](#) outlines the way that decisions are reached in data analytics operations, [Chapter 4](#) will describe some of the common analytical tools used to perform them.

The background and history of predictive analytics

Preamble

You must be interested in learning how to practice data analytics; otherwise, you would not be reading this book. We know that there are many books available that will give a good introduction to the process of data analytics. Most books on data analytics focus on the features and functions of various data analytics tools or algorithms. Some books focus on the challenges of performing data analytics tasks. This book is designed to give you an introduction to the practice of data analytics in the real world of business.

General considerations in building a business data analytics capacity

Several general considerations in building a business's data analytics capability are forming the data analytics team, building the data analytics platform, and forming a foundation of good data analytics practice. This book will not discuss the building of the data analytics platform. This subject is discussed in many other books, some in great detail. A good overview of how to build a data analytics platform is presented in *Data Analytics: Concepts and Techniques* (Han and Kamber, 2006). The primary focus of this book is to present a practical approach to building cost-effective data analytics models aimed at increasing company profitability, using tutorials and demo versions of common data analytics tools. There are, however, other general considerations that should be kept in mind:

- Background of analytical methods development
- Terminology
- Tool selection

Background of analytical methods development

Just as important as these considerations in practice is the background against which they must be performed. We must not imagine that the background doesn't matter... it does matter, whether or not we recognize it initially. The reason it matters is that the capabilities of statistical and data analytics methodology were not developed in a vacuum. Analytic methodology was developed in the context of prevailing statistical and analytic theory. However, the major driver in this development was a very pressing need to provide a simple and repeatable analysis methodology in medical science. From this beginning, modern statistical analysis and data analytics were developed. To understand the strengths and limitations of this body of methodology and use it effectively, we must understand the strengths and limitations of the statistical theory from which they developed. This theory was developed by scientists and mathematicians who brought together previous thinking and combined it with original thinking to bring structure to it. However, this thinking was not one-sided or unidirectional; there arose several views on how to solve analytic problems. To understand how to approach the solving of an analytic problem, we must understand the different ways different people tend to think. This history of statistical theory behind the development of various statistical techniques bears strongly on the ability of the technique to serve the tasks of a data analytics project.

Terminology: data analytics or predictive analytics?

This discipline used to be known as "data analytics" or "knowledge discovery in databases (KDD)," and still is, to some extent. The distinction between these classical terms was proposed by Fayyad et al. (1996) for data

analytics as the application of mathematical algorithms for extracting patterns from data, while KDD includes many other process steps before and after the pattern recognition operations. Currently, however, the vast majority of practitioners have traded those terms for the common term “predictive analytics” (PA). The new term was popularized initially by the fine conferences started by Eric Segal, Predictive Analytics World (PAW). In this book, we will use the terms PA and data analytics synonymously, recognizing that the latter term broadened in the meantime to include many elements of KDD. In addition to data analytics and predictive analytics, a third term, “data science” has arisen, initially in academic circles, but it has spread recently to many other disciplines also. How can we keep these terms straight?

According to Smith (2016), the current working definitions of predictive analytics (he calls it “data analytics”) and data science are inadequate for most purposes; at best, the distinctions are murky. One way to dispel this murk is to consider what goals they are designed to seek. Predictive analytics seeks to provide operational insights into issues that we either know we know or know we don’t know. Predictive analytics focuses on correlative analysis and predicts relationships between known random variables or sets of data in order to identify how an event will occur in the future. For example, identifying where to sell personal power generators and store locations as a function of future weather conditions (e.g., storms). While the weather may not have caused the buying behavior, it often strongly correlates to future sales.

On the other hand, the goal of data science is to provide strategic and actionable insights into the world where we don’t know what we don’t know. For example, it might focus on trying to identify a future technology that doesn’t exist today but that might have a big impact on an organization in the future.

Smith (2016) claims that assigning the central focus of predictive analytics to operations and assigning the central focus of data science to strategy helps to understand how both activities are integrated into the enterprise information management (EIM). The EIM consists of those capabilities necessary for managing today’s large-scale data assets. In addition to relational databases, data warehouses, and data marts, we see the emergence of big data solutions (Hadoop). Both data science and predictive analytics leverage data assets to provide day-to-day operational insights into very different functions, such as counting assets and predicting inventory.

There are many books available that will give a good introduction to the process of predictive analytics, such as Larose and Larose (2015), which focuses on the mathematical algorithms and model evaluation methods. Kelleher et al. (2020) focus on machine learning techniques in general. Other books like Han and Kamber

(2006) focus on the features and functions of data analytics tools or algorithms. These books leave many users with the question, “How can I apply these techniques in my company?” This book is designed to give you an introduction to the *practice* of predictive analytics in the real world of business.

Tool selection

One of the first things considered in building a predictive analytics capability in an organization is selecting the data analytics tool. It is difficult to penetrate the hype erected around the description of these tools by the vendors. The fact is that even the most mediocre of data analytics tools can create models that are at least 90% as good as the best tools. A 90% solution performed with a relatively cheap tool might be more cost effective in your organization than a more expensive tool. How do you choose your data analytics tool? There are few deep reviews available. Some detailed reviews are available that go beyond just a discussion of their features and functions, but they are rather dated (see Elder and Abbott, 1998). A number of valuable articles on analytics tool popularity are available at <https://www.kdnuggets.com/?s=tool+popularity>.

In addition, there are many other blogs and newsletters available, which provide much useful information. Several of the important blogs include:

- Data Science Central (available through LinkedIn) (https://www.linkedin.com/groups/35222?midToken=AQGpwYhuvOEU4g&trk=eml-b2_anet_digest_weekly-hero-8-groupPost~group&trkEmail=eml-b2_anet_digest_weekly-hero-8-groupPost~group-null-q3766~ipwtz308~5b)
- Machine Learning and Data Analytics (available through LinkedIn) (<https://www.linkedin.com/groups/4298680>)
- Predictive Analytics World blog by Eric Siegel (<http://www.predictiveanalyticsworld.com/blog/>)

Phases in the history of statistics and predictive analytics

Phase I: Prehistory – 1000 BC to 1921 AD

Analysis of patterns in data is not new. The concepts of average and grouping can be dated back to about 1000 BC in China (Daobin, 1999). In ancient China and Greece, statistics were gathered to help heads of state govern their countries in fiscal and military matters. These official activities point to the likelihood that the words “statistic” and “state” evolved from the same root. In the 16th and 17th centuries, games of chance were popular among the wealthy, prompting many questions about

probability to be addressed to famous mathematicians (Fermat, Leibnitz, etc.). These questions led to much research in mathematics and statistics during the ensuing years.

Two branches of statistical analysis developed in the 18th century, Bayesian and classical statistics. To treat both branches fairly in the context of history, both will be considered in the first phase of development of statistical analysis.

Bayesian analysis

Thomas Bayes was an 18th-century theologian and philosopher (Fig. 1.1).

At that time, it was primarily the theologians and philosophers who had enough time to speculate on mathematical topics. Bayes believed that the probability of an event's occurrence in the future is equal to the

probability of its past occurrence divided by the probability of all competing events. Analysis proceeds based on the concept of *conditional probability*: the probability of an event occurring given that another event has already occurred (past events). Bayesian analysis begins with the quantification of the investigator's existing state of knowledge, beliefs, and assumptions about past events. These *subjective priors* are combined with observed data in a current experiment quantified probabilistically through an objective function of some sort.

Bayes' theorem is stated mathematically as shown in Eq. (1.1) below:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (1.1)$$

where A is the event in question (see event) and B is the combined probability of all competing events (for $P(B) \neq 0$).

- $P(A)$ and $P(B)$ are the probabilities of observing A and B independently.
 - $P(B)$ represents the combined probabilities of all competing events.
- $P(A|B)$ is the conditional probability of observing event A , given that B is true.
- $P(B|A)$ is the conditional probability of observing event B , given that A is true.

Posed as a word problem, the conditional probability of the occurrence of event A (given that the combined probability of prior events (B) has already occurred) is equal to the product of the probability of B occurring, given that A has occurred times the probability of A , the quantity then being divided by the probability of B occurring.

Classical statistical analysis

Interest in probability picked up early among biologists following Mendel in the latter part of the 19th century. Sir Francis Galton, founder of the School of Eugenics in England, and his successor, Karl Pearson, developed the concepts of regression and correlation for analyzing genetic data. Their system required the calculation of



FIG. 1.1 Rev. Thomas Bayes (1702–1761).

Example - cancer at age 65

Consider that the general prevalence of cancer is 2%. This is known as the “base rate” or the prior probability of having cancer (before knowing that a person has cancer). Assuming that cancer and age are related, we can calculate the probability that a person has cancer at age 65, called the “current probability.” Then, suppose that the probability of being 65 years old is 0.3% and that the probability for someone diagnosed with cancer to be

65 is 0.4%. Knowing this, along with the base rate of 2%, we can calculate the probability of having cancer as a 65-year-old, which is equal to the probability of being 65 years of age while having cancer, times the probability of having cancer, divided by the probability of being 65 years of age:

$$(0.004 \times 0.02) / 0.003 = 0.02667 \text{ or } 2.667\%$$

two parameters: the mean (average value for each variable) and the standard deviation (average difference between each data point and the mean). Their system also required several assumptions about the data collected to be satisfied (see discussions of linearity, normality, and independency below). Later, Pearson and colleagues extended their work to the social sciences.

Nonparametric analysis

Even before Fisher developed his statistical analysis system, other statisticians began to build on the work of Galton and Pearson along a different path in the early 1900s. Instead of making assumptions of the experimental conditions from which analysis data are drawn therefore, they did not need to calculate means and standard deviations. They developed nonparametric methods of analyzing data, which made these assumptions unnecessary. Spearman developed his approach to correlation analysis that was not based on parameters like the mean and standard deviation values (Spearman, 1904). Spearman used the concept of ranking to measure the correlation between two datasets to express monotonic relationships, whether they were linear or nonlinear. During the next 50 years, many other nonparametric techniques were developed to assess relationships between two datasets (e.g., Wilcoxon Rank Sum test, the Mann-Whitney U test, and the Kruskal-Wallis one-way analysis of variance).

While the development of probability theory and correlation analysis (both parametric and nonparametric) flowed out of the work of Galton and Pearson, early predictive analytics methods followed Bayes' approach. A major concern, however, was that Bayesian approaches to inference testing could lead to widely different conclusions by different medical investigators, if they used different sets of prior probabilities. This set of prior probabilities included in the calculations of Bayes Rule were subjectively selected and referred to as *subjective priors*.

Phase II: Modern statistical analysis – 1922 – 1958

Sir Ronald Fisher: a turning point

The use of prior probabilities was a source of concern for both Pearson and his young protégé, Ronald Fisher.

"Karl Pearson was ambivalent about few things, but Bayes' rule was was of them. Uniform priors and subjectivity made him nervous. With few other tools available to statisticians, however he concluded sadly that 'the practical man will ... accept the results of inverse probability of the Bayes-Laplace brand till better are forthcoming.'" (McGrayne, 2011)

This situation bothered Ronald Fisher greatly, and he was committed to developing alternatives. Fisher is attributed by author Sharon Bertch McGrayne in *The Theory That Would Not Die* with dealing Bayes theorem

"a near-lethal blow" by using his eventual influence to make the Bayes approach disreputable in academia (McGrayne, 2011). He is also the father of modern statistics. As a close colleague of Pearson, Fisher's early academic work was also in eugenics, but while still in his 20s, Fisher had an elaborate and contentious misunderstanding over an academic article with Pearson that led to a falling out. As two strong-willed personalities, they never completely reconciled. In 1919, Fisher and his family moved to Rothemsted, where he took a position researching plant genetics and crop yields. As an agronomist there, he invented critical statistical techniques like ANOVA that are still in use today. For this reason, he is widely considered the most influential statistician of the 20th century. With him at Rothemsted was his young daughter, Joan, who would become both his biographer and the wife of the famous statistician George Box.

Rothemsted Experimental Station, already 70 years old when Fisher arrived, was dedicated to studying the crop yields of winter wheat and other crops. It was Fisher's job to take this data, including decades of existing, poorly organized data, and systematically perform difficult experiments on plots of land: their conditions, their soil types, and the use of fertilizer (which was in its infancy at the time). The modern technique of "split plot ANOVA" is named after its original use, comparing plots of land at Rothemsted.

"Fisher realized that he had what is now called a split-plot arrangement in which two different variances must be estimated, one the variance between patches within a plot, the other the variances between plots. The first was appropriate for possible differences and interactions involving potash dressings, the second for possible differences and interactions involving varieties of potato." (Box, 1978)

Fisher eventually stayed at Rothemsted for 14 years, but by 1925, he had already accomplished the invention of many of the techniques for which he was famous. He had learned to overcome many challenges, including dealing with small sample sizes and the careful design of experiments. He described his work in *Statistical Methods for Research Workers* (1925), and a decade later in the *Design of Experiments* (1935). Together, these books were so influential that it was nearly impossible to be a successful academic without adopting both Fisher's philosophy and his technique to the letter, which pushed Bayes out of academia for decades.

*"Fisher was really the first person to try seriously to excise Bayesian inference from statistics altogether. The generation of students who came through his lab at Rothamsted and the many more who learned statistics from his textbook didn't learn the Bayesian method at all. In *Statistical Methods for Research Workers*, Fisher wrote that 'inverse probability [that is, Bayesian inference] is founded upon an error, and must be wholly rejected.'"* (Aubrey, 2021)

Fisher developed a system for inference testing based on the concept of standard deviation, which is the average difference between all data values and the mean (average) value (analysis of variance). The classical statistical approach of Fisher (that flowed out of mathematical works of Gauss and Laplace) considered that the joint probability, rather than the conditional probability of the Bayesians, was the appropriate basis for analysis. The joint probability function expresses the probability that X takes the specific value x and Y takes the value y as a function of x and y jointly. There are *no* subjective priors in this calculation, accepting only those data that can be collected at the time of an experiment. This means that only data captured in a given experiment could be used to predict an outcome. This requirement is the primary distinction of Fisher's approach to that of the Bayesians'. Fisher's goal in developing his system of statistical inference was to provide researchers with a common set of tools for use in comparison with studies of effects of different treatments by different investigators. But in order to make his system work even with large samples, Fisher had to make a number of assumptions to define his "parametric model."

The parametric model of Sir Ronald Fisher

In his landmark paper (Fisher, 1922), Fisher (Fig. 1.2) began with the broad definition of probability as the intrinsic probability of an event's occurrence divided by the probability of occurrence of all competing events (very Bayesian). By the end of his paper, Fisher modified his definition of probability as the intrinsic probability of an event's occurrence *period*. This intrinsic probability was calculated from data from only a current experiment. He named this quantity *likelihood*. From that foundation, he developed the concept of standard deviation based on the normal distribution. Those that followed Fisher began to refer to likelihood as probability, thus



FIG. 1.2 Sir Ronald Fisher.

giving rise to two general theories of probability. The concept of likelihood approaches the classical concept of probability only as the sample size becomes very large and the effects of subjective priors approach zero (von Mises, 1957). In practice, these two conditions may be satisfied sufficiently if the initial distribution of the data is known and the sample size is relatively large (following the law of large numbers). This is true only because a relatively large sample size is likely to include data that were a part of prior experiments; therefore the effect of other prior experimental data approaches zero. For his important contribution to the theory and practice of statistical analysis, Ronald Fisher (Fig. 1.2) was knighted in England and is referred to as Sir Ronald Fisher today.

Assumptions of the parametric model

Data fits a known distribution (e.g., normal, logistic, and Poisson)

Fisher's early work was based on calculation of the parameter, *standard deviation*, which assumes that data are distributed in a *normal* distribution. The normal distribution is bell shaped, with the mean (average) at the top of the bell, with "tails" falling off evenly at the sides (Fig. 1.3).

Standard deviation of a variable is the square root of the quantity – the sum of all absolute deviations of all values from the mean squared divided by the total count of the data points ($n - 1$), as shown in Eq. (1.2).

$$S = \sqrt{\frac{\sum (x - \bar{x})^2}{n - 1}} \quad (1.2)$$

where x is the value of one data point, $\bar{x}$ is the mean, and n is the total number of data points. The subtraction of 1 from the total number expresses (to some extent) the increased uncertainty of the result due to grouping (summing the squared deviations). Subsequent developments used modified parameters based on the logistic and Poisson distributions (logistic regression and Poisson regression). The assumption that data follow a particular known distribution is necessary in order to draw upon the characteristics of the distribution function for making

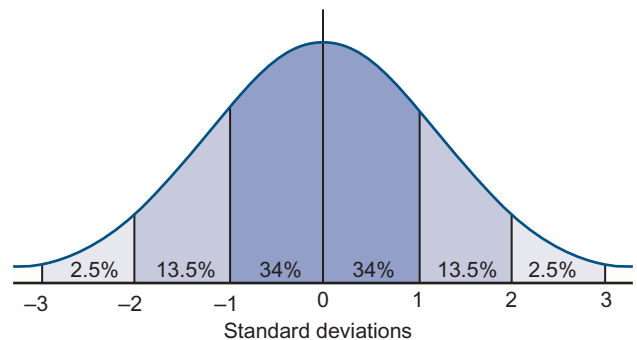


FIG. 1.3 The normal curve showing that 95% ($100 - 2.5 - 2.5$) of the data are found between -2 and $+2$ standard deviations from the mean.

inferences. All of these *parametric* methods run the gauntlet of dangers related to force-fitting data from the real world into a mathematical construct that may not fit.

Factor independency

In parametric predictive systems, the variable to be predicted (Y) is considered as a function of predictor variables (X s) that are assumed to have independent effects on Y . That is, the effect on Y of each X variable is not dependent on the effects on Y of any other X variable. This situation could be created in the laboratory by allowing only one factor (e.g., a treatment) to vary while keeping all other factors constant (e.g., temperature, moisture, and light). But in the real world, such laboratory control is not possible. As a result, it must be possible to consider situations in which some factors do affect other factors, that is, have a joint effect on Y . This problem is called *collinearity*. When it occurs between more than two factors, it is termed *multicollinearity*. The multicollinearity problem led statisticians to include an interaction term in the relationship that supposedly represented the combined effects. Use of this interaction term functioned as a magnificent kluge, and the reality of its effects was seldom analyzed. Later development included a number of interaction terms, one for each interaction the investigator thought might be present.

Linear additivity

Not only must the X variables be independent in the parametric model, but also their effects on Y must be cumulative and linear. That means that the effect of each factor is added to or subtracted from the combined effects of all X variables on Y . But what if the relationship between Y and predictors (X variables) is not additive but multiplicative or divisive? This is the case in modeling forests. General effects of light, moisture, and nutrients must be multiplied together (not added) to relate to tree growth. Such functions can only be expressed by exponential equations that usually generate very nonlinear relationships. Assuming linear additivity for these relationships in the natural world would cause large errors in the predicted values for tree growth (Botkin, 1993). This is often the case with their use in business data systems.

Constant variance (homoscedasticity)

The variance throughout the range of each variable is assumed to be constant. This means that if you divide the range of a variable into bins, the variance across all records for bin #1 is similar to the variance for all the other bins of that variable. If the variance throughout the range of a variable differs significantly from constancy, it is said to be *heteroscedastic*. The error in the predicted value caused by the combined heteroscedasticity among all variables can be quite significant.

Variables must be numerical and continuous

This assumption means that data must be numeric (or it must be transformable to a number before analysis) and the number must be part of a distribution that is inherently *continuous*. Integer values are not continuous; they are *discrete* (e.g., there are no integers between 1 and 2; hence, 1.3 is not an integer). Classical parametric statistical methods are not valid for use with discrete data because the probability distributions for continuous and discrete data are different. Still, scientists and business analysts have used them anyway. Problems will be greatest where the approximation to continuous data is not close.

Duality of thinking

The argument between the Bayesians and the Fisherians continues today; it represents a duality of thinking. The most important aspect of this discussion is that the Bayesian approach may prove to be the most practical way to predict outcomes in the business world. The reason is related to the common mantra in predictive analytics that the best guide for responses of people in the future is the record of past responses; human nature does not change. We should not restrict our consideration of past behavior to those variables measured as a part of the current project.

Consider again Eq. (1.1) (Bayes' rule). If we are interested only in finding the most probable outcome among different options of A , we can ignore the $P(B)$ factor in the denominator because it functions only as a normalizing factor. This means that we can express Eq. (1.3) for purposes of predictive analytics as the maximum of the numerator factors over all instances of i as,

$$\text{MAX of } P(E | T_i)P(T_i) \quad (1.3)$$

This concept has very important implications for prediction. Fisher's approach works in the laboratory to express an outcome on the basis of past information, but it may not reflect very well what will happen in the future in practical situations because not all of the relevant past information can be included in the analysis. Granted, Fisher's restriction to consider only the data provided by the current analysis controlled carefully in the laboratory is convenient to prevent different conclusions to be generated by different investigators using different sets of subjective priors. Yet for prediction in the real world, where nothing is controlled, the Bayesian approach may prove to be more practical.

Westheimer (2008) remarked that Helmholtz argued in his 1878 treatise *The Fact of Perception* (von Helmholtz, 1878) that our perception of physical phenomena is a product of *unconscious inference* from the combination of sensed data and prior experience (Westheimer, 2008). Much of the science of thermodynamics today is based on this notion, and it is very Bayesian. In the real world

of decision making in business and industry, the approach to predicting future events in managers' minds follows the same path. The approach of most people to analytics today appears to be more related to Fisher's ideas than to those of Bayes. This situation in predictive analytics may change in the future.

Why did this duality of thought arise in the development of statistics? There seems to be a broader duality, however, that pervades all of human thinking, which we can trace back to the ancient debate between Plato and Aristotle.

Two views of reality that underlay the duality in thinking in modern statistical analysis

Whenever we consider solving a problem or answering a question, we start by conceptualizing it. That means that we do one of two things: (1) try to reduce it to key elements or (2) try to conceive it in general terms. We call people who take each of these approaches "detail people" and "big picture people," respectively. What we don't consider is that this distinction has its roots deep in Greek philosophy in the works of Aristotle and Plato.

Aristotle

Aristotle (Fig. 1.4) believed that the true being of things (reality) could be discerned only by what we can perceive with our bodily senses (what the eye could see, the hand could touch, etc.). He believed that the highest level of intellectual activity was the detailed study of the tangible world around us. Only in that way could we understand reality. Based on this approach to truth, Aristotle was led to believe that you could break down a complex system into pieces, describe the pieces in detail, put the pieces together, and understand the whole. For Aristotle, the "whole" was equal to the sum of its tangible parts. This nature of the "whole" was viewed by Aristotle in a manner that was very *machinelike*.

Science gravitated toward Aristotle very early. The nature of the world around us was studied by looking very

closely at the physical elements and biological units (species) that composed it as if they functioned like machines. As our understanding of the natural world matured into the concept of the ecosystem, it was discovered that many characteristics of ecosystems could not be explained by traditional (Aristotelian) approaches. For example, in the science of forestry, we discovered that when you cut down a tropical rainforest on the periphery of its range, it may take a very long time to regenerate (if it does at all). We learned that the reason is that in areas of relative stress (e.g., peripheral areas), the primary characteristics necessary for the survival and growth of tropical trees are maintained by the forest itself! High rainfall leaches nutrients down beyond the reach of the tree roots, so almost all of the nutrients for tree growth must come from recently fallen leaves and branches. When you cut down the forest, you remove that source of nutrients. The forest canopy also maintains favorable conditions of light, moisture, and temperature required by young trees. Removing the forest removes the very factors necessary for it to continue to exist in that location. These factors *emerge* only when the system is whole and functioning. Many complex systems are like that, even business systems. In fact, these *emergent properties* may be the major drivers of system stability and predictability. In order to understand the failure of Aristotelian philosophy in completely defining the world, we must turn to his rival, Plato.

Plato

Plato (Fig. 1.5) was Aristotle's teacher for 20 years, and they both agreed to disagree on the nature of being. While Aristotle focused on describing tangible things in the world by detailed studies, Plato focused on the world of *ideas* that lay behind these tangibles. For Plato, the only thing that had lasting being was an *idea*. He believed that the most important things in human existence were beyond what the eye could see and the hand could touch. Plato believed that the influence of ideas (e.g., love, hate, and fear) transcended the world of tangible things that commanded so much of Aristotle's interest.



FIG. 1.4 Aristotle before the bust of Homer.



FIG. 1.5 Plato pointing up to signify the importance of ideas.

For Plato, the “whole” of reality was *greater* than the sum of its tangible parts.

The concept of the nature of being was developed initially in Western thinking upon a Platonic foundation. Platonism ruled philosophy for over 2000 years – up to the Enlightenment. Then the tide of Western thinking turned toward Aristotle. This division of thought on the nature of reality continued, however, and is reflected in our debates between “big picture” and “detail people” or “top-down” approaches to organization versus “bottom-up,” or “left-brained” people versus “right-brained.” These dichotomies of perception are rehashes of the ancient debate between Plato and Aristotle.

The distinction between the two views of reality also guided the approaches of Bayes and Fisher: Bayes was a big-picture thinker, and Fisher was a detailed thinker. The approach of Fisher was extended to analyzing relationships in many data domains between 1921 and the 1960s. However, it wasn’t until the introduction of early artificial neural network (ANN) algorithms in 1957 that the two views of reality began to converge. First, we will consider the spread of Fisher’s medical analysis techniques to many other data domains. With this context in mind, we can understand the trends of thinking that led to Phase III: Machine Learning.

The spread of modern statistical analysis

Many scientists adopted Fisher’s techniques and applied them in many data domains. One of the most successful application areas in the 1930s and 1940s was in Agronomy. By the 1960s, most of the examples in statistical analysis textbooks were taken from the science of Agronomy (e.g., [Steele and Torrie, 1960](#)). In the 1980s, it became obvious to statistical mathematicians that the rigorous Aristotelian approach of the past was too restrictive for analyzing highly nonlinear relationships in large datasets in complex systems of the real world. Mathematical research continued dominantly along Fisherian statistical lines by developing nonlinear versions of parametric methods. Multiple curvilinear regression was one of the earliest approaches for accounting for nonlinearity in continuous data distributions. But many nonlinear problems involved discrete, rather than continuous, distributions (see [Agresti, 1996](#)). Some of these methods are listed below:

- Logit model (including logistic regression): data are assumed to follow a logistic distribution, and the dependent variable is categorical (e.g., 1:0). In this method, the dependent variable (Y) is defined as an exponential natural log function of the predictor variables (X s). As such, this relationship can account for nonlinearities in the response of the X variables to the Y variable but not in the interaction between X variables.
- Probit model (including Poisson regression): it is similar to the logit model, except that the data come from counts of things (integers) and are assumed to follow a Poisson, rather than a logistic, distribution.
- The generalized linear model (GLM): the GLM expands the general estimation equation used in prediction, $Y = f\{X\}$, where f is some function and X is a vector of predictor variables. The left side of the equal sign was named as the *deterministic component*, the right side of the equation as the *random component*, and the equal sign as one of many possible *link functions*. Statisticians recognized that the deterministic component could be expressed as an exponential function (like the logistic function), the random component accumulated effects of the X variables and was still linear, and the link function could be any logical operator (equal to, greater than, less than, etc.). The equal sign was named the *identity link*.
- Now, mathematicians have a framework for defining a function that could fit datasets with much more nonlinearity. But it would be left to the development of neural networks (see below) to express functions with any degree of nonlinearity.

While these developments were happening in the Fisherian world, a stubborn group of Bayesians continued to push their approach. To the Bayesians, the practical significance (related to what happened in the past) is more significant than the statistical significance calculated from joint probability functions. For example, the practical need to correctly diagnose cancerous tumors (true positives) is more important than the error of misdiagnosing a tumor as cancerous when it is not (false positives). To this extent, their focus was rather Platonic, relating correct diagnosis to the data environment from which any particular sample was drawn, rather than just to data of the sample alone. In order to serve this practical need, however, they had to ignore the fact that you can consider only the probability of events that actually happened in the past data environment, not the probability of events that *could* have happened but did not ([Lee, 1989](#)).

In Fisherian statistics, the *observation* and the corresponding *alpha error* determine whether it is different from what is expected or not ([Newton and Rudestam, 1999](#)). The *alpha error* is the probability of being wrong when you think you are right, while *beta error* is the probability of being right when you think you are wrong. Fisherians set the alpha error in the beginning of the analysis and referred to significant differences between data populations in terms of the alpha error that was specified. Fisherians would add a suffix phrase to their prediction, such as “... at the 95% confidence level.” The confidence level (95% in this case) is the complement of the alpha error (0.05). It means that the investigator is

willing to be right only 95% of the time. Fisherians use the beta error to calculate the “power” or “robustness” of an analytic test. Bayesians feel free to twiddle with both the alpha and beta errors and contend that you cannot arrive at a true decision without considering the alternatives carefully. They maintain that a calculated probability level of 0.023 for a given event in the sample data does not imply that the probability of the event within the entire universe of events is 0.023.

Which approach is right, Fisherian or Bayesian? The answer depends on the nature of the study, the possibility of considering priors, and the relative cost of false-positive errors and false-negative errors. Before one is selected, we must bear in mind that all statistical tests have advantages and disadvantages. We must be informed about the strengths and weaknesses of both approaches and have a clear understanding of the meaning of the results produced by either one. Regardless of its problems and its “bad press” among the Fisherians, Bayesian statistics eventually did find its niche in the developing field of data analytics in business in the form of Bayesian belief networks and Bayes classifiers. In the 21st century, Bayes is experiencing a renaissance and Fisherian statistics have been increasingly under attack as some, fairly or not, attribute the “reproducibility crisis” to the misapplication of Fisher’s p-values (Aubrey, 2021). In business, success in practical applications depends to a great degree upon the analysis of all *viable* alternatives. Nonviable alternatives aren’t worth considering.

Data, data everywhere

The crushing practical needs of businesses to extract knowledge from data that could be leveraged immediately to increase revenues required new analytic techniques that enabled analysis of highly nonlinear relationships in very large datasets with unknown distributions. Development of new techniques followed *three* paths, rather than the two classical paths described above. The third path (machine learning) might be viewed as a blend of the Aristotelian and Platonic approaches to truth, but it was not Bayesian.

Phase III: The rise of machine learning methods – 1957 to present

The turning point represented by revolutionary methods of Fisher (1922) was followed in 1957 by an article by the psychologist Frank Rosenblatt (1958), which presented the notion of machine learning that arose out of the artificial intelligence (AI) community in the quest for the intelligent machine. Initially, machine learning methods followed two parallel pathways of developments, ANNs and decision trees. We can view the concept of the perceptron as another

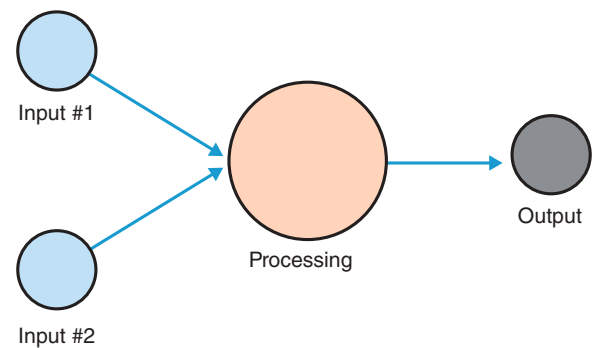


FIG. 1.6 The perceptron as conceived by Rosenblatt (1958), consisting of two input nodes (variables), a processing node (*tan*) and an output node (*gray*).

turning point, which led to the development of a many kinds of new analytical developments.

The perceptron

Rosenblatt introduced the concept of the perceptron, in which he believed that data should be analyzed the way the biological neuron does it. This concept consisted of one or more inputs, a processing operation, and one output (Fig. 1.6).

Rosenblatt’s initial configuration of the logic in the processing node of Fig. 1.6 consisted of a simple linear function to accomplish two tasks: (1) accumulate data from all inputs and (2) aggregate the accumulated input values to yield the output value, mimicking the basic function of a biological neuron. Unlike the biological neuron, the output of the perceptron was not very accurate as a predictor. Later, Rosenblatt reasoned that the initial architecture of the perceptron did not evaluate the nonlinear effects of the input nodes on the output node. He decided to expand the processing operation to include a number of nodes between the inputs and output, to express the effects of “internuncial” nodes, which he viewed as analogous to the neurons involved with signal processing in humans between sensor neurons (inputs) and motor neurons (outputs). Rosenblatt (1962) formalized this extended network architecture to form the multilayer perceptron (MLP). Later, the MLP algorithm was renamed as the ANN.

The multilayer perceptron

The original perceptron sought to express a nonlinear function directly (the “cause”) by means of assigning weights to the input variables, accumulate their effects, which “reacted” to produce an output value (the “effect”), following some sort of decision function. The perceptron represented a very simple analog of the way the human brain works by passing neural impulses from neuron to neuron across synapses (gaps between neurons). These gaps represent resistances to the flow of neural impulses across them. This resistance (constituted

by the length of the gap) in transmission of an impulse between two neurons in the human brain is variable, and that is how humans learn. The complex relationship of neurons and their associated synaptic connections is “trainable” and could “learn” to respond faster as required by the brain. Computer scientists began to express this sort of system in very crude terms in the form by expanding the central processing operation of the perceptron to form a layer of processing nodes between the input layer and the output layer of nodes. This internal (or “hidden”) layer was connected with the input and output nodes and associated with weights with an effect of the resistances to neural impulse flow across gaps between neurons. The body of hidden layer nodes and their weights were assumed to express the overall nonlinear relationship to the target variable. This architecture was termed the “multilayer perceptron” (MLP), and scientists found that the assumption associated with the hidden layer architecture could be leveraged to “learn” how to recognize complex nonlinear patterns in the input variables of a dataset.

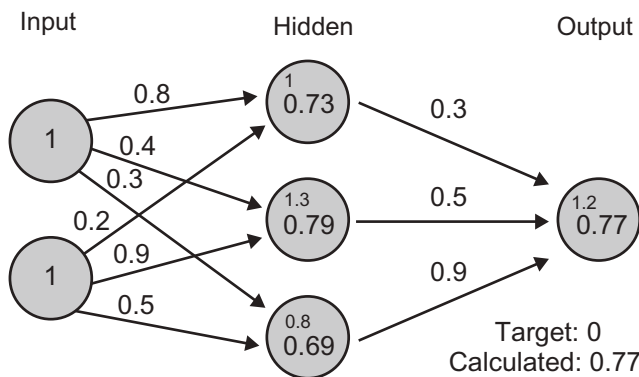


FIG. 1.7 A three-layer artificial neural network (ANN) showing the weights assigned to the connections.

The weights shown in the connections in Fig. 1.7 are the mathematical representations of resistances in the flow of impulses in the human neural network. We will dig deeper into how this structure works later in the chapter under the topic of back-propagation.

Decision trees

The second pathway of development was concerned with expressing the *effects* directly by developing methods to find “rules” that could be evaluated for separating the input values into one of several “bins” without having to express the functional relationship directly. These methods focused on expressing the rules explicitly (rule induction) or on expressing the relationship among the rules (decision tree, DT) that expressed the results. These methods avoided the assumptions of the parametric model and were well suited for analysis of nonlinear events (NLEs), both in terms of combined effects of the X variables with the Y variable and interactions between the independent variables (Fig. 1.8).

While decision trees and neural networks could express NLEs more completely than parametric statistical methods, they were still intrinsically additive in their aggregation functions.

Statistical learning theory

Logistic regression techniques can account for the *combined* effects of nonlinear relationships between all predictor variables by virtue of the nonlinear function that defines the dependent variable (Y). Yet there are still significant limitations to these linear learning machines (see [Minsky and Papert, 1969](#)). Even neural networks and decision trees suffered from this problem to some extent. One way of expressing these limitations is to view them according to their “hypothesis space.” The hypothesis space is a mathematical construct within which a solution is sought. However, this space of possible solutions may

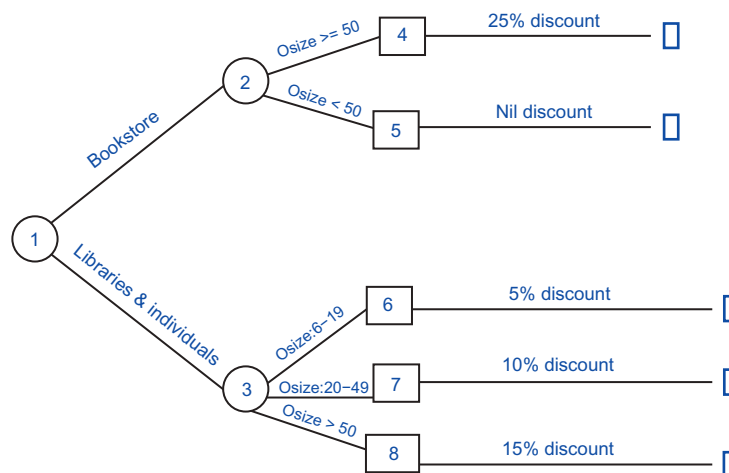


FIG. 1.8 A decision tree to determine the discount price for a customer.

be highly constrained by the linear functions in classical statistical analysis and machine-learning techniques. Complex problems in the real world may require much more expressive hypothesis spaces than can be provided by linear functions (Cristianini and Shawe-Taylor, 2000). Multilayer neural networks can account for much more of the nonlinear effects by virtue of the network architecture when using an effective error minimization technique (e.g., back-propagation). A better method was needed, however, to express sufficiently the highly nonlinear effects in many real-world datasets. This need led to the development of statistical learning theory.

An alternative approach to learning with a machine is to arrange input data points into vectors (like rows in a customer record). Such vectors are composed of elements (one for each attribute in the customer record). The vector space of all rows of customers in a database can be characterized conceptually and mathematically as a space with the N dimensions, where N is the number of customer attributes (predictive variables). When you view data in a customer record as a vector, you can take advantage of linear algebra concepts, one of which is that you can express all of the differences between the attributes of two customer records by calculating the *dot product* (or the inner product). The dot product of two vectors is the sum of all the products between corresponding attributes of the two vectors. Consequently, we can express our data as a series of dot products composed into an inner product space with N dimensions. Conversion of our data into inner products is referred to as “mapping” the data to an inner product space. Mapping can be accomplished by using a number of other strategies (called “kernels”), which might be more effective than that of the dot-product kernel in generating an appropriately effective decision space for a given problem.

Even classical statistical algorithms (like linear regression) can be expressed in this way. In statistical learning theory, various complex kernels replace the inner product. When you map data into these complex kernel spaces, the range of possible solutions to your problem increases significantly. The “data” in these spaces are referred to as “features,” rather than as attributes that characterized the original data. These kernel-based methods are called “kernel learning machines.”

A number of new learning techniques have taken advantage of the properties of kernel learning machines. The most common implementation is a support vector machine. When an ANN is trained, rows of customer data are fed in, and errors between predicted and observed values are calculated (an example of supervised learning). The learning function for training and the error minimization function (that defines the best approximate solution) are closely intertwined in neural networks. This is not the case with support vector machines. Because the learning process is separated

from the approximation process, you can experiment by using different kernel definitions with different learning theories. Therefore instead of choosing from among different architectures for a neural network application, you can experiment with different kernels in a support vector machine implementation.

Machine learning automation

Several commercial packages include algorithms based on statistical learning theory, notably STATISTICA Data Miner and SAP predictive analytics. In the future, we will see more of these powerful algorithms in commercial packages. The automated operation of SAP predictive analytics served as a precursor to the development of many automated predictive analytics applications, such as those listed in the KDnuggets newsletter of June 6, 2016:

- AutoDiscovery from ButlerScientifics – intelligent exploratory data analysis software that unveils complex relationships in scientific experiments or clinical studies data.
- Automatic Business Modeler from Algolytics – automatically builds accurate and interpretable predictive models. Commercial.
- Automatic Statistician project – a system that explores an open-ended space of possible statistical models to discover a good explanation of the data and then produces a detailed report with figures and natural-language text. Research project.
- DataRobot – an automated machine-learning platform built by top data scientists on Kaggle. Commercial.
- DMWay offers an automated end-to-end solution, powered by a sophisticated analytic engine that models all the steps taken by experienced data scientists during the analytic process. Commercial.
- Forecast – this DSX uses cross validation to test every algorithm in its extensive library, automates the discovery of the best model, and makes that model available for use.
- FeatureLab – built upon MIT Data Science Machine research project, automatically chooses optimal variables, builds appropriate models, and recommends the refinements best suited for your data.
- Loom Systems develops the first artificially intelligent data scientist. Built for low-touch operational simplicity, it automatically extracts value from big data and presents it along with recommended actions.
- MachineJS – automated machine learning – just give it a data file. Open-source on GitHub.
- Quill from Narrative Science transforms data into meaningful and insightful narratives people can simply read. Commercial.

- SAP predictive analytics combines SAP InfiniteInsight (as automated analytics) and SAP predictive analysis (as expert analytics) in a single desktop installation.
- Savvy from Yseop automatically turns spreadsheets and dashboards into written insight.
- Skytree machine-learning software – enterprise-grade machine learning uses all data and high-performance algorithms automated model building to deliver more accurate predictive models. Commercial.
- Tree-based Pipeline Optimization Tool (TPOT) – a Python tool that automatically creates and optimizes machine-learning pipelines using genetic programming. Research prototype.
- Xpanse Analytics – a platform building predictive models on raw transactional data stored in many tables able to generate thousands of features automatically without manual coding. Commercial.

Several open-source predictive analytics packages include SVMs also, notably KNIME, RapidMiner, and Weka.

Phase IV: Deep learning and Foundation models – 2012 to present

The rise of Reinforcement Learning

In 2012, a convergence of four factors led to a milestone event in the history of machine learning: larger datasets, faster computers, the insight to try using GPUs, and a radical increase in the number of hidden layers. We will discuss this history in more detail in [Chapter 17](#), but the milestone event was the 2012 ImageNet competition and the history-making win of the AlexNet model that year. Previous winners of the ImageNet computer vision competition had just one to three hidden layers (as we discussed in innovation of MLPs). AlexNet had 8, and massively reduced the error rate of the model compared to the previous year. Within a short time, it was common to have hundreds of hidden layers, a technique that was unprecedented before this milestone. The team was mentored by Geoff Hinton, who 6 years later shared the Turing Award, in part for this event. One of the team members, Ilya Sutskever, is credited with recognizing that GPUs could process the much larger and more complex models. Sutskever would go on to become highly influential in multiple areas, eventually including large language models (LLMs). Soon after this win, “Deep” Learning became the dominant solution in several use cases including image recognition and natural language processing (NLP). The common thread among the cases was unstructured data, areas where traditional machine learning, including “shallow” MLPs had failed. The term Deep Learning comes from the increased depth of the hidden layer thus resulting in the phrase “shallow neural

networks” used by some proponents of deep learning including Geoff Hinton to describe the older technique.

A processing innovation that flowed out of the third phase of analytics is the modification to make machine-learning algorithms more closely parallel to the way humans think. The first step introduced the concept of *reinforced learning*. A standard neural networks, for example, learns by evaluating the total prediction error across all rows and adjusting some learning parameters prior to another processing iteration through the dataset. Reinforcement learning defines a “reward” that is much more broadly related to the predictive success. For example, a good model might be judged to be one with a relatively high sensitivity (proportion of positive values predicted as positives) and a relatively high specificity (proportion of negative values predicted as negatives). These model evaluation criteria can be composed into the definition of the “reward.” The value of this reward can be used to maximize the total reward over many iterations of the learning process to produce a global maximum prediction.

When reinforcement learning was combined with highly evolved neural networks with many hidden layers, a new universe of analytical techniques emerged, which came to be known as “deep learning” algorithms. These powerful algorithms can fit outcomes to much more highly nonlinear functions.

What is deep learning?

The first comprehensive review was reported in [LeCun et al. \(2015\)](#). LeCun and Bengio shared the Turing Award with Hinton in 2018 for “conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing.”

Working independently and together, Hinton, LeCun and Bengio developed conceptual foundations for the field, identified surprising phenomena through experiments, and contributed engineering advances that demonstrated the practical advantages of deep neural networks. In recent years, deep learning methods have been responsible for astonishing breakthroughs in computer vision, speech recognition, natural language processing, and robotics—among other applications. While the use of artificial neural networks as a tool to help computers recognize patterns and simulate human intelligence had been introduced in the 1980s, by the early 2000s, LeCun, Hinton and Bengio were among a small group who remained committed to this approach. Though their efforts to rekindle the AI community’s interest in neural networks were initially met with skepticism, their ideas recently resulted in major technological advances, and their methodology is now the dominant paradigm in the field. (ACM, 2018) <https://awards.acm.org/about/2018-turing>

[Goodfellow et al. \(2016\)](#) lists four key trends in the development of DL (deep learning):

1. Deep learning has had a long and rich history but has gone by many names reflecting different philosophical viewpoints, and has waxed and waned in popularity;

2. Deep learning has become more useful as the amount of available training data has increased;
3. Deep learning models have grown in size over time as computer hardware and software infrastructure for deep learning has improved;
4. Deep learning has solved increasingly complicated applications with increasing accuracy over time.

Human cognition: the guiding concept of deep learning technology

One of the best perspectives from which to understand deep learning is to view it within the science of human cognition, which studies the intricate neuron pathways that facilitate thought. This approach requires a deeper dive into the structure and operation of the MLP. The MLP was elaborated to simulate the function of neuron pathways more closely. These neuron pathways are represented in deep learning algorithms as multiple middle (or “hidden”) layers in a neural network architecture. Data processing machines have been around for a long time, possibly as far back as Ancient Greece (the Antikythera mechanism – De Solla Price, 1974). One of the questions that arises inevitably is, can a machine think like a human? The quest to explore this question led to the development of AI technology, a branch of cognitive science. Initially, AI practitioners focused on solving problems that are hard for humans but easy for computers, such as building and using expert systems. The bigger challenge, however, was to develop techniques for solving problems that are easy for humans but hard for computers, like image and speech recognition (Schmidhuber, 2015).

The first step in building such AI systems was to mimic rather crudely the way the human brain analyzes sensory input data through the processing of sensory input in a neural network. Fig. 1.9 shows an artist’s rendition of a human neural network, composed of a number of neurons (the bloblike forms) connected together with slender pathways (dendrites).

Practitioners in cognitive science recognized that the human brain “thinks” through a series of linked neurons (nerve cells). The dendrites shown in Fig. 1.9 function to pass information in the form of electrical impulses between neurons. These electric impulses (or “signals”) can become either reduced in strength (attenuated), through biological counterparts to electrical resistors, or increased in strength by biological counterparts of electrical capacitors. This signal modulation along the path of neural impulse flow is analogous to the sequential processing of data in computers. This insight moved AI scientists to develop computer-expressed counterparts to these human neural networks called artificial neural networks, or ANNs.



FIG. 1.9 Artist’s conception of a network of neurons (a neural network) in the human brain. Source: https://www.google.com/search?q=deep+learning+images&biw=1920&bih=942&tbm=isch&imgil=SySt4Dda1-wHcM%253A%253BLGqLwznf8oZt8M%253Bhttp%25253A%25252F%25252Fhyperverge.co%25252Fa-beginners-introduction-to-deep-learning%25252F&source=iu&pf=m&fir=SySt4Dda1-wHcM%253A%25252CLGqLwznf8oZt8M%25252C_&usg=__Nb8nuBtLXzyo7vpMMioD4WEJS3c%3D&ved=0ahUKewiHwrmV9NPPAhUP-mMKHUCPBsUQyjdNQ&ei=Dnb9V4eDPY_0jwPANpqoDA#imgrc=bSOmXaxpV08xWM%3A

Early artificial neural networks with the MLP architecture

The first attempts in this simulation resulted in the development of ANNs (the perceptron) focused on distinguishing patterns in single datasets. The biggest difference, however, between these early AI methods and human cognitive abilities is the way humans use experience to adapt to and refine initial sensory patterns.

As discussed above, early ANNs had only two layers, a layer of data inputs (analogous to human sensory inputs) and a layer of outputs, analogous to the initiation of muscular action. The problem with the early ANNs was that this neural architecture and data processing approach did not function very well when the output response was significantly nonlinear with respect to the inputs. To solve this problem, AI algorithm designers added another layer between the input layer and the output layer. They named this structure the “hidden layer.” Separate weights between nodes in each layer were optimized. The relationships between the hidden layer and the input and output layers that are resident in these optimized weights functioned to express a large degree of nonlinearity between inputs and output. Fig. 1.10 is a repeat of Fig. 1.7, which shows the architecture of the MLP (Rosenblatt, 1962). This figure is shown here again because a proper understanding of deep learning neural networks (DLNNs) and Foundation models requires a deeper understanding of how Foundation models are related to deep learning networks, which are in turn related to classical neural networks. We must trace that relationship before LLMs can be discussed.

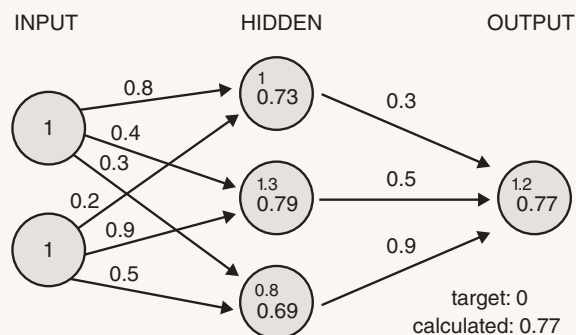


FIG. 1.10 The architecture of the multilayer perceptron (MLP), with node weights.

Fig. 1.7 shows the overall level architecture of an MLP. The following discussion digs deeper into the structure and function of an MLP to set the stage for presentation of more complicated forms. The large number 1 in the two left circles of Fig. 1.10 represents inputs; the numbers on the arrows represent weights; the small numbers in the center circles are the sums of the weight $\times$ input values directed to them; the large numbers in the center circles represent the resulting output of the nonlinear activation function controlling the “firing” of that node. The final value of 0.77 in the output node is the sum of the “hidden” (middle) layer node values times the weight associated with its link to the output node, processed through a nonlinear activation function.

Back-propagation: the learning engine

The concept of back-propagation was described formally by LeCun (1986) and by Rumelhart et al. (1986). The process directs the first record to flow through the neural network; the final value of the output node (the prediction) is then compared to the target value for that record, and an error is calculated. A portion of this error is used to update the weights used to predict the values of the second record to flow through the network. This process employs a form of a descent across an error gradient to minimize the prediction error (termed the gradient descent method). In this manner, the setting of the values for the weights between each node is “learned” by processing values for variables in a sequential set of records (a file). Other error minimization approaches, like Equilibrium Propagation (E-prop) are also used.

More elaborate architectures: deep learning neural networks

The neural networks in the human brain have many more than just three layers, in which many of the layers are built from sensory input patterns formed by many different experiences. This means that different variable data are input to different layers, each one the result of different

experiences. The encapsulation of this experience-based component of human neural networks was simulated by adding additional layers to the ANN architecture, fed by different sets of data inputs analogous to different experiences. The resulting ANN architecture was far “deeper” in terms of hidden layers and data inputs than the original MLP and was termed a deep learning neural network (DLNN). One of the early applications of DLNNs was to build the Deep Blue IBM chess-playing computer system, which beat the reigning world champion chess player, Gary Kasparov, in 1997 (Schmidhuber, 2015).

Major elaborations of DLNNs include:

1. Representational learning elements
2. Convolutions
3. Separate data sets input in some hidden nodes

Representational learning

An earlier development of learning machines involved mapping raw inputs to transformed features that “represented” raw data inputs. Analysis of these transformed inputs was called “representational learning” (RL). The problem with RL applied to early speech and image analysis was that it was difficult to extract nuances like speech accent or viewing angle of an image. DLNNs solve this problem in RL by using a group of simple representations to build more complex representations.

One application of a DLNN for image recognition dedicates the processing of one hidden layer to distinguish image elements contours, a second hidden layer to distinguish edges, and a third hidden layer to combine contours and edges to compose image parts (Schmidhuber, 2015).

Convolutional neural networks

One of the earliest applications of complex ANNs (those with more than three layers) was image processing. Studies in computational neuroscience of vision (initially of a cat) suggested that a subtle change in the mathematical operations of a machine learning algorithm could have profound benefits on both computational efficiency and noise reduction. The change was to use one function to modify another function to make it work better. The basic idea was to use one function to estimate (or generalize) the state of another function without having to solve the other function directly. Such a modification is called a *convolution*. For example, we might use an aggregating function to collect noisy inputs from a submarine sonar sensor and calculate the average over a small interval of time. Rather than submitting all instances of the noisy signal, only the average signal value per unit time is input. This approach can have very significant effects on lowering the processing time to generate the output and the storage requirements for intermediate products. This averaging is called *pooling*.

This example of sonar signal processing involves only one input, the signal; there are no predictor variables. Imagine how this approach could be leveraged to modify the processing of machine learning algorithms with many predictor variable inputs. That is how a convolutional neural network (CNN) was conceived.

Neural network algorithm developers recognized that the mathematical processing (i.e., matrix multiplication operations) during image analysis could be highly optimized in each layer of the neural network by using various convolutions to estimate parameter values. A convolution is accomplished by multiplying values in a smaller dimensional matrix by each value in the larger matrix (e.g., image of pixels). The recoded image facilitated the sensing of spatial relationships among groups of pixels on the image. These spatial relationships could be related to image classes (e.g., dogs or cats) for classification of the image. These CNNs were very efficient and effective in the analysis of any dataset that can be represented as a two-dimensional data structure (like pixels in an image). Some CNNs were developed to work with time-series datasets, which are essentially two-dimensional data structures (output dimension and the time dimension). Some specialized CNNs incorporated time delays in the input processing strategy, *time-dilated neural networks* (TDNNs).

In the early 1990s, AT&T developed CNNs for checking images in banks (Goodfellow et al. (2016)). Subsequently, Nippon Electric Corporation (NEC) adapted this technology to read over 10% of the checks processed by them in the United States. Now, all checks are read with variations of this technology.

Recurrent neural networks

Another kind of neural network that may be incorporated into deep learning systems is the recurrent neural network (RNN). RNNs are very different from CNNs in the ways they can analyze temporal data inputs, and generate sequential data output (Vorhies, 2016).

In contrast to the standard feed-forward ANNs, RNNs have bidirectional data flow. The standard feed-forward data flow occurs, followed sometimes by feed backward of data processed in later steps to affect the processing of earlier steps. This processing is referred to as back-propagation through time. This design does not make the assumption of other ANNs that data inputs are independent of each other in their effect on the output.

RNNs were modified subsequently to process various types of temporal data.

- Blocks of text of different lengths
- Audio speech signals
- Sequences of stock prices
- Streams of sensor data

Each of these types of temporal data can be viewed as sequential data through time. RNNs can learn from any data expressed as a sequence of data elements.

Memory

The feed-backward design in RNNs can loop not only just one data unit backward in time, but it can “look” backward many time units in the past. This feature permits the simulation of learning over a time scale specified by the user. Proper tuning of the algorithm can control how long it “remembers” and when to “forget.”

This ability to regulate when to remember and when to forget can help solve an underlying problem with RNNs. Each time step in the processing of an RNN is equivalent to training a feed-forward ANN with 100 hidden layers. This situation leads to very small gradients in the error surface over which the gradient descent error minimization process must operate. As the number of time steps increases, the extent of the error surface declines exponentially, and is known as the “vanishing gradient problem.” The most common technique is to use the long short-term memory (LSTM) approach. This feature is particularly useful with sensor data that has long delays, or when there is a mixture of high and low frequency data (Vorhies, 2016).

Vorhies (2016) lists a number of applications for which RNNs are useful:

- Speech recognition
- Text processing
- Handwriting recognition
- Image recognition

AI technology has been around for a long time. ANNs are slightly more recent (see Chapter 13). Very complex ANNs composing kinds of DLNN algorithms are about 10 years old. Use of the DLNNs has expanded exponentially since their introduction. A very generalized form of a complex of DLNNs is Foundation models. This new kind of model is introduced in this chapter, and additional information on LLMs (a type of Foundation model) is presented in Chapter 25.

Foundation models

Foundation models, also known as base models or pretrained models, are large neural network models that are trained on vast amounts of data to learn general patterns and relationships within the data. These models serve as the building blocks for various NLP tasks such as text generation, translation, summarization, sentiment analysis, question answering, and many other applications.

Foundation models are typically trained in a self-supervised manner, where they learn to predict missing parts of sentences or generate coherent sentences based on the surrounding context. This pretraining phase allows the model to learn useful representations of language and common-sense knowledge.

Once pretraining is complete, a Foundation model can be fine-tuned on specific downstream tasks by

training it on task-specific datasets. This process helps the model specialize in the particular task it is being trained for, such as sentiment classification or named entity recognition.

An interesting phenomenon associated with Foundation models is their *emergent properties* (Bommasani and Liang, 2021). An emergent property is one induced by the operation of a system of causes, which is not apparent at the level of the individual causes. These emergent properties can help to provide feedback to substantially improve performance, but they also represent a risk. Steinhardt (2021) fears that emergent properties of Foundation models will make deception by the model author easier to accomplish. Steinhardt proposes several fixes for this problem that include the development of anomaly detection systems for use with Foundation models.

Transformers and the concept of “attention”

A lot of “buzz” has developed about a new processing approach in advanced neural networks called the “Transformer.” This is a novel neural network architecture based on a self-attention mechanism that Vaswani et al. (2017) believed to be particularly well suited for language understanding. The rapid rise of LLMs since the “explosion” of interest in them followed the introduction of ChatGPT in December 2022. To help you understand what a transformer is and what it does in language processing, a little review of neural networks is necessary.

Consider one of the common response functions in data science, the logistic function, expressed by the plot shown in Fig. 1.11.

The logistic function expresses the dynamics of growth of entities in many aspects of nature (e.g., tree growth). We can elaborate the normal MLP neural network architecture of Fig. 1.10, to express the more

complicated architecture of a DLNN shown in Fig. 1.12.

Each connection of the final hidden layer node with an output node follows a response function, like the logistic growth function. Such a complicated neural network architecture may have dozens (or hundreds) of hidden layers necessary to express various nuances of highly nonlinear data patterns of many (maybe thousands) of data elements (variables) that compose a target data pattern. In a normal DLNN, each connection with an output node uses the same response function. Data flows through the neural network in one direction, from input nodes to output nodes. This data flow architecture is called a “feed-forward” design. Recurrent DLNNs can incorporate data from previous records (backward localized data flow), but the overall data flow is forward. In a CNN that processes images, pixels are passed through a lower-dimensional window in a process called a “convolution.” The convolution functions as a processing function for all pixels in an image, for example.

Can a neural network be configured to change the output function for some hidden nodes and not for others, and still be a feed-forward data flow process? The answer is yes, and that is what the self-attention (or simply attention) process does (Vaswani et al., 2017). The attention process permits the network to focus more on some elements (e.g., words in a text processing routine) than on other elements. The result is that the application can build a list of words according to their relevance to a query subject from a vast list of source documents harvested from the Internet. This is what an LLM (like ChatGPT) does.

Prominent examples of Foundation models

The most prominent examples of Foundation models include OpenAI’s GPT (generative pretrained transformer) models, such as GPT3 and GPT4, as well as other models like BERT (Bidirectional Encoder Representations

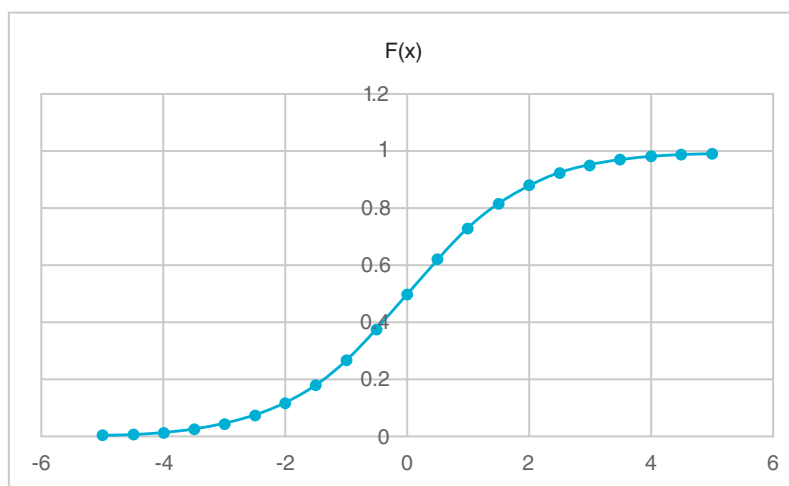


FIG. 1.11 The logistic function.

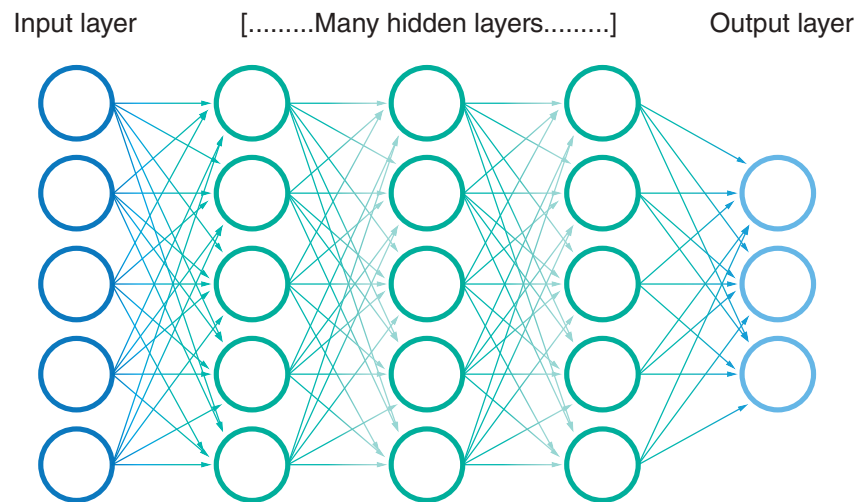


FIG. 1.12 Conceptual architecture of a deep learning neural network. Link weights are not shown in this figure for the sake of simplicity.

from Transformers) and RoBERTa (Robustly Optimized BERT). These models have significantly advanced the field of NLP and have been used as a basis for a wide range of applications and research in natural language understanding and generation. Note: Much of the previous text in this section was generated by GPT from OpenAI (source: <https://chat.openai.com/>). This type of foundation is called a *large language model (LLM)*, which can digest millions of online text (and sometimes image) documents to compose answers to queries.

How do large language models work?

An LLM is a type of neural network. This type of network architecture is called a *transformer*, which processes and generates text in sequence. In a simple neural network architecture, each neuron can be connected to every neuron in the layer above it. In other architectures, a neuron may be connected to some other neurons that are near it in a grid but not others further away. Such a partially connected neural network is called a convolutional neural network (CNN). CNNs preprocess input data with a mathematical expression called a *convolution*. This mathematical operation is analogous to “viewing” a grid of values (like an image composed of pixels) through a lower dimensional filter and changing the values to reflect how close to the filter values the image values are. This process can be used to classify groups of pixels in an image that have similar transformed values. This classification operation can be trained to “recognize” similar groups of pixels in an input image that are similar to those in the classified groups during the training operation. This capability of CNNs has enabled them to serve as very accurate “recognizers” of images (e.g., faces) or sound prints (e.g., voice patterns). A transformer, however, is somewhat different. A transformer uses the concept of “attention,” in which some neurons

are more strongly connected (or “pay more attention to”) to other neurons in a sequence (Vaswani et al., 2017). Text is essentially a sequence of word types (e.g., nouns, adjectives, and verbs). Thus the concept of attention in the processing algorithm architecture works well to compose text data from many sources to form sentences on a given topic in a paragraph.

Rather than using large volumes of existing online text data directly, an LLM is built from programmed rules that the LLM uses to build the model. The LLM essentially builds itself! This self-building property of an LLM can be adapted for many purposes, such as:

- Automating various processes
- Optimizing costs and minimizing processing times
- Increasing personalization
- Increasing accuracies of process fidelity
- Summarizing vast amounts of textual information

The capability of LLMs to summarize vast amounts of textual information has been leveraged to build a number of query-answering applications like ChatGPT by OpenAI and Microsoft, and Gemini by Google.

Gemini

Gemini is an LLM developed by Google, also known as a conversational AI or chatbot trained to be informative and comprehensive. Gemini has been trained on a massive amount of text data, and it is able to communicate and generate humanlike text in response to a wide range of prompts and questions. For example, it can provide summaries of factual topics or create stories.

Gemini is a powerful tool that can be used for a variety of purposes including:

- Education: Gemini can be used to help students learn new concepts and practice their skills. For example, Gemini can provide summaries of factual

topics, generate practice problems, and help students write essays.

- **Business:** Gemini can be used to help businesses improve their customer service, marketing, and product development. For example, Gemini can answer customer questions, generate marketing materials, and help businesses develop new products and services.
- **Personal:** Gemini can be used to help people with their personal tasks, such as writing letters, planning events, and creating presentations. For example, Gemini can help people write letters, plan events, and create presentations.

The previous paragraphs were composed from a slightly edited response to the question “Describe Gemini” submitted online to Gemini at <https://gemini.google.com/?hl=en>.

Recently, Gemini has been enhanced to return images in addition to text where appropriate. For example, Gemini was asked to: “Explain ROC curves and include an image in the results.” The return from the query is presented below as quoted text and a figure reproduced from Wikipedia. The ROC plot shown in Fig. 1.13 was created in KNIME to substitute for the plot provided by Gemini (e.g., Wikipedia) to avoid any copyright issues.

“ROC curves are a graphical plot that shows the performance of a binary classifier system as its discrimination threshold is varied. The curve shown in Fig. 1.13 is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The TPR is the fraction of actual positives that are correctly identified, while the FPR is the fraction of actual negatives that are incorrectly identified.”

“An ROC curve can be used to compare the performance of different binary classifiers, or to evaluate the performance of a single classifier over a range of thresholds. A perfect classifier would have an ROC curve that passes through the upper-left corner of the plot, with a true positive rate of 1 and a false positive rate of 0. In practice, however, no classifier is perfect, and the ROC curve will always fall below this ideal curve.

The area under the ROC curve (AUC) is a single number that summarizes the performance of a binary classifier. A higher AUC indicates a better classifier. An AUC of 0.5 indicates a classifier that is no better than random guessing, while an AUC of 1 indicates a perfect classifier.

ROC curves are a useful tool for evaluating the performance of binary classifiers. They can be used to compare different classifiers, to evaluate the performance of a single classifier over a range of thresholds, and to select a classifier that is well-suited for a particular application.”

Can an LLM replace low-level programmers?

One of the fears related to LLMs bandied about in the press is the fear by low-level programmers that this AI technology will put them out of a job. Is this likely? Editors of *The Economist* magazine don’t think so (*The Economist*, 2023). Rather, they expect that programming activity will move upward in complexity, leaving the “grunt” work to LLMs. A good case in point is the result of a query submitted to Gemini to write a Python script to train a Random Forest Model with cross validation. Gemini not only wrote a valid Python script to perform the task, but it also specified the Internet source of the Iris dataset to run it. Python programmers build a

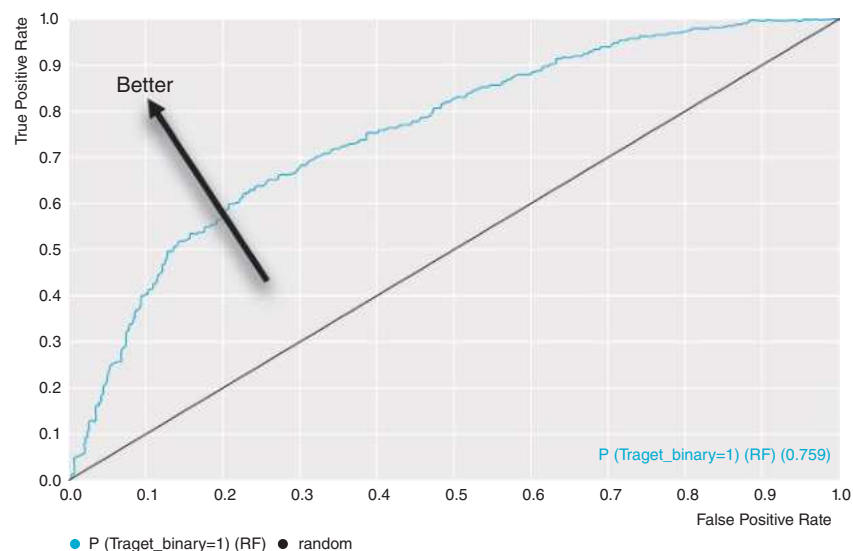


FIG. 1.13 The ROC curve generated by a Gradient Boosted Trees algorithm. The black arrow indicates the direction of movement of the curved blue line to represent a better model. The area under the curve above the diagonal black line (0.759) is shown at the bottom right of the plot.

library of Python scripts and spend their time putting them together and optimizing the algorithms to produce the final product.

Rapidly expanding LLM developments

The introduction of ChatGPT3 in November 2022 ignited a firestorm of new AI startups. In January 2023, Microsoft invested \$10 billion in OpenAI, developer of ChatGPT (<https://www.bloomberg.com/news/articles/2023-01-23/microsoft-makes-multibillion-dollar-investment-in-openai?leadSource=uverify%20wall>). Much more investment in LLMs will follow soon. Microsoft plans to incorporate ChatGPT into the Azure Cloud. OpenAI revealed ChatGPT4-beta in March 2023 and released the full version to business in August 2023. Many more applications that embed LLMs in their operations will become available soon.

Current trends of development in predictive analytics

1. Automation of much of data preparation and modeling processes
2. Development of a rich choice of open-source tools
3. Specialized analytics processing for the following:
 - a. Social network analysis
 - b. Sentiment analysis
 - c. Genomic sequence analysis
4. Predictive analytics will become the central dogma of data processing in every organization. Information technology (IT) departments will transform from a focus on storage and retrieval to a focus on providing a platform for operationalizing predictive analytics across the enterprise. This change in focus will drive development of the following:
 - a. Analytics-as-a-service (AAAS) provided to clients inside and outside the organization
 - b. Distributed analytics
 - c. Development of more deep-learning technologies
 - d. "Cloud" computing – using the Internet to distribute data and analytic computing tasks to many computers anywhere in the world but without a centralized hardware infrastructure of grid computing. This is happening already in the services of Microsoft Azure and Google Analytics.
 - e. Smarter, more responsible, and scalable AI applications (*Gartner Insights*, 2023)
 - f. Incorporation of LLMs as "assistants" in many applications
5. The top trends for 2023 (*Marr*, 2023):
 - a. Data democratization to put data to work for the entire workforce, not just for data scientists

- b. The explosion of AI applications in most businesses and sectors of economy
- c. Data-as-a-service (DAAS) enabled by the Cloud
- d. Access to real-time data for analytical applications
- e. Development of data governance and regulation

Postscript

What lies beyond this for predictive analytics? As we accumulate more and more data, we will probably discover increasingly clever ways to simulate more closely the operation of the most complex learning machine in the universe – the human brain. Whether we will attain the holy grail of AI research – to match and thereby inevitably exceed human performance – is a hotly debated topic. Some welcome that day, envisioning phenomenal benefits, and others see doom for humanity! Clearly, the power of analytic needs, even today, to be guided by our best natures to continue to benefit. The new deep-learning techniques in predictive analytics may be viewed as harbingers of a fifth generation in the development analytic techniques. The big new element in analytics is the development of LLMs, which can be incorporated into application operations on the cloud (e.g., Microsoft Azure), which in turn can supercharge the performance of analytics operations. Some technical soothsayers claim that a new phase of the Internet has begun.

Keywords and phrases

Probability, Algorithms, Perceptron, Bayesian, Logistic, Nonlinear, Parametric, Classifier, Convolutional, Automated, Interactions, Predictive analytics, Neural networks, Foundation models, Machine learning, Deep learning, Data flows, emergent properties, large language model.

References

- ACM, 2018. Fathers of the deep learning revolution receive ACM A.M. Turing Award. <https://awards.acm.org/about/2018-turing>
- Agresti, A., 1996. An introduction to categorical data analysis. John Wiley & Sons. 290 p.
- Aubrey, J. M. (2021). The misuse of Bayesian inference in machine learning. In: Migdal, P. D., Volz, M. (Eds.), *Philosophy of Machine Learning* (pp. 137-153). Springer, Berlin.
- Bommasani, R., Liang, P., 2021. Reflections on foundation models. *Stanford News*. <https://hai.stanford.edu/news/reflections-foundation-models>
- Botkin, D., 1993. Forest dynamics: An ecological model. Oxford University Press. 309 p.
- Box, J.F.R., 1978. A. Fisher: The life of a scientist. Wiley.
- Cristianini, N., Shawe-Taylor, J., 2000. An introduction to support vector machines. Cambridge University Press.

- Daobin, P. (1999). Proceedings of the 52nd Conference of the International Statistical Institute, Finland. Based on <https://www.stat.fi/isi99/proceedings/arkisto/varasto/peng0640.pdf>.
- De Solla Price, D. J. (1974). Gears from the Greeks. The Antikythera Mechanism: A Calendar Computer from ca. 80 B.C. Transactions of American Philosophical Society, New Series, 64, 19. <https://doi.org/10.2307/1006146>.
- Elder, J., Abbott, D., 1998. A comparison of leading data mining tools. Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining. <https://www.aaai.org/Papers/KDD/1998/Tutorials/Abbott.pdf>
- Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (Eds.), 1996. Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park, 611 p.
- Fisher, R.A., 1922. On the mathematical foundation of theoretical statistics. Philos. Trans. Royal Soc, London, A 222, 594–604.
- Gartner Insights. 2023. Top ten data analytics trends for 2021. <https://www.gartner.com/smarterwithgartner/gartner-top-10-data-and-analytics-trends-for-2021>.
- Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*, (pp. 800). Cambridge, MA: MIT Press.
- Han, J., Kamber, M., 2006. Data mining: Concepts and techniques. Morgan Kaufmann, 2nd ed. San Francisco, 770 p.
- Helmholtz, H. von 1878. Die Thatsachen in der Wahrnehmung, In Vorträge und Reden. Vieweg, pp. 213–247.
- Kelleher, J., Naamee, B., D’Arcy, A., 2020. Fundamentals of machine learning for predictive data analytics, 2nd ed. MIT Press, 856 p.
- Larose, D.T., Larose, C.D., 2015. Data mining and predictive analytics. John Wiley & Sons, 794 pp.
- LeCun, Y. (1986). Learning processes in an asymmetric threshold network. In: Bienenstock, E., Fogelman-Soulie, F., Weisbuch, G. (Eds.), Disordered Systems and Biological Organization (pp. 237–244). Springer-Verlag.
- LeCun, Y., Bengio, Y., Hinton, G., 2015. “Deep Learning”. Nature 521 (7553), 436–444. <https://doi.org/10.1038/nature14539> PMID 26017442. S2CID 3074096.
- Lee, P.M., 1989. Bayesian sensitivity: An introduction. Oxford University Press.
- Marr, B. (2023). Top 5 trends in data science and analytics in 2023. Forbes Magazine, online article Oct. 31, 2023. <https://www.forbes.com/sites/bernardmarr/2022/10/31/the-top-5-data-science-and-analytics-trends-in-2023/?sh=3cf11a6f5c41>
- McGrayne, S.B., 2011. The Theory That Would Not Die: How Bayes’ Rule Cracked the Enigma Code, Hunted Down Russian Submarines, and Emerged Triumphant from Two Centuries of Controversy. Yale University Press, New Haven, CT, 360 p.
- Minsky, M., Papert, S., 1969. Perceptrons: An introduction to computational geometry. MIT Press. (3rd Edition published in 1988).
- Newton, R.R., Rudestan, K.E., 1999. Your statistical consultant. Sage Publishing, 390 p.
- Rosenblatt, F., 1958. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65 (6), 386–408. <https://doi.org/10.1037/h0042519>
- Rosenblatt, F., 1962. Principles of neurodynamics: Perceptrons and the theory of brain mechanisms. Spartan Books. <http://catalog.hathitrust.org/Record/000203591>
- Rumelhart, D., Hinton, G., Williams, R., 1986. Learning representations by back-propagating errors. Nature 323, 533–536.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks 61:85–117.
- Smith, J. 2016. Data analytics vs. data science – Two separate but interconnected disciplines. <https://datascientistsinsights.com/2013/09/09/data-analytics-vs-data-science-two-separate-but-interconnected-disciplines/>
- Spearman, C., 1904. The proof and measurement of association between two things”. American Journal of Psychology 15 (1), 72–101. <https://doi.org/10.2307/1412159> JSTOR 1412159
- Steele, R., Torrie, J., 1960. Principles and procedures of statistics. McGraw-Hill Book Company.
- Steinhardt, J., 2021. On the risks of emergent behavior in foundation models. Stanford Center for Research on Foundation Models. <https://crfm.stanford.edu/commentary/2021/10/18/steinhardt.html>
- The Economist – editorial. (2023). Your job is (probably) safe from artificial intelligence. The Economist Group Limited.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, I. Polosukhin. 2017. Attention is all you need. 31st Conference on Neural Information Processing Systems (NIPS 2017). <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>
- Von Mises, R., 1957. Probability, statistics, and truth. Dover Publications, 244 p.
- Vorhies, W. (2016). Has deep learning made traditional machine learning irrelevant? DataScienceCentral online newsletter. <https://www.datasciencecentral.com/has-deep-learning-made-traditional-machine-learning-irrelevant/>
- Westheimer, G., 2008. Was Helmholtz a Bayesian? Perception 37, 1–10.

Theoretical considerations for data analytics

Preamble

In [Chapter 1](#), we explored the historical background of statistical analysis and data analytics. Statistical analysis is a relatively old discipline (particularly if you consider its origins in China). But data analytics is a relatively new field, which developed under the name “data analytics” during the 1990s and coalesced into an academic field of its own as Data Science during the early years of the twenty-first century. It represents a confluence of several well-established fields of interest:

- Traditional statistical analysis
- Artificial intelligence
- Machine learning
- Development of large databases

Traditional statistical analysis follows the *deductive method* in the search for relationships in data sets. Artificial intelligence (AI; e.g., expert systems) and machine learning (ML) techniques (e.g., neural nets and decision trees) follow the *inductive method* to find faint patterns of relationship in data sets. Deduction (or deductive reasoning) is the Aristotelian process of analyzing detailed data, calculating a number of metrics, and forming some conclusions based (or deduced) solely on the mathematics of those metrics. Induction is the more Platonic process of using information in a data set as a “springboard” to make general conclusions, which are not wholly contained directly in the input data. The scientific method follows the inductive approach but has strong Aristotelian elements in the preliminary steps.

The scientific method

The scientific method is as follows:

1. Define the problem.
2. Gather existing information about a phenomenon.
3. Form one or more hypotheses.

4. Collect new experimental data.
5. Analyze the information in the new data set.
6. Interpret results.
7. Synthesize (infer) conclusions based on input data, experimental results, and *intuition*.
8. Form new hypotheses for further testing.
9. Do it again (iterate) as other scientists seek to replicate results of the first iteration.

Steps 1–5 involve deduction, and steps 6–9 involve induction. Even though the scientific method is based strongly on deductive reasoning, the final products arise through inductive reasoning. Data analytics is a lot like that. In fact, ML algorithms used in data analytics are designed to mimic the process that occurs in the mind of the scientist. Data analytics uses mathematics, but the results are *not* mathematically determined. This statement may sound somewhat contradictory until you view it in terms of the human brain. You can describe many of the processes in the human conceptual pathway with mathematical relationships, but the *result* of being human goes far beyond the mathematical descriptions of these processes. Intuition, mother’s wisdom regarding their offspring, and “gut-” level feelings about who should win the next election are all *intuitive models* of reality created by the human brain. They are based largely on empirical data, but the mind extrapolates beyond the data to form the conclusions following a purely inductive reasoning process.

Some important differences between the scientific method and traditional machine learning

As we’ve just seen, an important difference between the scientific method and ML methods is that ML methods begin with the inductive step. Let’s look at this a bit more closely. When using the scientific method, our initial hypothesis is generated most often from our own prior research or the published prior research of others.

Based on our hypothesis, we design an experiment and collect data that will be used to test our hypothesis.

This sequence of events does not describe the beginning of an ML project very well. While both analysis approaches can establish truths about our data and allow us to make predictions about future data, ML methods can be viewed as starting with machine-assisted induction and postponing the testing of hypotheses until after a model has been identified. Also, ML methods analyze data that is gathered through the normal course of doing business, so it preexists the hypothesis, and the collection of it was not designed to test the hypotheses at all.

The reason that ML methods cannot begin with a hypothesis is that one is never available. If the problem space were well understood, the data analytics process would not be needed – ML methods search for as-yet unknown connections. ML techniques are used *precisely because* an initial hypothesis is not available.

It is a common mistake to see too strong an analogy between the scientific method and ML methods and to perceive an equivalence between Business Understanding and the formation of a hypothesis. While the steps are indeed similar, the order is (and must) be different. As we've determined, ML methods begin with induction. The next concern is what form the knowledge discovery process takes from there in Business Understanding.

The Cross-Industry-Standard-Process for Data Mining (CRISP-DM) process model has become a popular approach to describe the nature and sequence of common analytical modeling operations. During the Business Understanding phase of CRISP-DM "focuses on understanding the project objectives and requirements from a business perspective, then converting this knowledge into a data analytics problem definition and a preliminary plan designed to achieve the objectives." (CRISP-DM document).

As one of the CRISP-DM project team members puts it:

"Business Understanding is understanding the problem space (and eventual success criteria at the business level); it's not about formulating hypotheses." Colin Sherer, private communication.

Or as Tom Khabaza, another CRISP-DM coauthor, puts it in his Nine Laws:

1st Law of Data Analytics – "Business Goals Law:"

Business objectives are the origin of every data analytics solution

This defines the field of data analytics: data analytics is concerned with solving business problems and achieving business goals. data analytics is not primarily a technology; it is a process, which has one or more business objectives at its heart. Without a business objective (whether or not this is articulated), there is no data analytics. Khabaza's 1st Law. http://khabaza.codimension.net/index_files/9laws.htm

If the first step in ML analysis is the statement of a business problem, what step in ML analysis mining is analogous to the statement of a hypothesis? The answer

to this question is based on the notion that the modeling algorithm is a form of machine-assisted induction, out of which the hypothesis emerges. The resulting model can be viewed as a kind of hypothesis. Therefore the combination of the initial process of business evaluation and the process of model performance assessment is how the hypothesis is scientifically tested. Therefore turning the sequence on its head, ML analysis performs deduction *after* induction, not *before* it.

What is the process of data analytics?

In the early days of ML algorithms, the process of using them to predict outcomes was called "data mining." The allusion was made to the process of mining gold (actionable information) from a mountain (of data). The early term data mining can be defined in several ways, which differ primarily in their focus. One of the earliest definitions is

The non-trivial extraction of implicit, previously unknown, and potentially useful information from data (Frawley et al., 1991).

As data analytics developed as a professional activity, it was necessary to distinguish it from the previous activity of statistical modeling and the broader activity of knowledge discovery. For the purposes of this handbook, we will use the following working definitions:

- **Statistical modeling:** The use of parametric statistical algorithms to group or predict an outcome or event, based on predictor variables
- **Data analytics:** The use of ML algorithms to find faint patterns of relationship between data elements in large, noisy, and messy data sets, which can lead to actions to increase benefit in some form (diagnosis, profit, detection, etc.)
- **Knowledge discovery:** The entire process of data access, data exploration, data preparation, modeling, model deployment, and model monitoring. This broad process includes data analytics activities, as shown in Fig. 2.1.

Fig. 2.1 shows the relationships among data analytics, Knowledge Discovery in Databases (KDD), and Data Science.

As the practice of data analytics developed further, the focus of the definitions shifted to specific aspects of the information and its sources. In 1996, Fayyad et al. proposed a new term, Knowledge Discovery, defined as: "... discovery in databases is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data."

This definition focuses on the *patterns* in the data that are discovered, rather than just gaining information in a

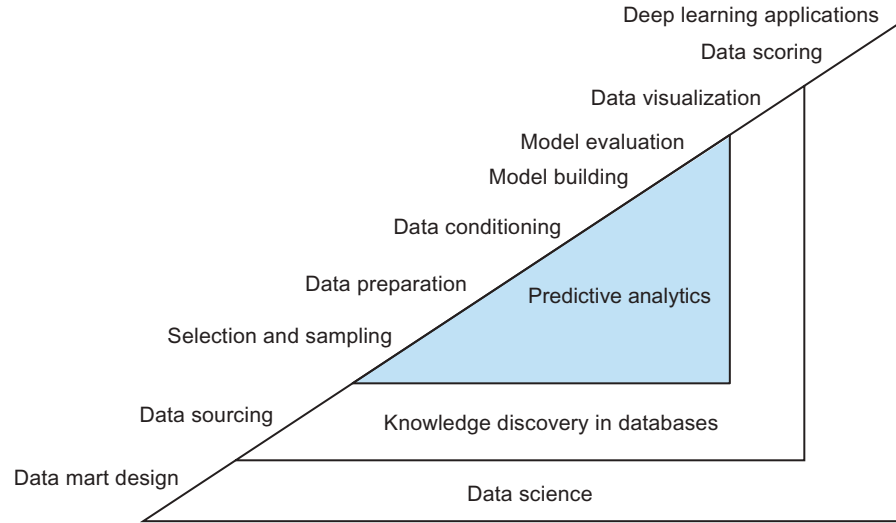


FIG. 2.1 Relationship among data science, knowledge discovery, and data analytics.

generic sense. These patterns are often faint and hard to distinguish, and they can only be *sensed* by analysis algorithms that can evaluate highly nonlinear relationships between predictor variables and their targets. This definition of data analytics developed along with the rise of ML tools. Tools like decision trees and neural nets permit the analysis of nonlinear patterns in data easier than is possible in parametric statistical algorithms. The reason is that ML algorithms learn more the way humans do – by example, rather than by calculation of metrics based on averages and data distributions.

The definition of data analytics was confined originally to just the process of model building. However, as the practice matured, data analytics tool packages included other necessary tools to facilitate the preparation of data and for evaluating and displaying models. Soon, the definition of data analytics expanded to include those operations in Fig. 2.1 (and some include model visualization also). As the number and kinds of applications of data analytics technology developed, sometimes the outcome was not a prediction per se but the maintenance of a state of operation (e.g., power system maintenance and iris scanners in security systems). By 2009, the predictive aspect of the term data mining had been largely replaced by “Predictive Analytics.” The collective discipline of “data analytics” was coined in about 2010 to include the aspects of predictive analytics and the nonpredictive aspects of analysis (e.g., data visualization and preparation operations). In this book, we will focus primarily on the practice of Predictive Analytics, under the broader umbrella of data analytics. Both terms will be used interchangeably, even though data analytics contains some nonpredictive applications.

The term Data Science also includes data analytics, which analyzes patterns of relationship among entities

other than databases (e.g., images and sound graphs). The blue-colored section of the figure shown in Fig. 2.1 is the predictive analytics process, which combines the mathematics used to discover interesting patterns in data with the entire process of extracting data and using resulting models to apply to other data sets to leverage the information for some purpose. This process blends business systems engineering, elegant statistical methods, and industrial-strength computing power to find *structure* (connections, patterns, and associations) rather than statistical parameters (means, weights, thresholds, knots). In Chapter 3, we will expand this rather linear organization of predictive analytics processes to describe the iterative, closed-loop system with feedback that comprises the modern approach to the practice of predictive analytics.

A theoretical framework for the data analytics process

The evolutionary nature of the definition and focus of predictive analytics occurred primarily as a matter of experience and necessity. A major problem with this development was the lack of a consistent body of theory, which could encompass all aspects of the nature of information, where it comes from, and how it is used. This logical concept is sometimes called a *model-theoretic*. Model theory links logic with algebraic expressions of structure to describe a system or complex process with a body of terms with a consistent syntax and the relationships between them (semantics). Most expressions of predictive analytics activities include inconsistent terms (e.g., attribute and predictor), which may imply different logical semantic relations with the data elements employed.

Mannila (2000) summarized a number of criteria that should be satisfied in an approach to develop a model-theoretic for predictive analytics. These criteria include the ability to

- Model typical predictive analytics tasks (clustering, rule discovery, classification)
- Describe data and the inductive generalizations derived from the data
- Express information from a variety of forms of data (relational data, sequences, text, Web)
- Support interactive and iterative processes
- Express comprehensible relationships
- Incorporate users in the process
- Incorporate multiple criteria for defining what is an “interesting” discovery

Mannila describes a number of approaches to developing an acceptable model-theoretic but concludes that none of them satisfy all the above criteria. The closest we can come is to combine the microeconomic approach with the inductive database approach.

Microeconomic approach

The starting point of the microeconomic approach is that predictive analytics is concerned with finding actionable *patterns* in data that have some *utility* to form a decision aimed at getting something done (e.g., employing interdiction strategies to reduce attrition). The goal is to find the decision that maximizes the total utility across all customers.

Inductive database approach

An inductive database includes all the data available in a given structure *plus* all the questions (queries) that could be asked about patterns in the data. Both stored and derived facts are handled in the same way. One of the most important functions of the human brain is to serve as a pattern recognition engine. Detailed data are submerged in the unconscious memory, and actions are driven primarily by the stored patterns.

Mannila suggests that the microeconomic approach can express most of the requirements for a model-theoretic based on stored facts, but the inductive database approach is much more facile to express derived facts. One attempt to implement this was made in the development of the Predictive Modeling Markup Language (PMML) as a superset of the standard Extended Markup Language (XML). Most data analytics packages available today store internal information (e.g., arrays) in XML format and can output results (analytical models) in the form of PMML. This combination of XML and PMML permits expression of the same data elements and the predictive

analytics process in either a physical database environment or a Web environment. When you choose your analytical mining tool, look for these capabilities.

Strengths of the data analytics process

Traditional statistical studies use past information to *determine* a future state of a system (often called *prediction*), whereas predictive analytics studies use past information to construct patterns based not solely on the input data but also on the *logical consequences* of those data. This process is also called *prediction*, but it contains a vital element missing in statistical analysis: the ability to provide an orderly expression of *what might be* in the future, compared to *what was* in the past (based on the assumptions of the statistical method).

Compared to traditional statistical studies, which are often hindsight, the field of predictive analytics finds patterns and classifications that look toward and even predict the future. In summary, predictive analytics can (1) provide a more complete understanding of data by finding patterns previously not seen and (2) make models that predict, thus enabling people to make better decisions, take action, and therefore mold future events.

Customer centric versus account centric: a new way to look at your data

Most computer databases in business were designed for the efficient storage and retrieval of account or product information. Business operations were controlled by accounting systems; it was natural that the application of computers to business followed the same data structures. The focus of these data structures was on transactions, and multiple transactions were stored for a given account. Data in early transactional business systems were held in Indexed Sequential Access Method (ISAM) databases. But as data volumes increased and the need for flexibility increased, Relational Database Management Systems (RDBMS) were developed. Relational theory developed by C.J. Codd distributed data into tables linked by primary and foreign keys, which progressively reduced data redundancy (like customer names) in (eventually) six “normal forms” of data organization. Some of the very large relational systems using NCR Teradata technology extend into the hundreds of terabytes. These systems provide relatively efficient systems for storage and retrieval of account-centric information.

Account-centric systems were quite efficient for their intended purpose, but they have a major drawback: it is difficult to manage *customers* per se as the primary

responders, rather than accounts. One person could have one account or multiple accounts. One account could be owned by more than one person. As a result, it was very difficult for a company on an RDBMS to relate its business to specific customers. Also, accounts (per se) don't buy products or services, and products don't buy themselves. *People* buy products and services, and our business operations (and the databases that serve them) should be oriented around the customer, not around accounts.

When we store data in a customer-centric format, extracts to build the Customer Analytical Record (CAR) are much easier to create (see below for more details on the CAR). And customer-centric databases are much easier to update *in relation to the customer*.

The physical data mart

One solution to this problem is to organize data structures to hold specific aspects (dimensions) of customer information. These structures can be represented by tables with common keys to link them together. This approach was championed by Oracle to hold customer-related information apart from the transactional data associated with them. The basic architecture was organized around a central (fact) table, which stored general information about a customer. This fact table formed the hub of a structure like a wheel (Fig. 2.2). This structure became known as the *star-schema*.

Another name for a star-schema is a multidimensional database. In an online store, the dimensions can hold data elements for Products, Orders, Back-orders, etc. The transactional data are often stored in another very different data structure. The customer database system is refreshed daily with summaries and aggregations from the transactional system. This smaller database is "dependent" on the larger database to create the summary and aggregated data stored in it. When the larger database is a data warehouse, the smaller dependent database is referred to as a dependent data mart. In Chapter 3, we will see how a system of dependent data marts can be organized around a

relational data warehouse to form the Corporate Information Factory.

The virtual data mart

As computing power and disk storage capacity increased, it became obvious in the early 1990s that a business could appeal to customers directly by using characteristics and historical account information, and Customer Relationship Management (CRM) was born. One-to-one marketing appeals could be supported, and businesses became "smarter" in their ability to convince customers to buy more goods and services. The success of CRM operations changed the way some companies looked at their data. No longer must companies view their databases in terms of just accounts and products, but rather, they could view their customers directly, in terms of all their associated accounts, products, and demographic data. These "logical" data marts could even be implemented as "views" in a relational database system.

Household databases

Another way to gain customer-related insights is to associate all accounts to the customers who own them and to associate all individual members of the same household. This process is called *householding*, and it requires some fuzzy matching to aggregate all accounts to the same customer because names may not be spelled exactly the same way in all records and full middle names might be used in one case, while middle initials are used in another case. An analogous situation occurs when trying to gather all individuals into the same household, because not all addresses are listed in exactly the same format. This process of fuzzy matching can be performed by many data integration and data quality tools available in the market today (DataFlux, Trillium, Informatica Data Quality, IBM Quality Stage).

For example, the househoused data structure in a bank could consist of the following tables:

- Accounts
- Individuals
- Households

Historical data could be combined with each of the preceding hierarchical levels of aggregation. Alternatively, the preceding tables could be restricted to current data, and historical data could be installed in historical versions of the same tables (e.g., Accounts_Hst), linked together with common keys. This compound structure would optimize speed of database queries and simplify data extraction for most applications requiring only current data. Also, the historical data would be available for trending in the historical tables.

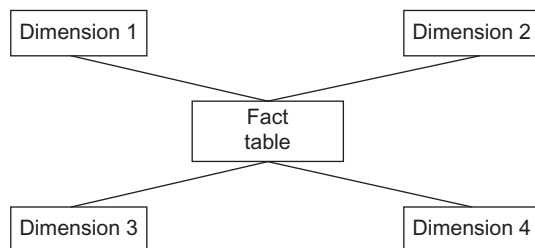


FIG. 2.2 A simple star-schema database structure.

The data paradigm shift

The organization of data structures suitable for predictive analytics requires a basic shift in thinking about data in business. Data do not serve the account; data should be organized to serve the customer who buys goods and services. To directly serve customers, data must be organized in a customer-centric data structure to permit the following:

- Relationship of all data elements must be relevant to the customer.
- Data structures must make it relatively easy to convert all required data elements into a form suitable for predictive analytics: the CAR.

Creation of the CAR

All input data must be loaded into the CAR (Accenture Global Services, 2006). This process is similar to preparing for a vacation by automobile. If your camping equipment is stored in one place in your basement, you can easily access it and load it into the automobile. If it is spread throughout your house and mixed in with non-camping equipment, access will be more difficult because you have to separate (extract) it from other items. Gathering data for predictive analytics is a lot like that. If your source data is a data warehouse, this process will *denormalize* your data. Denormalization is the process of extracting data from normalized tables in the relational model of a data warehouse. Data from these tables must be associated with the proper individuals (or households) along the way. Data integration tools (like SAS DataFlux or Informatica) are required to extract and transform data from the relational database tables to build the CAR. See any one of a number of good books on relational data warehousing to understand what this process entails. If your data are already in a dimensional or householding data structure, you are already halfway there. The CAR includes the following:

1. All data elements are organized into one record per customer.
2. One or more “target” (Y) variables are assigned or derived.

The CAR is expressed as a textual version of the following equation:

$$Y = X_1 + X_2 + X_3 + \dots X_n$$

This expression represents a computerized “memory” of the information about a customer. These data constructs are analyzed by either statistical or ML “algorithms,” following specific methodological operations. Algorithms are mathematical expressions that describe

relationships between the variable predicted (Y or the customer response) and the predictor variables ($X_1 + X_2 + X_3 + \dots X_n$). Predictive analytics algorithms are discussed in Chapters 13 and 17.

The CAR is analyzed by parametric statistical or ML algorithms within the broader process of Knowledge Discovery in Databases (KDD), as shown in Fig. 2.1. The data analytics aspect of KDD consists of an ordered series of activities aimed at training and evaluating the best patterns (for ML) or equations (for parametric statistical procedures). These optimum patterns or equations are called *models*.

Major activities of predictive analytics

Major predictive analytics activities include the following general operations (Hand et al., 2001):

1. **Exploratory data analysis:** These data exploration activities include interactive and visual techniques that allow you to “view” a data set in terms of summary statistical parameters and graphical display to “get a feel” for any patterns or trends that are in the data set.
2. **Descriptive modeling:** This activity forms higher-level “views” of a data set, which can include the following:
 - a. Determination of overall probability distributions of the data set (sometimes called *density estimation*)
 - b. Models describing the relationship between variables (sometimes called *dependency modeling*)
 - c. Dividing data into groups by *cluster analysis* or *segmentation*. Cluster analysis is a little different, as the clustering algorithms try to find “natural groups” either with many “clusters” or in one type of cluster analysis. The user can specify that all cases must be put into a number of clusters (e.g., 3). For segmentation, the goal is to find relatively homogeneous groups of entities related in generally the same way to the variable to be modeled (the *dependent* or *target* variable).
3. **Predictive modeling: classification and regression:** The goal here is to build a model where the value of one variable can be predicted from the values of other variables. Classification is used for “categorical” variables (e.g., Yes/No variables or multiple-choice answers for a variable like 1–5 for “like best” to “like least”). Regression is used for “continuous” variables (e.g., variables where the values can be any number, with decimals, between one number and another; age of a person would be an example, or blood pressure, or number of cases of a product coming off an assembly line each day).
4. **Discovering patterns and rules:** This activity can involve anything from finding the combinations of

items that frequently occur together in transaction databases (e.g., products that are usually purchased together, at the same time, by a customer at a convenience store, etc.) or things like finding groupings of stars, maybe new stars, in astronomy, to finding genetic patterns in DNA microarray assays. Analyses like these can be used to generate association rules; for example, if a person goes to the store to buy milk, they will also buy orange juice. Development of association rules is supported by algorithms in many commercial analytical software products. An advanced association method is Sequence, Association, and Link (SAL) analysis. SAL analysis develops not only the associations but also the sequences of the associated items. From these sequenced associations, “links” can be calculated, resulting in Web link graphs or rule graphs (see the NTSB Text Mining Tutorial, included with this book, for nice illustrations of both rule graphs and SAL graphs).

5. **Retrieval by content:** This activity type begins with a known pattern of interest and follows the goal to find similar patterns in the new data set. This approach to pattern recognition is most often used with text material (e.g., written documents, brought into analysis as Word docs, PDFs, or even text content of Web pages) or image data sets.

To those unfamiliar with these predictive analytics activities, their operations might appear magical or invoke images of the wizard. But contrary to the image of data miners as magicians, their activities are very

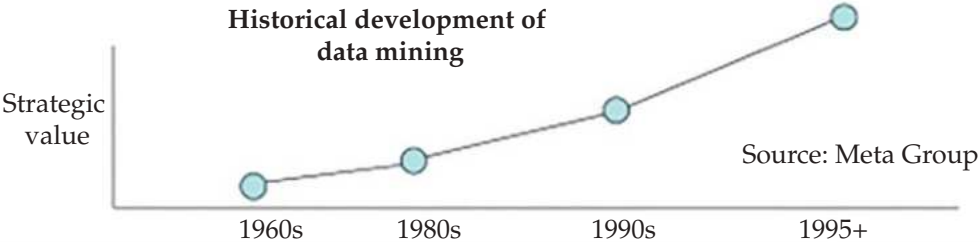
simple in principle. They perform their activities following a very crude analog to the way the human brain learns. ML algorithms learn case by case, just the way we do.

Data input to our senses are stored in our brains not in the form of individual inputs but in the form of *patterns*. These patterns are composed of a set of neural signal strengths our brains have associated with known inputs in the past. In addition to their abilities to build and store patterns, our brains are very sophisticated pattern recognition engines. We may spend a lifetime building a conceptual pattern of “the good life,” event by event and pleasure by pleasure. When we compare our lives with those in other countries, we unconsciously compare what we know about their lives (data inputs) with the patterns of our good lives. Analogously, an ML algorithm builds the pattern that is “sensed” in a data set. The pattern is saved in terms of mathematical weights, constants, or groupings. The mined pattern can be used to compare mathematical patterns in other data sets and to score their quality. Granted, data analysts have to perform many detailed numerical operations required by the limitations of our tools and available data. But the principles behind these operations are very similar to the ways our brains work.

Data analytics and predictive analytics did not arise as a new academic discipline from the studies in universities. Rather, they represented the logical next step in a series of developments in business to use data and computers to do business better in the future. Table 2.1 shows the historical roots of data analytics.

The early history of the development of data analytics technology is shown in Table 2.1 and its associated figure.

TABLE 2.1 Historical development of data analytics



Developmental Step	Data collection	Data Access	Data Warehousing and Decision Support	Data Mining
Business question	"What was my total revenue last year?"	"What were the sales in Ohio last March?"	"What were the sales in Ohio last March?"-Drill down to Dayton	"What will be the sales in Dayton next month?"
Enabling technology	Computers tapes and disks	Relational databases and SQL	Data warehouses multidimensional databases	Advanced algorithms multiprocessor massive databases
Characteristics	Delivery of static past data summaries	Delivery of dynamic past data at record level	Delivery of dynamic past data at multiple level	Prospective proactive information delivery

Major challenges of data analytics and predictive analytics

Some of the major challenges of data analytics projects include

- Use of data in transactional databases for data analytics
- Data reduction
- Data transformation
- Data cleaning
- Data sparsity
- Data rarity (rare case pattern recognition and thus “data set balancing”)

Each of these challenges will be discussed in the ensuing chapters.

General examples of predictive analytics applications

Predictive analytics technology can be applied anywhere a decision is made, based on some body of evidence. The diversity of applications in the past included the following:

- **Sales forecasting:** One of the earliest applications of predictive analytics technology
- **Shelf management:** A logical follow-on to sales forecasting
- **Scientific discovery:** A way to identify which among many billions of stellar objects in an image are worthy of attention
- **Gaming:** A method of predicting which customers have the highest potential for spending
- **Sports:** A method of discovering which players/game situations have the highest potential for high scoring
- **Customer relationship management:** Retention, cross-sell/up-sell propensity
- **Customer acquisition:** A way to identify the prospects most likely to respond to a membership offer

Major issues in data analytics

Some major issues of data analytics include the following (adapted from [Han and Kamber, 2006](#)):

1. **Discovery of different kinds of information in databases:** It is necessary to integrate data from diverse input sources, including data warehouses/data marts, Excel spreadsheets, text documents, and image data. This integration may be quite complex and time consuming.

2. **Interactive discovery of knowledge at multiple levels of abstraction:** Account-level data must be combined with individual-level data and coordinated with data with different time-grains (daily, monthly, etc.). This issue requires careful transformation of each type of input data to make them consistent with each other.
3. **Incorporation of background information:** Some of the most powerful predictor variables are those gathered from outside the corporate database. These data can include demographic and firmographic data, historical data, and other third-party data. Integration of this external data with internal data can be very tricky and imprecise. Inexact (“fuzzy”) matching is necessary in many cases. This process can be very time consuming also.
4. **Data access query languages and ad hoc data access:** Analysts must interface closely with database management systems to access data. Structured Query Language (SQL) is the most common query tool used to extract data from large databases. Sometimes, specialized query languages must be used in the place of SQL. This requirement means that data miners must become proficient (at least to some extent) in the programming skills with these languages. This is the most important interface between data analytics and database management operations.
5. **Presentation and visualization of data analytics results:** Presenting highly technical results to nontechnical managers can be very challenging. Graphics and visualizations of the result data can be very valuable to communicate properly with managers who are more graphical rather than numerical in their analytical skills.
6. **Handling “noisy” or incomplete data:** Many items of data (“fields”) for a given customer or account (a “record”) are often blank. One of the most challenging tasks in data analytics is filling those blanks with intuitive values. We will discuss some approaches for filling blank data fields in [Chapter 4](#). In addition to data that is not there, some data present in a record represent randomness and are analogous to noise in a signal transmission. Different analytical algorithms have different sensitivities to missing data and noise. Part of the art of data analytics is selecting the algorithm with the right balance of sensitivity to these “distractions” and also having a relatively high potential to recognize the target pattern.
7. **Pattern evaluation – the “interestingness” problem:** Many patterns may exist in a data set. The challenge for data analytics is to distinguish

those patterns that are “interesting” and useful to solve the data analytics problem at hand. Various measures of interestingness have been proposed for selecting and ranking patterns according to their potential interest to the user. Applying good measures of interestingness can highlight those variables likely to contribute significantly to the model and eliminate unnecessary variables. This activity can save much time and computing “cycles” in the model-building process.

8. Efficiency and scalability of data analytics

algorithms: Efficiency of a data analytics algorithm can be measured in terms of its predictive power and the time it takes to produce a model. Scalability issues can arise when an algorithm or model built on a relatively small data set is applied to a much larger data set. Good analytics algorithms and models are linearly scalable; that is, time consumed in processing increases geometrically rather than exponentially with the size of the data set.

9. Parallel, distributed, and incremental mining

algorithms: Large analytical problems can be processed much more efficiently by “dividing and conquering” the problem with multiple processors in parallel computers. Another strategy for processing large data sets is to distribute the processing to multiple computers and compose the results from the combined outputs. Finally, some data analytics problems (e.g., power grid controls) must be solved by using incremental algorithms, or those that work on continuous streams of data, rather than large “chunks” of data. A good example of such an algorithm is a Generalized Regression Neural Net (GRNN). Many power grids are controlled by GRNNs.

10. Handling of relational and complex types of data:

Much input data might come from relational databases (a system of “normalized” tables linked together by common keys). Other input data might come from complex multidimensional databases (elaborations of star-schemas). The data analytics process must be flexible enough to encompass both.

11. Mining information from heterogeneous and global information systems: Analytical tools must have the ability to process data input from very different database structures. In tools with graphical user interfaces (GUIs), multiple nodes must be configured to input data from very different data structures.

These requirements have not changed in the last 10 years and neither have the requirements for success in a data analytics project.

General requirements for success in a data analytics project

Following are general requirements for success of a data analytics project:

1. Significant gain is expected. Usually, either:
 - a. Results will identify “low-hanging fruit,” as in a customer acquisition model where analytic techniques haven’t been tried before (and anything rational will work better), or
 - b. Improved results can be highly leveraged; that is, an incremental improvement in a vital process will have a strong bottom line impact. For instance, reducing “charge-offs” in credit scoring from 10% to 9.8% could make a difference of millions of dollars.
2. A team is skilled in each required activity. For other than very small projects, it is unlikely that one person will be sufficiently skilled in all activities. Even if that is so, one person will not have the time to do it all, including data extraction, data integration, analytical modeling, and report generation and presentation. But more importantly, the analytic and business people must cooperate closely so that analytic expertise can build on the existing domain and process knowledge.
3. Data vigilance: Capture and maintain the accumulating information stream (e.g., model results from a series of marketing campaigns).
4. Time: Learning occurs over multiple cycles. Early models can be improved by performing error analyses, which can point to changes in the data preparation and modeling methodology to improve future models.
5. Also, champion-challenger tests with multiple algorithms can produce models with enhanced predictability. Successive iterations of model enhancement can generate successive increases in success.

Each of these types of data analytics applications followed a common methodology in principle. We will expand on the subject of the data analytics process in [Chapter 3](#).

Example of a data analytics project: classify a bat's species by its sound

Approach:

1. Use time-frequency features of echolocation signals to classify bat species in the field (no capture is necessary).
2. University of Illinois biologists gathered 98 signals from 19 bats representing 6 species ([Melendez et al., 2006](#)).

3. Thirty-five data features were calculated from the signals, such as low frequency at the 3 dB level; time position of the signal peak; and amplitude ratio of the first and second harmonics.
4. Multiple data analytics algorithms were employed to relate the features to the species.

The groupings of bat signals are depicted in terms of color and shape of the plotted symbols. From these groupings, we can see that it is likely that a modeling algorithm could distinguish among many of them (as many colored groups cluster) but not all (as there are multiple clusters for most bat types). The first set of models used decision trees and was 46% accurate. A second set used a new tree algorithm that looks two steps ahead (see TX2Step on <http://64.78.4.148/PRODUCTS/tabid/56/Default.aspx>) and did better at 58%. A third set of models used neural nets with different configurations of inputs. The best neural net solution increased the correct prediction rate to 68%, and it was observed that the simplest neural net architecture (the one with the fewest input variables) did best. The reduced set of inputs for the neural networks had been suggested by the inputs chosen by the two decision tree algorithms. Further models with nearest neighbors, using this same reduced set of inputs, also did as well as the best neural networks. Lastly, an ensemble of the estimates from four different types of models did better than any of the individual models.

The bat signal modeling example illustrates several key points in the process of creating data analytics models:

1. Multiple algorithms are better than a single algorithm.
2. Multiple configurations of an algorithm permit identification of the best configuration, which yields the best model.
3. Iteration is important and is the only way to ensure that the final model is the right one for a given application.
4. Data analytics can speed up the solution cycle, allowing you to concentrate on higher aspects of the problem.
5. Data analytics can address information gaps in difficult-to-characterize problems.

Fig. 2.3 shows a plot of the 98 bat signals.

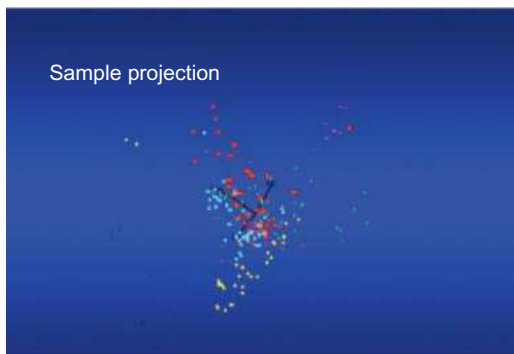


FIG. 2.3 Sample plot of bat signals.

The importance of domain knowledge

One data analyst might build a model with a data set and find very low predictability in the results. Another analyst might start with the same data set but create a model with a much higher predictability. Why the difference? In most cases like this, the difference is in the data preparation, not the modeling algorithm chosen. Although some algorithms may be superior to others to model a given data set, the best model is no better than the predictor variables input to it. The second analyst may know much more about the business domain from which the data came. This intimate knowledge facilitates the derivation of powerful predictor variables from the set of existing variables. In Chapter 21, we will see that the derivation of time-lagged variables (a.k.a. temporal abstractions) permitted the creation of a much more powerful model than without them. There is simply no substitution for domain knowledge. If you don't have it, get it by either learning it before building a model or bring it into the project team in the form of one who does know it.

Caveats with data analytics solutions

Hand (2005) summarized some warnings about using data analytics tools for pattern discovery:

1. **Data quality:** Poor data quality may not be explicitly revealed by the data analytics methods, and this poor data quality will produce poor models. It is possible that poor data will support the building of a model with relatively high predictability, but the model will be a fantasy.
2. **Opportunity:** Multiple opportunities can transform the seemingly impossible to a very probable event. Hand refers to this as the *problem of multiplicity*, or the law of truly large numbers. For example, the odds of a person winning the lottery in the United States are extremely small, and the odds of that person winning it twice are fantastically so. But the odds of *someone in the United States* winning it twice (in a given year) are actually better than ever. As another example, you can search the digits of pi for "prophetic" strings such as your birthday or significant dates in history and usually find them – given enough digits.
3. **Interventions:** One unintended result of a data analytics model is that some changes will be made to invalidate it. For example, developing fraud detection models may lead to some effective short-term preventative measures. But soon thereafter, fraudsters may evolve in their behavior to avoid these interventions in their operations.
4. **Separability:** Often, it is difficult to separate the interesting information from the mundane

information in a data set. Many patterns may exist in a data set, but only a few may be of interest to the data miner for solving a given problem. The definition of the target variable is one of the most important factors that determine which pattern the algorithm will find. For one purpose, retention of a customer may be defined very distinctively by using a variable like `close_date` to derive the target. In another case, a 70% decline in customer activity over the last two billing periods might be the best way to define the target variable. The pattern found by the data analytics algorithm for the first case might be very different from that of the second case.

5. **Obviousness:** Some patterns discovered in a data set might not be useful at all because they are quite obvious, even without data analytics analysis. For example, you could find that there is an almost equal number of married men as married women (duh!). Or you could learn that ovarian cancer occurs primarily in women and that check fraud occurs most often for customers with checking accounts.
6. **Nonstationarity:** Nonstationarity occurs when the process that generates a data set changes of its own accord. For example, a model of deer browsing propensity on leaves of certain species will be quite useless when the deer population declines rapidly. Any historical data on browsing will have little relationship to patterns after the population crash.

Why did data analytics arise?

Now we can go on to a final dimension of our subject of analytical theory. Statistical analysis has been around for a long time. Why did data analytics development occur when it did? Necessity may indeed be the mother of invention. During the past 50 years, business, industry, and society have accumulated a huge amount of data. It has been estimated that over 90% of the total knowledge we have now has been learned since 1950. Faced with huge data sets, analysts could bring computers to their “knees” with the processing of classical statistical analyses. A new form of learning was needed. A new approach to decision making based on input data had to be created to work in this environment of huge data sets. Scientists in AI disciplines proposed that we use an approach modeled on the human brain rather than on Fisher’s Parametric Model. From early AI research, neural nets were developed as crude analogs to the human thought process, and decision trees (hierarchical systems of Yes/No answers to questions) were developed as a systematic approach to discovering “truth” in the world around us.

Data analytics approaches were also applied to relatively small data sets, with predictive accuracies equal to or better than statistical techniques. Some medical and pharmaceutical data sets have relatively few cases but many hundreds of thousands of data attributes (fields). One such data set was used in the 2001 KDD Cup competition, which had only about 2000 cases, but each case had over 139,000 attributes! Such data sets are not very tractable with parametric statistical techniques. However, some analytical algorithms (like MARS) can handle data sets like this with relative ease.

Postscript

You may wonder why a whole chapter was devoted to what may appear to be rather esoteric theoretical concerns. The reason is that it is necessary to understand the issues that prompted the development of ML methods amidst the many classical and parametric statistical methods before you can understand the capability of ML methods. In [Chapter 3](#), the general data analytics process is discussed, with a focus on how the process addresses many of the issues and problems with previous analytical methods.

Key words and phrases

Discovery, Hypothesis, Systems, Inductive, Signal, Scientific methodology, Data integration, Business understanding, Data quality, Pattern recognition, Data analysis process.

References

- Accenture Global Services. (2006) 2006 Standardized customer application and record for inputting customer data into analytical applications. US Patent #7047251, May 16, 2006.
- Payyad, U., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996). *Advances in knowledge discovery and data mining*. AAAI Press.
- Frawley, W., Piatetsky-Shapiro, G., & Matheus, C. (1992). Knowledge discovery in databases. *AI Magazine*, 13(3), 1–30.
- Han, J., Kamber, M. (2006). *Data analytics: Concepts and techniques*, 2nd ed. Morgan Kaufmann.
- Hand, D.J. (2005). *What you get is what you want? Some dangers of black box data analytics*. *M2005 Conference Proceedings*. SAS Institute, Inc.
- Hand, D., Mannila, H., & Smyth, P. (2001). *Principles of data mining* (pp. 578). Cambridge, MA: The MIT Press.
- Mannila, H. (2000). Theoretical frameworks for data analytics. *SIGKDD Explorations*, 1(2), 30–32.
- Melendez, K., Jones, D., & Feng, A. (2006). Classification of communication signals of the little brown bat. *The Journal of the Acoustical Society of America*, 120, 1095–1102.

The data analytics process

Preamble

Data analytics practitioners are fond of saying that data analytics is as much art as it is science. What they mean by this statement is that the data analytics process is a scientific endeavor overlain with a significant amount of artistic practice. This chapter will expand on this statement in the context of the many practical challenges of data analytics in working with real-world data sets.

Two approaches to understanding and problem solving

Before an investigation can occur following any process design, the basic approach must be defined. There are two basic approaches to discovering truth: the *deductive* approach and the *inductive* approach. The deductive approach starts with a few axioms – simple true statements about how the world works. The understanding of the phenomenon can be deduced from the nature of the axioms. This approach works fine in mathematics, but it does not work very well for describing how the natural world works. During the Renaissance, an alternate approach to truth was formulated, which turned the deductive method upside down. This new method approached truth *inductively* rather than *deductively*. That is, the definition of simple truths describing the phenomenon is the *goal* of the investigation, not the *starting point*! Development of the inductive approach to truth in science led to the formation of the scientific method.

In his classic work on problem solving, George Pólya showed that the mathematical method and the scientific method are similar in their use of an iterative approach but differ in steps followed (Pólya, 1957). Table 3.1 compares these two methods.

The method followed in the data analytics process for business is a blend of the mathematical and scientific methods. The basic data analytics process flow follows the mathematical method, but some steps from the scientific method are included (i.e., characterization, and test and experiment).

The role of process in data analytics

Chapter 2 introduced the term *knowledge discovery*, referring to the collateral process of data access, data preprocessing, data postprocessing, and interpretation of results. The combined process was referred in the mid-1990s as the *KDD process* (Fig. 3.1).

This is a very useful approach to express all the steps necessary for finding and exploiting relationships in data; however, it was expressed as a waterfall process, characterized by the unidirectional flow of information from one phase to another. Unfortunately, reality does not follow that strict waterfall design. The circular arrows do indicate the iterative nature within a processing phase, but there is no indication of iteration *between* phases; this almost always occurs in the processing flow of real-world systems.

The flow of data from one phase of analytical operations to the next phase includes iteration within each phase but also iteration between phases (backward data flow). Therefore data flow may follow an overall feed-forward flow shown in the waterfall process design of Fig. 3.1 and also backflow of data to a previous phase. In addition, the overall feed-forward design is not strictly linear but circular, because results from one model are used to refine the model design to produce an updated model. This circular data flow design with iterations within phases and backflow to previous phases is expressed in the Cross Industry Standard Process for Data Mining (CRISP-DM) format.

CRISP-DM

The CRISP-DM format for expressing the predictive analytic/data analytics process is the most complete available. It was created in 1998 by a consortium of NCR, Integral Solutions Limited, and Daimler-Benz companies. The process defines a hierarchy consisting of six major phases, generic tasks, specialized tasks, and process instances. The major phases of CRISP-DM are related in Fig. 3.2 as it is applied to fraud modeling.

TABLE 3.1 Comparison of the steps in the mathematical and scientific methods

Mathematical method	Scientific method
1. Understanding	1. Characterization from experience and observation
2. Analysis	2. Hypothesis: a proposed explanation
3. Synthesis	3. Deduction: prediction from hypothesis
4. Review/extend	4. Test and experiment

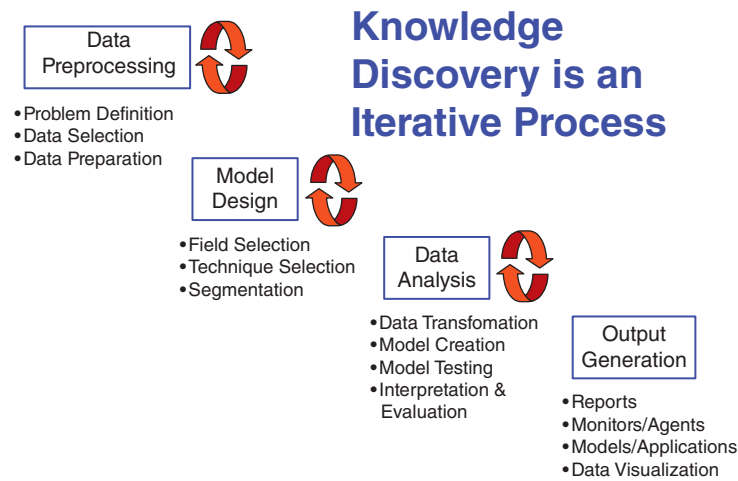


FIG. 3.1 Knowledge discovery expressed as a waterfall process.

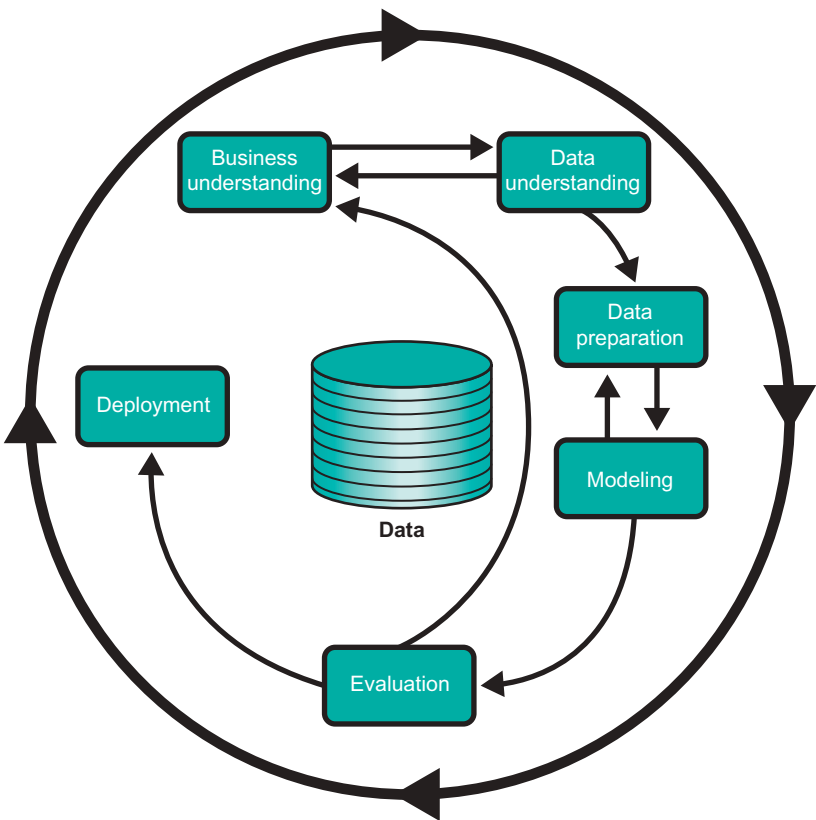


FIG. 3.2 Phases of the circular CRISP-DM process with backflow loops.

Each phase of the CRISP-DM process shown in Fig. 3.2 consists of a number of second-level generic activities, each with several specialized operations. A fourth level (tasks) could be defined in this process, but these tasks are very domain specific; that is, they must be defined in terms of the specific business problem to be solved in the context of the specific data used to solve it. This organization can be viewed in terms of the following hierarchy:

Data analytics phases, activities, operations, and tasks

The expanded data flow process hierarchy described in the following sections is based largely on Chapman et al. (2000), but some activities have been added (shown with asterisks). Each phase in the list is annotated with the degree to which it pertains to art or science. The art of data analytics will be discussed in further detail in the section titled “The art of data analytics.”

Business understanding phase (mostly art)

Before you can begin data analytics, you must have a clear understanding of what you want to do and what success will look like in terms of the business processes that will benefit. Each of the following major tasks that promote understanding of the business problem should be followed in most cases.

Define the business objectives of the data analytics model

You should understand the background that spawned the business needs that a data analytics model might serve. For example, a body of unstructured data might exist in your company, including notes, memos, and reports. Information in these unstructured formats is not present in a database, so it cannot be queried like normal data. The business objective is to find a way to capture relevant information in these unstructured formats into a data format that will support decision making. In this case, a text mining model might be useful to capture relevant information in these documents. An important part of formulating the business objectives is to include individuals from all business units of the company who are affected by the problem and will benefit from its solution (the “stakeholders”). You must compose a set of success criteria from interactions with these stakeholders. Only in that way will you know from the beginning what a “good” model will look like, in terms of metrics accepted by the stakeholders. For example, one telecommunications company in Brazil defined that customer attrition occurred (“churn”) when the average

total minutes of use (MOU) declined more than 70% over the previous two billing periods. In addition to success criteria, all stakeholders should be fully aware of the benefits of success in the project and apprised of its cost in terms of resources (human and financial). A good example is of a Texas bank that defined customer churn with variables that were at least 2 weeks old (e.g., #deposits 2 weeks ago). The model gave the Relationship Manager 2 weeks to present the high-risk churners with an offer to encourage the customer not to leave the bank (an interdiction strategy). This approach is a very important factor in “engineering” success of the data analytics project.

Assess the business environment for data analytics

Building a data analytics model is a lot like erecting a building. In addition to knowing what the building will look like when it is done, we must plan for its construction. The first thing you must do is to take inventory of your resources. That may mean listing the data integration, data quality, and analytical tools at your disposal. If an important tool is missing, you have to acquire it or figure out how to do specific tasks with the tools you have. A shortfall in tools and materials may increase the risk of schedule slippage (or even failure). Any other significant risks should be identified (e.g., risk of failure of obtaining the necessary approvals from management or for data access), and contingency plans should be formed to handle various possible roadblocks.

In addition to assessing the modeling environment, you should assess one or more deployment environments. Many data analytics models have just sat on the shelf because they were impractical or too costly to implement. Restrictions in the deployment environment might dictate the form and power of the model. For example, if the model is used to guide the Underwriting department to minimize loss risk, model output might be required in the form of business rules. In that case, a decision tree model might be the best choice, and one from which only a few rules must be induced to guide the underwriters. For deployment of a customer retention model, a prediction produced by a neural net model might be used to drive an interdiction campaign to retain high-value customers with relatively high attrition propensities (like the Texas bank did). Agreement from the Marketing department to conduct such a campaign should be obtained before modeling operations begin.

Finally, results of the business assessment should be fully documented, together with sufficient explanatory material and terminology to serve as a standalone document for later reference.

Formulate the analytical goals and objectives of the project

Formulating the analytical goals and objectives of the project might seem moot (in relation to the business goal), but it is critical to the success of the data analytics project. The primary goal of the data analytics exercise is *not* to train a good predictive model per se, but rather to *deploy* a good predictive model to meet the business objective! Often, data miners take this for granted. But it does not happen automatically. Well-deployed models must be engineered rather than just envisioned. Many models that have been envisioned for their usefulness in a company have been relegated to the shelf (as it were) because they could not be implemented efficiently and effectively.

Analytical project goals may include the following:

- Building (or helping to build) a suitable database from which modeling data sets can be extracted easily
- Developing and deploying a model which generates significant business value
- Building a knowledge base of modeling “learnings” which can be leveraged later to do a better job with data analytics (easier, faster, cheaper)

Each of these data analytics goals is associated with a set of objectives. For example, a good set of objectives for the goal of developing and deploying a model can include the following:

- Acquiring a suitable data set for modeling
- Creating a short list of predictor variables
- Creating a model of acceptable accuracy
- Deploying the model in production operations
- Monitoring for acceptable performance
- Updating the model with current data
- Providing feedback on intelligence gained by application of the model

Each of these objectives will be implemented by a set of tasks. An example of a task list for the objective of creating a short list of predictor variables might include these tasks:

- Identification/derivation of the target variable (the variable to be predicted)
- Univariate and bivariate analysis of candidate predictor variables
- Multivariate analysis of candidate predictor variables
- Correlation analysis of candidate predictor variables
- Preliminary screening of variables with various metrics and screening algorithms (e.g., Gini scoring or with some measure of “interestingness”)
- Preliminary modeling of the target variable (e.g., with a decision tree algorithm) to select variables for the short list

Each of these tasks will be composed of subtasks or steps followed to accomplish the task. For example, steps followed in the univariate or bivariate analysis task could include:

- Generation of various descriptive statistics (e.g., means, standard deviations, etc.)
- Bivariate scatterplots
- Association and linkage analysis

Perhaps by now, you can see where we must go next with this body of methodology. Of course, we must compose all the objectives, tasks, and steps into a project plan and then manage that plan. We must “plan the work” and then “work the plan.” This plan should assign start dates and end dates to each objective, task, and step and identify the resources needed to accomplish it (people, money, and equipment).

Microsoft Project is a good project planning package to use for project plan formation and tracking. The embedded goal of good project planning is to finish the plan ahead of schedule and under budget. Much has been written about how to do this, so we won’t go into this topic any further here. For more information, buy a good book on project planning. Much good information on project planning is available on the Internet, notably from the Project Management Institute (PMI). PMI certification as a Project Management Professional (PMP) is a good adjunct to successful data analytics.

Data understanding phase (mostly science)

The objectives and tasks in this and subsequent sections will be presented in a rough outline format.

An expanded outline is available on the DVD, which explains in general terms some of the operations usually performed for each objective of a database marketing project. (See database marketing documents on the DVD.) The asterisks next to some of the tasks indicate those added to the standard CRISP-DM methodology document. Where possible, the tutorials will refer to the specific data analytics objectives and tasks performed and why they are necessary for the data set and modeling goal in focus.

The CRISP-DM activity for data understanding was specified separately in the diagram, but in this book, we will treat it together with data preparation. Following are some of the common data understanding activities:

1. Data acquisition
 - a. Data access* Tasks included to augment those in [Chapman et al. \(2000\)](#).
 - b. Data integration*
 - c. Initial data collection report

2. Data description
 - a. Variables*
 - b. Cases*
 - c. Descriptive statistics*
 - d. Data description report
3. Data quality assessment
 - a. Missing values*
 - b. Outliers*
 - c. Data quality report

Data acquisition

Before you can do anything with your data, you have to acquire it. This statement appears on the surface to be self-evident. Determining how to find and extract the right data for modeling, however, is not at all self-evident. First, you have to identify the various data sources available to you. These data may be nicely gathered together in an enterprise data warehouse. While this situation is ideal, most situations will be quite different. Data may be scattered in many business units of the company in various data “silos,” spreadsheets, files, and hard copy (paper) lists (Fig. 3.3). Your next challenge is to put all this information together.

Data integration

Combining data sets is not an easy thing to do. Usually, data are in different formats or exist in different levels of aggregation or are expressed in different units. A big part of the integration activity is to build a data

map, which expresses how each data element in each data set must be prepared to express it in a common format and a common record structure. Data in relational databases must be either “flattened” (gathered together into one row or record) or the data map must be traversed to access the data in the databases directly through in-database access utilities, available in some analytical tools. The advantage of doing in-database mining is that your data do not have to be extracted into an external file for processing; many data analytics tools can operate on the data quite nicely right where they are. The disadvantage of in-database mining is that all data must be submitted to the necessary preprocessing activities to make them useful for mining. This preprocessing takes time! So one of the advantages of extracting data to an external file is that processing and modeling can be performed much faster, particularly if your modeling data set can be contained in main memory.

In addition to extraction (E), many fields of data must be transformed (T) and new fields (variables) derived. Fig. 3.4 illustrates several of the kinds of transforms that must be done before the data can be loaded (L) into the customer analytic database. This ETL (extract-transform-load) process is very similar to that followed in building enterprise data warehouses.

Data description

You can make some big mistakes by beginning to prepare data before you adequately describe those data. Data description tasks are not just fodder for a

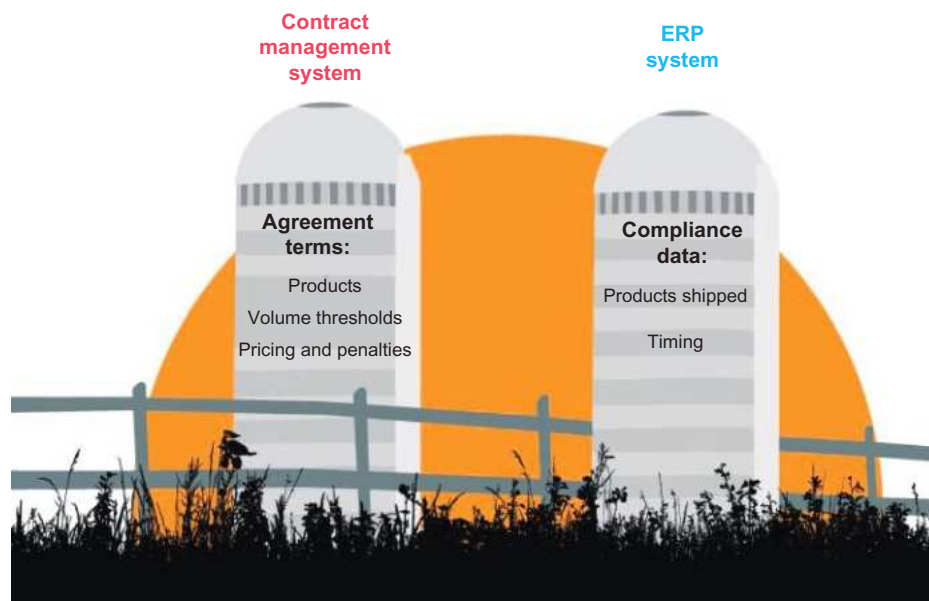


FIG. 3.3 Data in separate storage systems (“silos”) must be accessed and integrated into the customer analytic record. Source: https://www2.iaccm.com/images/content/1421862801_fig1.jpg.

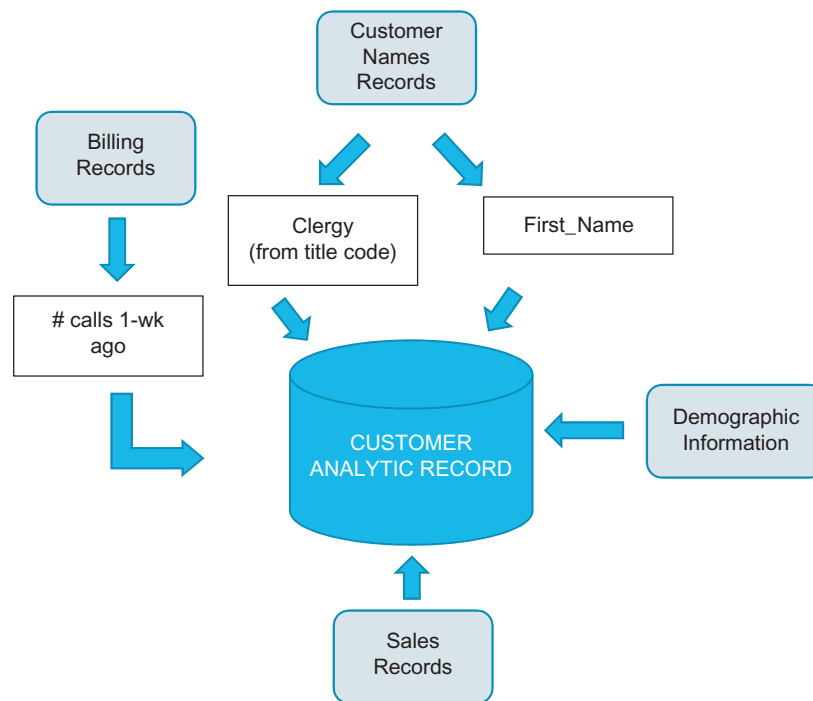


FIG. 3.4 The data integration processes of Extract (E), Transform (T), and Load (L) to build the customer analytic record.

data “readiness” report. These tasks must be properly fit to the characteristics of your data. Before you can do that, you must *know* your data. For each data element available to you, look through as many data records as you can to get a general feeling for each variable. Statistical packages and most data analytics tools have some simple descriptive statistical capabilities to help you characterize your data set. See a more detailed discussion of descriptive data operations in Chapter 6.

Data quality assessment

I have heard the same refrain many times from data warehousing technicians: “Our data is clean!” But it never is. The reason for this misconception is that analytics require a special kind of “cleanliness.” Most analytical algorithms cannot accept blanks in any variable used for prediction. One of the hardest and most important operations in data preparation is filling blank entries in all variables. This process is called *imputation*, and it can be accomplished by some simple operations and by some rather sophisticated operations. Another vexing problem in data sets is outliers, or values beyond the normal range of the response you are modeling. A third kind of uncleanness is the presence of several coding systems in the historical records of the same database field. These codes must be reconciled across all historical records.

Data preparation phase (a mixture of art and science)

Basic data preparation operations access, transform, and condition data to create a data set in the proper format suitable for analytical modeling. The major problem with data extracted from databases is that the underlying structure of the data set is not compatible with most statistical and data analytics algorithms. Most data in databases are stored at the account level, often in a series of time-stamped activities (e.g., sales). One of the greatest challenges is rearranging these data to express responses on the basis of the entity to be modeled. For example, customer sales records must be gathered together in the same row of a data set for each customer. Additional preparation must be done to condition the data set to fit the input requirements of the modeling algorithm. There are a number of basic issues that must be addressed in this process.

Basic issues that must be resolved in data preparation:

- How do I clean up the data? – Data Cleansing
- How do I express data variables? – Data Transformation
- How do I handle missing values? – Data Imputation
- Are all cases treated the same? – Data Weighting and Balancing
- What do I do about outliers and other unwanted data? – Data Filtering

- How do I handle temporal (time-series) data? – Data Abstraction
- Can I reduce the amount of data to use? – Data Reduction Records? – Data Sampling
- Variables? – Dimensionality Reduction
- Values? – Data Discretization
- Can I create some new variables? – Data Derivation

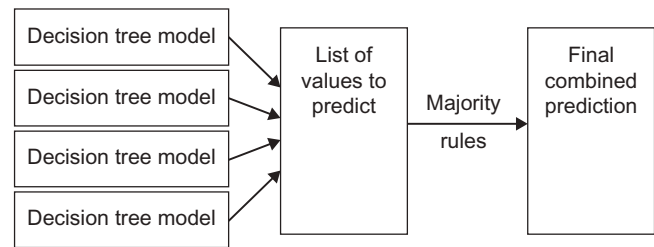


FIG. 3.5 Flow chart of operations followed in building an ensemble model.

Modeling (a mixture of art and science)

A general discussion of modeling activities is presented in the following sections. You can take a “deep dive” into some of these activities in the ensuing chapters. For example, more detailed presentations of the modeling operations are presented in [Chapters 5–15](#).

Steps in the modeling phase of CRISP-DM

Note that modeling activities with asterisks have been added to the CRISP-DM list of activities.

1. Select modeling techniques.
 - a. *Choose modeling algorithms**. The methods you choose to prepare your data will depend to some degree on which modeling algorithm you choose. If you choose a parametric statistical algorithm (such as multiple linear regression), you may have to transform some variables to account for significant nonlinearity. If you choose a Support Vector Machine, you might have to standardize your data to fit its requirements.
 - b. *Choose modeling architecture* (single analysis, ensemble, etc.)*. A simple, straightforward analysis will include submitting your data to the algorithm and evaluating the models created. Sometimes that is all you have to do, and sometimes it is not. There are many ways to enhance this simplistic approach to refine your models and improve their performance. Some algorithms, like neural nets, permit you to adjust the algorithm architecture to improve performance (add hidden layers or increase the learning rate). Even these techniques may not be sufficient to optimize performance. You can create a series of models using different algorithms (ensembles), or you can model on different samples of data and compare or combine results (bootstrap, jackknife resampling, and V-fold cross validation). Finally, you can build some simple feedback processes in your models to iteratively improve your model (boosting). [Fig. 3.5](#) shows how an ensemble model works.

- c. *Specify modeling assumptions*. Every modeling algorithm makes assumptions. Your challenge is to choose an algorithm whose assumptions fit your data and your modeling goal. For example, you can use a multiple linear regression algorithm safely if your data set does not violate significantly any of the important assumptions of the Parametric Model (the body of assumptions behind parametric statistical theory). You can use a neural net for classification if your target variable is categorical. Among neural nets, a Radial Basis Function (RBF) neural net can handle outliers better than an ordinary neural net. Many modeling tools provide both kinds of neural nets. If data outliers are important in your data environment, an RBF neural net would be a good choice.

2. Create an experimental design.

Many analytical projects fail because the experimental design is faulty. It is very important to include an analysis of the response under normal conditions to compare results to those under various treatments. For example, an article in the *London Times* on April 22, 2007, described a study of cancer clusters around seven cell phone towers in different parts of the UK. But there are about 47,000 cell phone towers in the UK. This means that only about 0.015% of the towers were included in the study. The researchers could have said just as easily that cell phone towers prevent cancer 99.985% of the time! In other words, there was no control. A proper control study would have been to analyze cancer occurrence near a random sample of the 47,000 towers in the UK and compare the results with those of the cancer clusters. Maybe the cell phone towers had nothing whatsoever to do with causing cancer. A more frightening statistic can be gleaned from an article in the *Journal of the National Cancer Institute* ([Murray et al., 2008](#)), in which the authors found that in 75 articles on cancer prevention and control published in 41 journals, only about half of the studies had used a sufficiently rigorous experimen-

tal design and had used an appropriate algorithm necessary to justify their conclusions. The next time you read about a report of a cancer study in the newspaper, take it with a grain of salt. You might get just as good an insight from flipping a coin. (Note: In a double-blind experiment, neither the individuals nor the researchers know who belongs to the control group and the experimental group.) These warnings should not scare you away from doing studies like these, but they may inoculate you against these errors and help you to avoid them. When George Santayana (2018) quipped, “Those who cannot remember the past are condemned to repeat it,” he was referring to the errors of history. We will not repeat the analytical errors of those in the studies reported by Murray et al. (2008) if we are aware of the dangers of bad experimental design and take steps to correct them before we start the data analytics process.

3. Build the model.

Model building is mostly art and will be discussed in greater detail in the section “The art of data analytics.” Here, we can consider the steps to follow in building an analytical model. The general steps in modeling are as follows:

a. Set parameters (if the algorithm is not automatic).

Many modeling algorithms (like neural nets) start with various default settings. Study the defaults and the theory behind these settings. The algorithm settings are programmed into the function of the model for a reason. Often, the reason is that different data sets require slightly different model settings. The default settings are a good place to start. Create other models with different settings and see what happens. You may (and often do) find that subsequent models are more powerful predictions compared to the default model.

b. Build various types of models. Using one type of algorithm to model a data set is good, but using multiple algorithms is far better. One algorithm gives one “perspective” on the information pattern in your data set, like looking at the world with one eye. However, multiple algorithms will give you multiple perspectives on your information patterns. Let them “vote” on which predicted value or category is right for a case. Then follow some heuristic (decision rule) to decide which predicted value is to be accepted. You can pick the average of numerical values or let the majority rule in classification. Such a modeling tactic is called *ensemble modeling* (refer to Fig. 3.4). Another way to use multiple models is to use resampling methods to build models on different randomly selected subsets of data.

4. Assess the model (mostly science).

What is a good model? The best way to determine this is to wait until you can verify the predictions in reality. But you can’t wait until then to evaluate various models. You must compare your candidate models several times during the course of analytical modeling. The most common way to evaluate a model is to compare it to what you would expect to happen if you did not use the model. There are some very effective techniques for doing this, using various tables and graphs (coincidence tables, lift charts, ROI curves, and normal probability charts). Also, there are some statistical measures of error. These model assessment techniques (and others) will be discussed as a part of the discussion of model assessment in Chapter 15. Model assessment is one of the iterative activities performed in modeling. Models should be assessed by one or more of the assessment techniques, which may give some insight into where the model failed. Model parameters can then be adjusted with the help of this insight and a new model built. This process expresses the CRISP-DM data analytics cycle in a more concrete form. This cycle continues until the assessment techniques converge on an optimum predictive power. Statisticians use the term *convergence* to express the point in the diminishing returns of error minimization that reaches some predetermined expression of the minimum error. For example, a standard error statistic of 0.01 might be set as the stopping point of convergence in a regression algorithm. Sometimes a minimum rate of convergence is selected as the stopping point. Neural nets use analogous stopping functions to end model training.

5. Evaluate the model (mostly science).

After you create the best model you can under circumstances of time, budget, and practicality, it is time to evaluate results, review the process as a whole, and determine the next steps to follow with future modeling efforts. A good approach to accomplish these tasks is to include them in a modeling report. This report will help to synthesize conclusions to form general insights about the modeling effort and to point the way to improving the results during the next modeling effort. We will explore this topic fully in Chapter 15 also.

Evaluating results of the model may be easy, or it can be rather difficult. An example of an easy evaluation is to observe the number of correct predictions compared to the total number of predictions. If that percentage is relatively high, you might conclude that the model was a success. There is a weakness in this approach, however, because on this basis alone, you can’t say that this high predictability would not have happened without the model. Ah, we come

back to the importance of the experimental design. It becomes clearer now that without a good experimental design, evaluation of the model results is moot.

Difficulties of a different kind occur when we try to evaluate what the model did not find but should have (based on actual data). These errors are called *false-negatives* and can occur when your cancer risk model predicts very low cancer risk. It might take years to confirm whether or not that prediction was good and then it might be too late – the patient is dead.

A more insidious error may arise in evaluating results that the model did not find but *might have found* if we had used different data. This type of error points to the very heart of the difference between standard parametric methods and Bayesian methods. Bayesian analysis can incorporate results from past studies, along with data from a present study, to predict more globally what might happen in the future. Parametric or Bayesian methods, which approach is best? Here again, the answer depends on the nature of your problem and the experimental design you choose to solve it. If you want to make a good prediction under all types of cases that might occur, then you must create an experimental design to do that. You may be able to use parametric methods to do it if you include data likely to cover all case types and you use the right sampling methods to select your data set for analysis. On the other hand, it may be much more efficient to use a Bayesian method (like a Naïve Bayesian belief net) to incorporate results from past studies, which included case types for which you have no available data. Digging for nuggets of information in a large data warehouse is like digging for fossils in the ground. At the very outset, we must say “Caveat fossor” (let the digger, or the miner, beware).

Fig. 3.6 shows a gold miner panning for gold. The data miner must be just as patient and meticulous in separating valuable information from other information as the gold miner must separate the flakes of gold from the sand around it. Others who see the glimmer in the pan must confirm that it is not iron pyrite (fool’s gold).

After you evaluate the results, you should evaluate the entire process that generated them. This part of the evaluation should consider your modeling goals, your results, and their relationship to the negotiated criteria for success. You should list what went right with the project and what went wrong. One outcome of such an evaluation might be that the results could have been better (that is, you could have built a better model), but the success criteria did meet the goals of the data analytics project. Such a project can be deemed a success in the present, but it can also point to ways to accomplish higher goals in future projects.

Finally, evaluation of modeling results should include a list of possible modeling goals for the future and the



FIG. 3.6 A gold miner panning for gold.

modeling approaches to accomplish them. The modeling report should discuss briefly the next steps and how to accomplish them. These steps should be expressed in terms of what support must be gained among the stakeholders targeted by these new projects, the processes in the company that must be put in place to accomplish the new projects, and the expected benefit to the company for doing so. In other words, the *business case* for future studies must be built on the merits of the present study. This requirement is one of the major differences between the practice of data analytics in business and academia.

Deployment (mostly art)

1. Create deployment plan
2. Plan model monitoring and maintenance
3. Produce final written report
4. Produce final modeling presentation
5. Review project

Closing the information loop* (art)

Dashed arrows could be added in the CRISP-DM diagram in Fig. 3.2 to indicate:

- Feedback of model deployment results to the database*
- Feedback of model deployment results to the business understanding phase*

The art of data analytics

As discussed in Chapter 2, creating a model of relationships in a data set is somewhat like sculpting a statue. The sculptor follows an iterative process of

chipping and viewing until the finished statue emerges. Likewise, the data miner continues modifying and tuning the model until there are no further improvements in the predictability of the model. This analogy is rather crude, but it does serve to illustrate the point that a large part of the data analytics process is very artistic!

Artistic steps in data analytics

Deriving new variables

Often, several of the strongest predictors in your model will be those you derive yourself. These derived variables can be the transforms of existing variables. Common transforms employ

- Log functions
- Power terms (squares, cubes, etc.)
- Trends in existing variables
- Abstractions (collectives, like *Asian*; temporal, like date-offsets; statistical, like means)

This subject will be discussed in greater detail in [Chapter 10 – Feature Engineering](#).

Selecting predictor variables

The objective of selecting predictor variables is probably the most artistic among all of the data analytics objectives. There are many approaches and methods for selecting the subset of variables (the short list) for use in the model training set. This short list is an extremely valuable piece of information for the data miner to obtain before beginning to train the model. One reason for this importance is that too few variables will generate a model with a relatively low predictability and too many variables will just confuse the model “signal.” This dynamic follows the principle of Occam’s razor, coined by William of Occam (a fourteenth-century clergyman). This principle is often expressed in Latin as the *lex parsimoniae* (“law of parsimony” or “law of succinctness”): *entia non sunt multiplicanda praeter necessitatem*, roughly translated as “entities must not be multiplied beyond necessity.” Data analytics algorithms follow this

principle in their operation. Various methods of developing a short list are presented in [Chapter 11 – Feature Selection](#).

Postscript

Many business people are very closely attuned to the need for policies and well-defined processes necessary to assure profitability; many analytical people in business are not! Mathematicians and statisticians are very cognizant of the calculation processes and the importance of proper experimental design. But for some reason, it is tempting for data miners to stray from the pathway of the correct process to generate their models. Yielding to this temptation may lead data miners off the “narrow path” into the “Slough of Despond” (as it did to Pilgrim in John Bunyan’s *The Pilgrim’s Progress*). Maybe this proclivity is due to being too focused on the powerful technology. Or perhaps the problem lies with the academic background of many data miners, which focuses more on theory than practice in the real world. It is *crucial* to the successful completion of a data analytics project to follow the well-worn path of accepted process. In [Chapters 5–15](#), we will see how this process model is followed in preparing data for data analytics.

Keywords and phrases

Process, Objectives, Goals, Design, Experiment, Error, Predictor variables, Data quality, Data integration, Data Understanding, Data description, Project plan, Assessment techniques.

References

- Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R. (2000). *CRISP-DM 1.0*. SPSS.
- Murray, D., Pals, S., Biltstein, J., Alfano, C., and Lehman, J. (2008). *Journal of the National Cancer Institute* 100 (7), 483–491.
- Santayana, G. (2018). *Life of Reason or The Phases of Human Progress* (pp. 298). CreateSpace Independent Publishing Platform. ISBN-13: 978-1987622140. (originally published in 5 volumes in 1905-6).

Tools

Prelude

The practice of data analytics includes the use of a number of techniques, which have been developed to serve as a set of tools in the toolbox of the analyst. In the early days of data analytics (aka *data mining*), many of these tools had to be built (usually in SQL or Perl) and used in an ad hoc fashion for every job. Many of these functions have been included as separate objects in analytical tool packages or productized separately. Most jobs will require the data analytics analyst to become proficient in at least one of these tool packages, the one chosen by the organization to be used by its analysts.

If a tool package has not been chosen yet, the first decision must be to choose a tool type that fits the analytical tool packages, which can be classified into four major categories:

1. Accessory tools
2. Pure coding languages
3. Menu-based procedural platforms
4. Multipersona graphical programming platforms (aka “No Code/Low Code”)

At first, the huge variety of tools may be inviting as there is a specialist tool for nearly every need. However, it can also seem daunting as it is impossible to learn them all. We will offer some advice for tool selection, organizational adoption, and professional development in this chapter after the discussion of the tool categories.

Accessory tools

Accessory tools permit data analysts to access, integrate, explore, and visualize patterns in data sets before analytical models can be trained. Most of these tools include SQL generation abilities. Many SQL tools are available for extracting data from databases, including MS SQL Server, Linux SQL tools, MySQL, and Embarcadero. These tools can be used to explore the nature of data in databases prior to extraction. They can

be used to extract data also, but other tools (such as ETL tools described below) may serve better. Some data access and data integration tools (e.g., BusinessObjects, DataFlux, DataStage) can serve as SQL generators to access and process data. Some data mining tools offer SQL query capabilities. These tools can be classified as SQL tools, data integration tools, data exploration tools, and data visualization tools.

Data access tools using the SQL language

These tools access data in database structures with the Structured Query Language (SQL). Common tools that provide SQL coding environments include:

1. Microsoft SQL Server
2. Solarwinds SQL Sentry
3. Idera Rapid SQL
4. Microsoft Power BI with SQL generation capabilities
5. BusinessObjects with SQL generation capabilities

Data integration tools using ODBC connectors

These tools access data with online database connectivity (ODBC) connectors configured to retrieve data from specific database structures. A different ODBC connector is required for every database table storage configuration. These tools are used primarily in database and data warehousing operations. They can also be used for integrating data from diverse sources into a series of tables for use in data exploration and further aggregation to form a file suitable for modeling. The following list of data integration tools is just a set of examples of a large number of such tools available in the marketplace.

1. Informatica Power Center
2. AWS Glue
3. Talend Data Fabric
4. IBM Datastage
5. Oracle Data Integrator
6. Alteryx
7. KNIME

Data exploration tools

Data exploration tools facilitate examination of data elements in the integrated tables or the aggregated modeling file for purposes of designing derived variables from specific ranges of elements or combination of data elements. Some common tools used for data exploration for data processing and modeling operations include:

1. Looker
2. Microsoft Power BI
3. Tableau
4. BusinessObjects
5. Alteryx
6. KNIME

Some common data exploration concerns to support modeling operations include:

- Determining data distribution features of data elements
- Ranges of data elements to help identify outliers
- Calculating the number of missing values per data element
- Lists of classes of categorical variables to aid in variable derivation
- Variance and correlation calculations to permit high correlation and low variance screening operations

Data visualization tools

The last set of accessory tools can be used to display relationships among data elements graphically prior to deriving new variables and analytical modeling. Some common data visualization tools include:

1. Tableau
2. Infogram
3. Google Data Studio
4. Grafana
5. Charted
6. Alteryx
7. KNIME

Some common data visualization tasks include:

- Scatter plots showing relationships between values of 2 or multiple data elements
- Pie charts showing relative coverage of classes in a categorical data element
- Line graphs showing relationships between values of two data elements
- Receiver Operating Characteristic (ROC) curves

Pure coding languages

The second category of analytical tools requires scripts to be written in a pure coding language. The most common languages used currently are:

1. SAS
2. R
3. Python

These tools are programming languages complete with a syntax and many libraries of routines and packages for performing most data preparation and analytical modeling operations. The three most common languages used in data analytics are SAS, R, and Python.

SAS

Statistical Analysis System (SAS) was the first computer language dedicated to expressing data manipulation and modeling operations on a common data set format (the SAS data set). SAS became the dominant statistical programming language in the 1980s and was used to develop a huge number of applications in business and industry. Benefits of SAS in the 1990s were the availability of a large number of SAS coders in the marketplace, strong support base, and stable releases. The problem with SAS is that the structure and syntax require considerable experience to express in SAS code the intricacies of many complex data operations. Also, SAS focuses almost exclusively on parametric statistical modeling algorithms in the face of rapid development of machine learning algorithms like artificial neural networks and decision trees in the 1990s.

R

R was developed by Ross Ihaka and Robert Gentleman at the University of Auckland in 1994–2000. Initially, it contained only parametric and nonparametric statistical procedures and time-series analysis. R became a General Public License (GPL) project, and Version 1.0 was released into the public domain. R is used primarily by practitioners in the scientific community. The major benefit of R is that it is open source, and new features and libraries are released directly to the public, all of which are downloadable to the user for free. SAS can cost many thousands of dollars. The syntax of R is much like C++ and (like SAS) requires considerable experience to be used effectively.

SAS vs. R

SAS can be compared to R according to five criteria (source: Sakshi Gupta at <https://www.springboard.com/blog/data-science/sas-vs-r/>; accessed 5/13/2023).

1. Learning curve: SAS is simpler; R is far more complex and may require a long time to learn.
2. Cost: SAS is one of the costliest analytical tools on the market; R is completely free.
3. File sharing: SAS works only with SAS data sets; R can input and output many file formats.
4. Scalability: R works only in RAM memory, while SAS works in streaming mode with no limitations on size of the data structures.
5. Visualization: Visualization features in SAS are relatively limited, with few customization options, while R provides a rich set of visualization libraries and customization options.

Python

Python was the first “kid” on the analytics language block, even before R. It was developed by Guido Van Rossum in 1991 as a very readable language that provided much of the functionality of the C++ and Java languages but without their cryptic syntax. It is reputed that the inspiration for the name came from the BBC’s TV show “Monty Python’s Flying Circus.” The main objective of Python is to provide a readable code base with the ability for developers to add to it to support advanced developments. That is what happened since then – big time! In the early years, Python was primarily used as a general-purpose, object-oriented programming language, but very successfully. Much of Google’s search engine was written in Python. With the arrival of code libraries like scikit-learn in 2007, the primary advanced development area shifted to data analytics. Now almost every known machine learning algorithm and general data manipulation operation is available in Python.

R vs. Python

The basic difference between R and Python is that R is focused primarily on statistical operations on SAS data sets, whereas Python is designed to perform more general data manipulation operations on data sets accessed from various sources. Python includes most of the statistical algorithms included in R. Because up to 90% of the time spent in most data science modeling projects is consumed by data access, data manipulation, and data cleaning operations, Python excels in the mundane side of data science operations. Machine learning algorithms were added to Python in the 1990s, principally in the scikit-learn library. A parallel development occurred in the R world to add machine learning algorithms in R to complement the statistical algorithms it supported initially. This parallel development progressed as a “war” between R and Python for the “soul”

of data analytics, which Python has largely won. Today, most job ads for data analytics positions state that Python expertise is either a requirement or at least a nice-to-have skill set.

Menu-based procedural platforms

A series of embedded menus was the first design for the presentation and selection of specific analytical capabilities in analytical tools. Some prominent menu-based tools include:

1. MATLAB
2. IBM SPSS Statistics
3. Weka
4. Tibco Statistica menu interface

(Note: Statistica also provides a graphical programming interface and a semiautomated Recipes interface.)

Example of a menu interface: Tibco Statistica Data Miner

The Statistica Data Miner (SDM) package from Tibco has moved around a lot from the originator StatSoft through Dell and Quest Software to Tibco. This package is built around analysis of an internal spreadsheet (similar in appearance and function to Excel) using three different user interfaces: menu-based, graphical programming, and semiautomated data mining recipes. Fig. 4.1 shows the top-level menu of SDM to analyze a spreadsheet data set.

Fig. 4.2 shows the rich set of machine learning modeling algorithms available in SDM.

Fig. 4.3 shows the intermediate menu for Random Forests for Regression and Classification to select the classification version of the algorithm to build a model on the spreadsheet data set.

After the user clicks the OK button on the model type selection screen in Fig. 4.3, the Random Forest options screen appears, as shown in Fig. 4.4.

The next step is to specify the target variable (the *dependent* variable) and the predictor variables by clicking on the Variables box in the options menu to display the variables selection screen shown in Fig. 4.5.

The user then clicks OK twice, and SDM trains the Random Forests model. This sequence of menus and configuration screens serves as an example of how the menu-based interface operates.

The final step is to submit data in the selected target and predictor variables to the modeling algorithm. SDM builds Fig. 4.6 in real time and presents the final trained model in terms of training and text classification errors expressed in terms of misclassification rates.

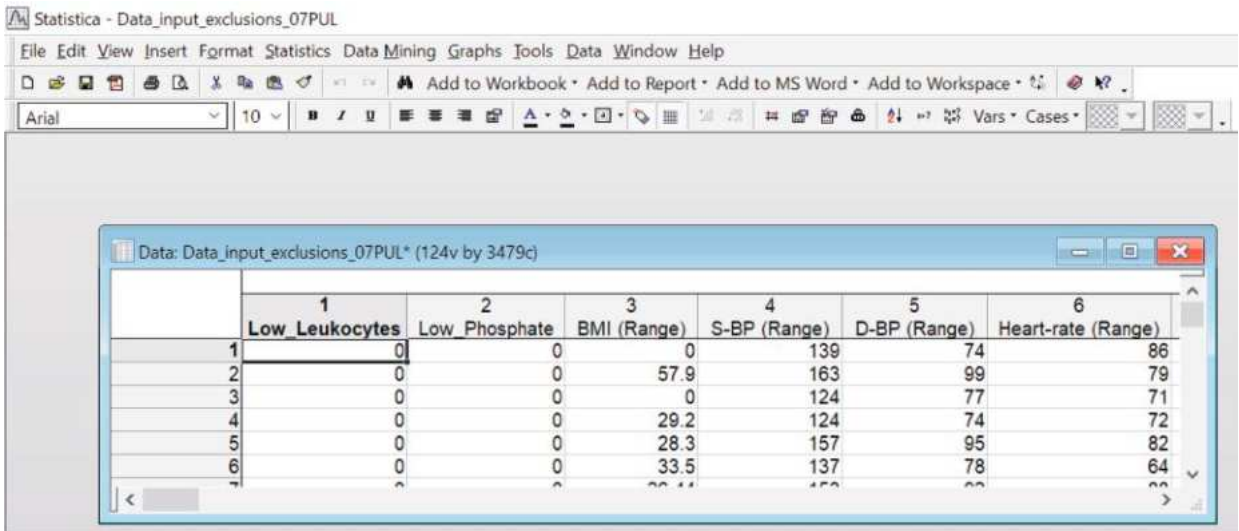


FIG. 4.1 Top-level menu of the SDM interface.

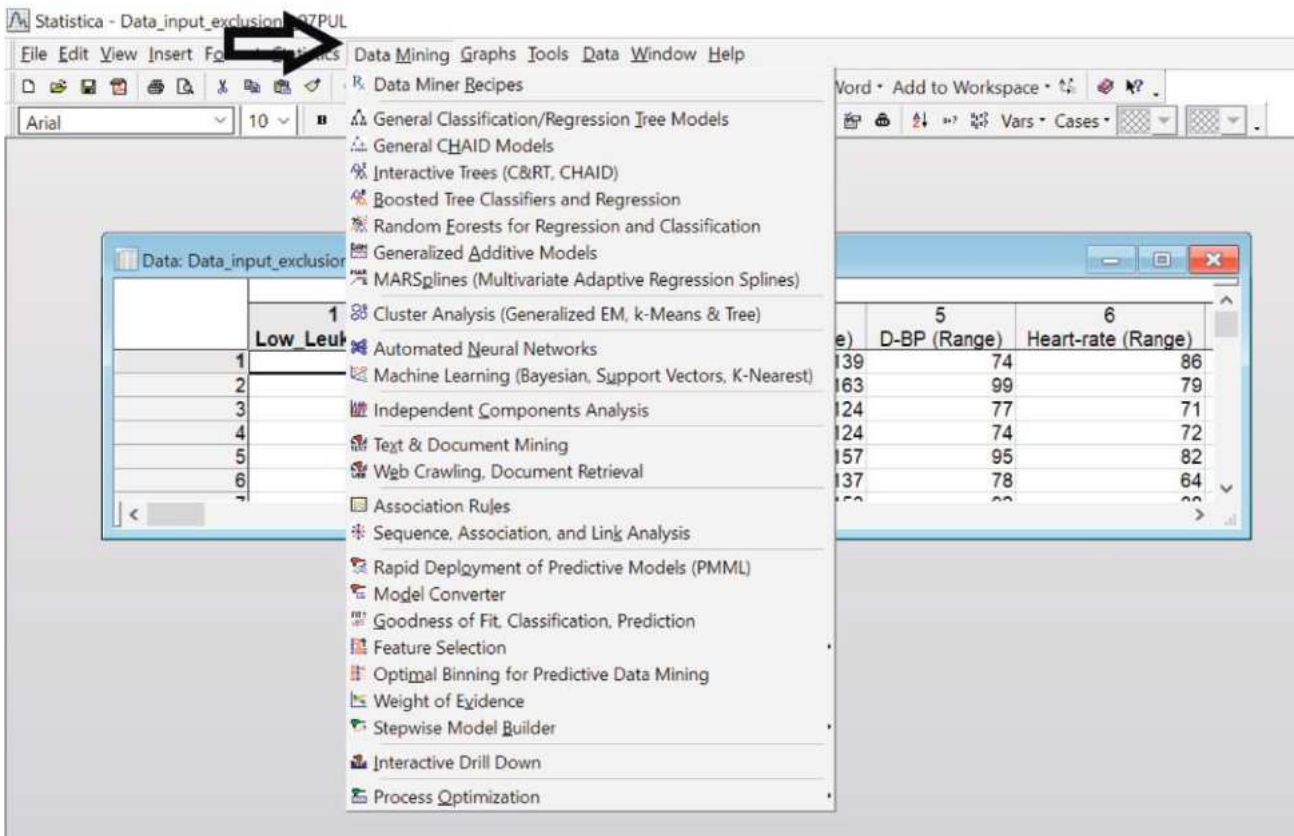


FIG. 4.2 Types of machine learning modeling algorithms available in the SDM submenu.

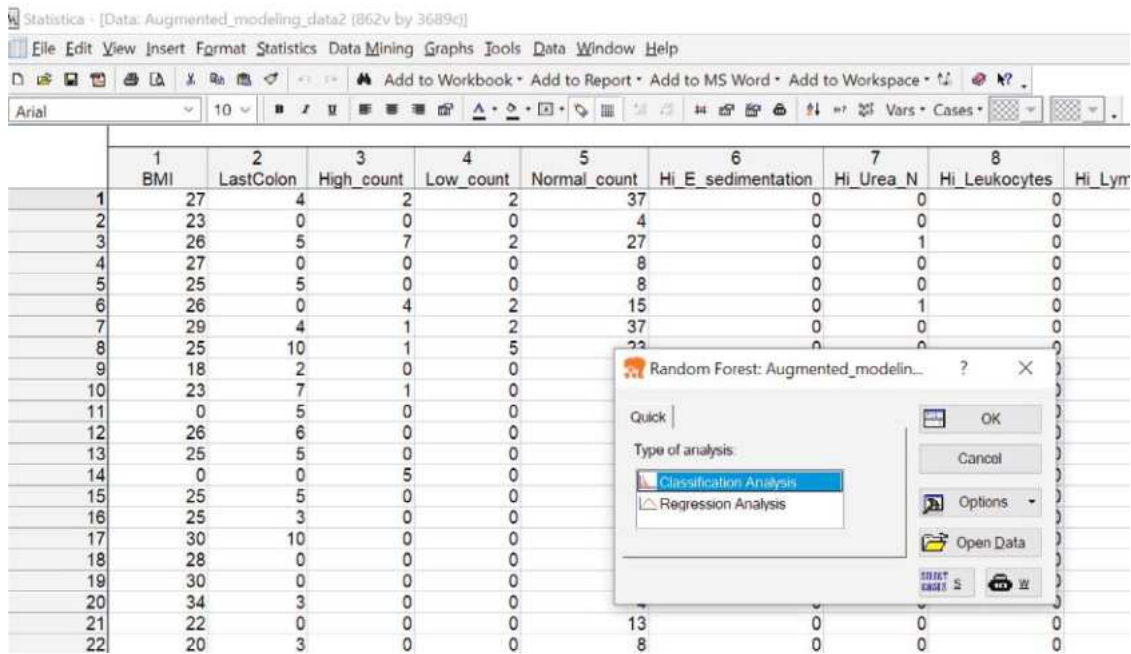


FIG. 4.3 The Random Forest model type selection menu in SDM.

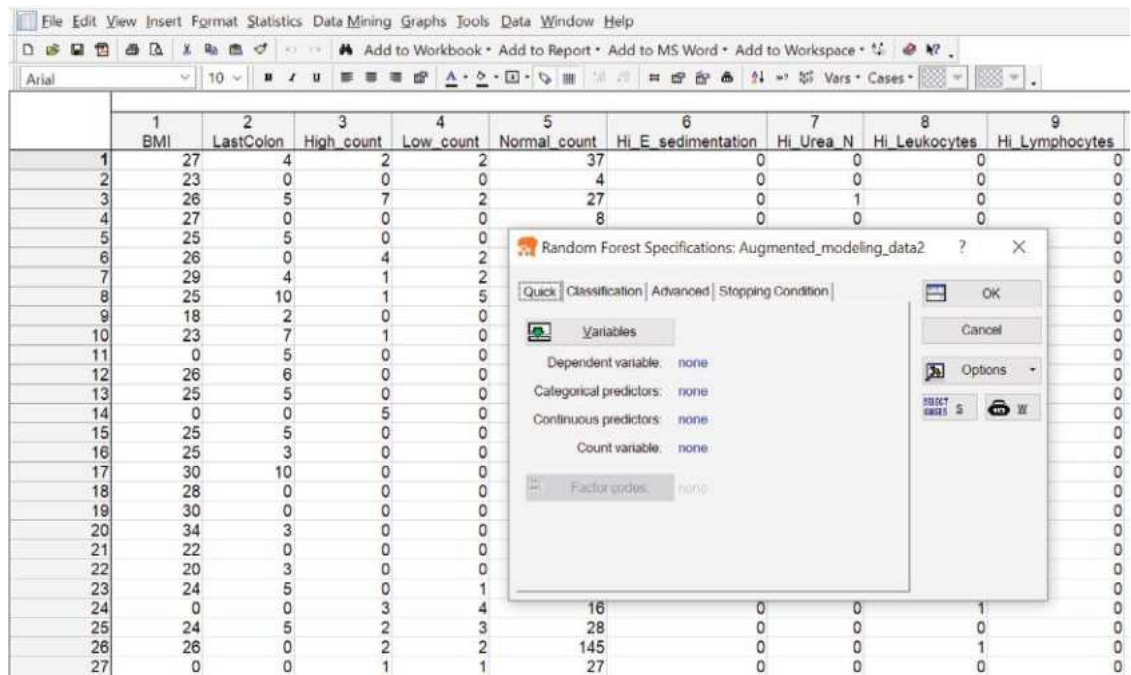


FIG. 4.4 The Random Forest options selection screen in SDM.

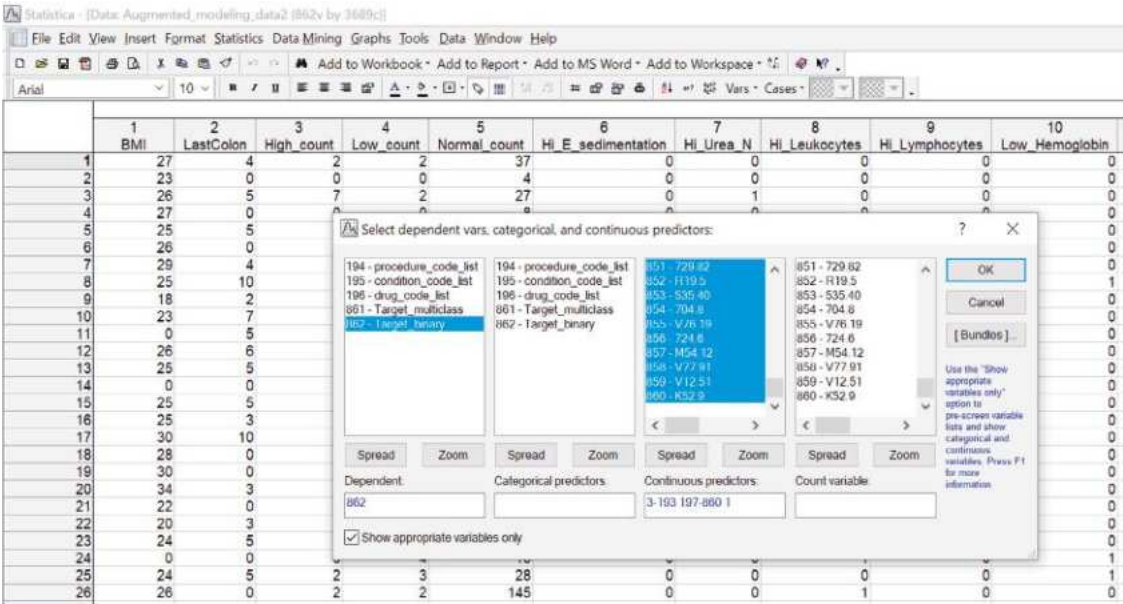


FIG. 4.5 SDM variable selection screen.

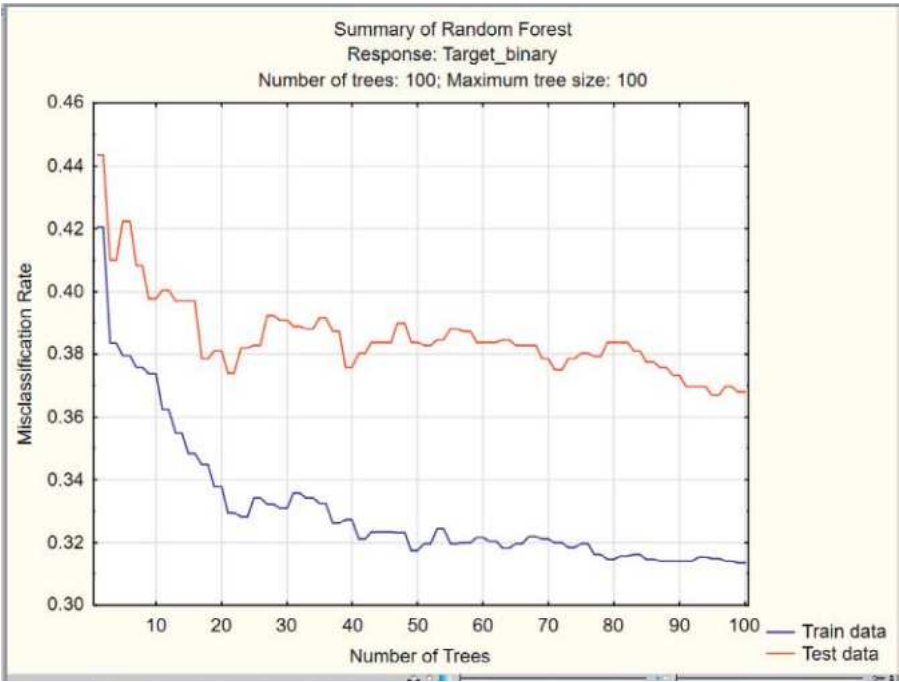


FIG. 4.6 Misclassification error in construction of 100 trees in a Random Forest modeling operation.

Multipersona data science and machine learning (DSML) platforms with graphical programming canvasses and AutoML capabilities

Gartner (2023) defines this class of analytics platforms as: “a cohesive and composable portfolio of products and capabilities, offering augmented and

automated support to a diversity of user types and their collaboration.”

- 1. KNIME
- 2. IBM SPSS Modeler
- 3. Alteryx
- 4. RapidMiner
- 5. DataRobot
- 6. SAS Enterprise Miner

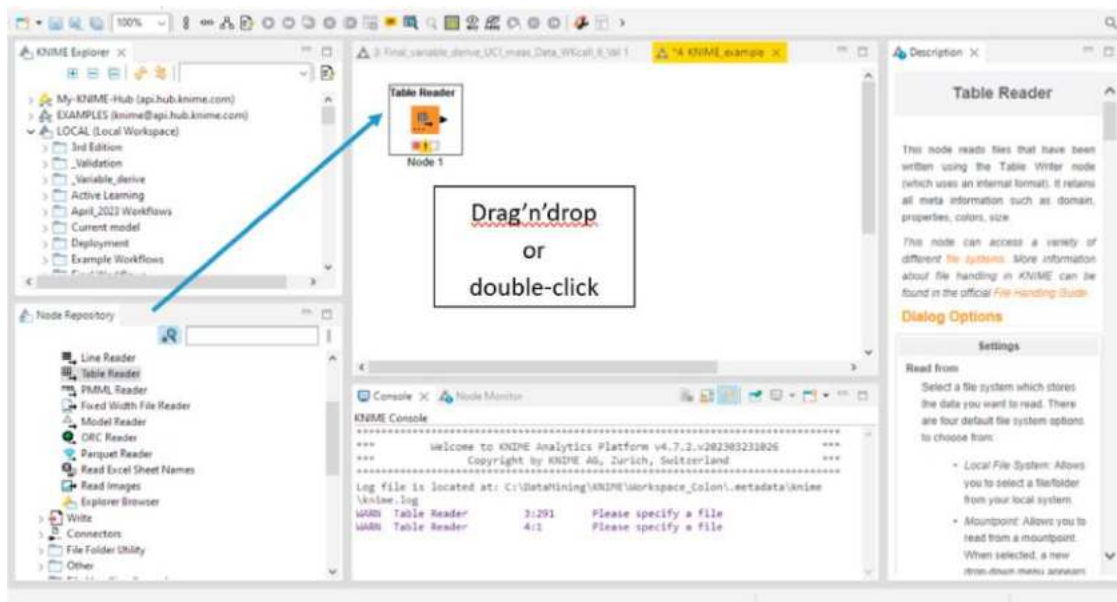


FIG. 4.7 Start of a modeling workflow explaining how to move an operational node (Table Reader) from the tool palette to the modeling canvas by drag'n'drop or by double-clicking on the node.

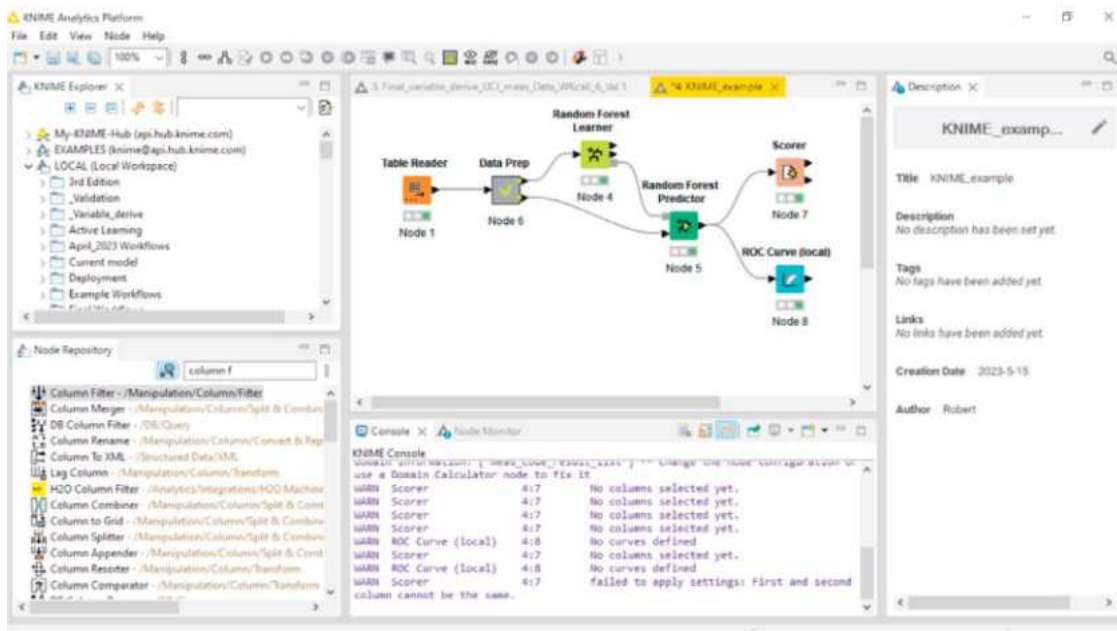


FIG. 4.8 KNIME workflow to train a Random Forest model to predict precancerous colon polyps.

Example of a DSML platform – KNIME

Fig. 4.7 shows the beginning of the building of workflow with a table reader node.

Fig. 4.8 shows the completed modeling canvas to build a Random Forest model with a colon cancer data set.

Output

Fig. 4.9 shows the accuracy table and the ROC curve to report the results of the trained model.

Code-based integration in menu-based and multipersona analytics platforms

Some tools provide APIs to integrate SAS, R, or Python into a platform with a menu or graphical programming interface. These tools combine the benefits of a user-friendly menu interface with the power and flexibility of a code base. As such, these tools can function as front ends and back ends to the procedural language operations.

Row ID	I TruePo...	I FalsePo...	I TrueNe...	I FalseN...	D Recall	D Precision	D Sensitivity	D Specificity	D F-meas...	D Accuracy	D Cohen'...
1	346	159	358	171	0.669	0.685	0.669	0.692	0.677	?	?
0	358	171	346	159	0.692	0.677	0.692	0.669	0.685	?	?

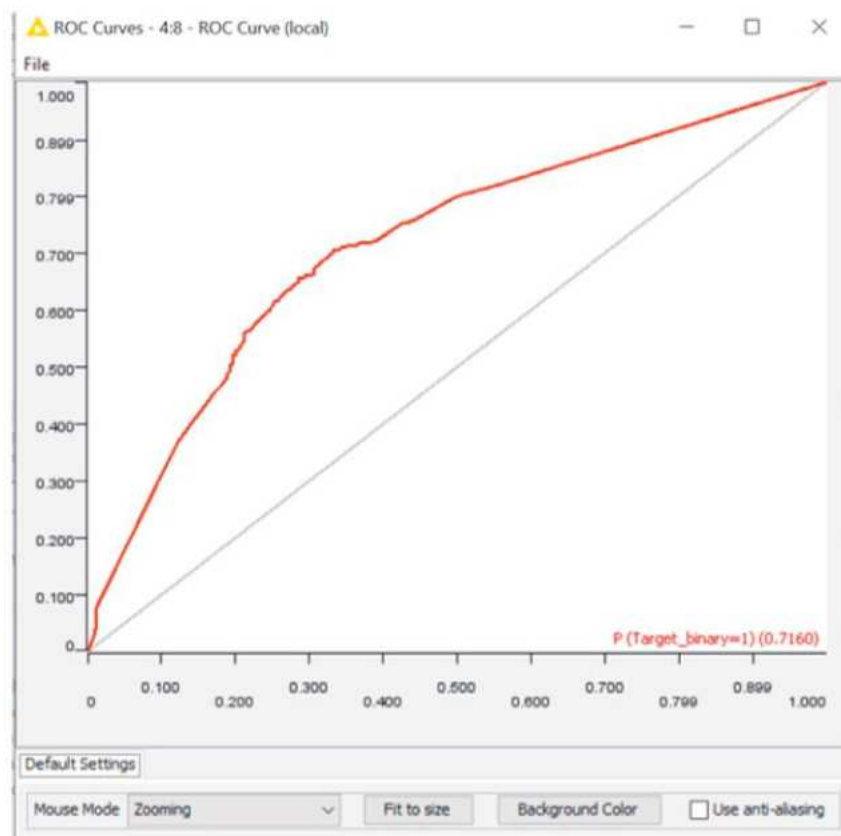


FIG. 4.9 Accuracy table and ROC curve (with the AUC value as 0.7160 shown in red at the bottom right of the figure).

1. Altair: R, SAS, Python integration and code generation
2. Tibco Statistica Data Miner: R integration
3. R Studio: R integration
4. KNIME: R and Python integration

Advice on tool selection and professional development

As we've discussed, Python is a dominant player in DSML. Some familiarity with it is recommended, but that is not the same as saying that all data analytics is performed in Python. When asked "Which tools were used within the last year?" by the 2020 Rexer Analytics Data Science Survey, use of Python or R was nearly 75%, with corporate analysts preferring Python and academics preferring R. However, when forced to pick "one tool" that was used "most frequently," rates dropped dramatically to just 23% for corporate use of Python and

41% for academic use of R. More than one-third of these same respondents list some menu-based or low-code tools among the tools that they use. By implication, analytics are using multiple tools concurrently and learning several tools during their career.

Some organizations will try to enforce a one-tool policy, and it drives their hiring. A cottage industry has arisen to prepare for corporate coding exams. While this may be deemed necessary for software development, it is rarely successful at identifying talented analysts. Professional development time is better spent working on real-world projects and not concentrating solely on the logistics of using a single tool. A career of any length is going to force a lifelong learning approach to tool use. Even if one attempted to restrict work to a single tool, at a minimum, one must navigate constant evolution of features and updates.

One argument that is made in favor of organization-wide adoption of a single coding language by the entire team is that it facilitates collaboration and deployment.

Unfortunately, it can have the opposite effect on collaboration. While it may be true that a small team whose members all use the same tool can collaborate easily, it can cause that same team to be closed off from other teams in the organization. Poorly documented code in any language will be difficult for management, subject matter experts, and IT professionals to understand. Collaboration with these adjacent specialties is also important. Since low-no-code options are nearly self-documenting they will often facilitate cross-team collaboration, not hinder it. So a full organizational view should be taken when deciding the best set of options for the organization. Also, code developed during the modeling phase never survives deployment unchanged – there is always some substantial modification. Therefore in practice, it is quite common to switch tools from model development to model deployment. Tool vendors anticipate this and it need not be a barrier. Single-tool adoption is rare in practice, and herculean efforts to enforce it are unnecessary.

We urge aspiring data scientists to learn multiple tools. We do this for multiple reasons:

- Just as no knowledge of Python may limit one's career prospects in the current climate, knowledge of only Python may also have this effect.
- Any data scientist with decades of experience will be able to relay the frequently changing popularity of languages, platforms, and libraries for performing analysis.
- Knowledge of only one tool limits collaboration – while most data scientists currently learn Python or R, their colleagues in industry and academia often do code at all

Postscript

This chapter ends Part I – Introduction to data analytics concepts and tools. It is necessary to have at least an understanding of the concepts and tool availability before embarking on the “grunt work” of building an analytical model. Part II contains eight chapters that cover various aspects of data preparation.

Key words and phrases

SAS, KNIME, Statistica, Integration, Exploration, Graphical programming, Data integration, Data visualization, Menu interface, Data manipulation, SQL generation.

Reference

Gartner (2023) Multipersona data science and machine learning platforms. Accessed 09/02/2023. <https://www.gartner.com/reviews/market/multipersona-data-science-and-machine-learning-platforms>

Data preparation

This part of the third edition includes [Chapters 5 to 12](#), and it part represents a major addition to the second edition. Topics of data preparation were scattered through many chapters, although the focus of data preparation examples was on customer relationship management (CRM) modeling. We will maintain that focus in the third edition, but the presentation of various data preparation operations will be much better organized to permit data science practitioners to cross apply those operations to prepare data properly for other types of modeling tasks.

In this age of automation, there is a great tendency among data scientists to move as quickly as possible into the analytical aspects of modeling. That movement is almost always a big mistake. The reason is that input data sets are almost never in the proper shape, data condition, and logical constitution to generate the optimum analytical solution. [Chapters 5 to 12](#) describe operations that change the shape, condition, and logical constitution necessary to build the optimum model.

Shape changes

Physical shape

The shape of a data set describes how the data values are expressed. Before the analyst can evaluate any aspect of a data set, it has to be accessed from possibly a number of different sources and integrated into the proper order of data elements (aka fields or variables) for each entity to be modeled (e.g., customer). The customer-centric organization of data is called the Customer Analytic Record (CAR). [Chapter 5](#) describes this process of data access and formation into the physical shape of the CAR. [Chapter 6](#) describes processes to guide the analyst in understanding the nature of the CAR data set before additional operations can be planned for transforming the CAR, one of which may require changes in the mathematical shape of the data set. [Chapter 7](#) shows various data visualization techniques that help the analyst to understand the data set prior to making additional changes.

Mathematical shape

In any analytical operation (particularly parametric statistical operations), input data elements (aka variables) may have widely different scales. In many cases, the range of the scale of a variable affects its mathematical effect in calculating the solution. The operation of “standardization” causes the same numerical range to be imposed on all variables, with individual values proportionately scaled along the common range in the same relationship they had in the original range. Another shape transition modifies the actual shape of the data plot of variable values along their range to be more compatible with assumptions of the algorithm used to build a model. For example, parametric statistical algorithms (like linear regression) assume that the data distributions of all variables are bell-shaped (aka “Normal”). Transformation of the original data distribution to appear more normal in its data plot changes the shape of the data set from a mathematical perspective. Another shape change is performed by recoding, or “mapping”, original data values according to some mathematical transformation function to permit the modeling algorithm (e.g., a support vector machine) to define differences in data patterns more easily for a classification operation.

Data conditioning

Chapters 8 and 9 describe techniques for cleaning extraneous or inappropriate data values and performing other changes in data shape to clearly define the information signal in the data set that the modeling algorithm will use to perform the intended operation of classification or numerical prediction. Chapter 9 focuses on one of the most important data conditioning operations necessary to support classification modeling operations – balancing. In a classification operation, the target variable contains a category code that the algorithm will predict. The mathematics of classification will be biased toward the category that has the greatest proportion of records in the data set. A balancing operation must be performed to render the number of records for all target classes to be the same. This can be accomplished by increasing the number of records (by duplication or synthesis) for the relative rare target values to equal the number of records of the common case. Or, balancing can be accomplished by reducing the number of cases of the common case(s) to equal the rarest case. This balancing operation can make a huge difference in the predictability of the model when the relative numbers of common case and rare case records are vastly different.

Logical constitution

Chapters 10 and 11 describe the feature engineering and selection operations that may be the most important aspects of a modeling project. Chapter 10 shows that these features (transformed variables) are very important additions to the original variables for building analytical models. In many analytical models, several of the top 10 important predictors are those that the analyst derives from original variables. It is a grave mistake for the analyst to skimp on the effort devoted to this activity. Without these additional features, the analyst cannot present to the modeling algorithm some important aspects of information in the data set necessary to build the optimum model. Chapter 11 focuses on feature selection to develop the “short list” of features to be presented to the modeling algorithm. The need for this short list is related to the principle of Occam’s razor, described in the chapter. The best model will be formed by the list of variables that are “necessary” to define the solution; other variables just contribute more noise than signal in the information they provide.

Chapter 12 summarizes all of the data preparation operations described in Chapters 5 to 11 in the form of a data preparation “cookbook.” Analysts can use this cookbook as a recipe for planning their modeling projects.

Data access

Preamble

This chapter describes how stored business data are accessed and transformed into a format suitable for analytical operations. These operations can be performed manually immediately prior to the analysis, or they can be accessed from a special database optimized for support analytic operations – the analytical data mart. This chapter also describes the building of the structure of the data mart and the extraction, transformation, and loading (ETL) operations to fill the analytical data mart.

Knowledge from data: predictive analytics

We are in the midst of a revolution in the use of predictive analytics (PA) in our society. All organizations, public and private, use PA to meet their goals quicker, cheaper, or faster. Most medium- to large-sized companies have built analytics teams to help them increase profitability. Even politicians use PA to get votes. President Obama set up a team of analysts in Chicago to assist him in his 2012 reelection campaign. They targeted the friends (influencers) of people tagged as “swing voters” who could be convinced to vote for Obama. It worked! One in five people targeted acted to try to influence their friends to vote for Obama (Sherner, 2012). There may have been 5 million more young American voters in 2012 than in 2008, but the point is that Obama’s appeal driven by PA captured their votes.

Overall structure of the database – data modeling

Data scientists focus on gaining knowledge from data by building analytical models to predict things. IT professionals describe the subject of structuring the data in the database as *data modeling*. You should not confuse the two terms. Analytical models are not data models. Data models are structural in nature, while analytical models are mathematical in nature.

Account-centric format of data in early computer databases

Most computer systems in businesses were developed initially to serve accounting operations, where the account number was the primary organizing entity. Later, other database systems were developed to hold product information as the primary element of organization. These account-centric or product-centric database systems worked well for storage, limited manipulation, and retrieval of information to serve billing systems primarily. Customers were associated with the accounts, but the focus was not on them originally; it was upon the accounts that the customers owned. Only in the last 20 years has the focus of greatest interest shifted from account-centric to customer-centric systems.

IT data analysts are used to working in data environments that employ random access data methods; consequently, the order of fields in a table doesn’t matter. However, some modeling tools remember the metadata (field names, properties, and order) that was input during the previous round of modeling. A saved modeling stream in the modeling tool may assume that a given metadata describes the data stream input to it when the metadata is actually different. That incompatibility may cause the modeling application to crash! An example is the IBM SPSS Modeler AutoDataPrep node and the KNIME Domain Calculator node. Another reason for modeling plural analysts to be involved (directly or indirectly) in the ETL activities is that they know that he cannot know everything at the beginning that he may need to know. After some preliminary data description or analysis operations, it may become clear that the analyst must go back to a point partway through the ETL operations stream and make some changes (e.g., use a different series of codes in the aggregation process or bring through the process additional data elements). Therefore the modeling analyst should shepherd the IT operations and do as many of them as possible.

Data access from account-centric data sources

The access to and extraction of data from a variety of data sources for use in analytical models pose several challenges for the analyst. The first challenge we face is to extract data from available source systems. General extract (E), transform (T), and loading (L) operations are used to accomplish this task. These operations are discussed in the ETL Process section that follows. These operations can be performed by SQL scripts run directly or via online database connection (ODBC) drivers to extract data from the source systems to the analytical tool used. Most analytical tools have several ODBC drivers available for connection to many common database source systems. The extracted data is usually account-centric in organization; that is, the account number serves as the key to relating information to various tables of the database.

The second challenge we face is to transform the account-centric data to a format conducive to analysis, with the central focus on customers, not accounts. Most business databases or data warehouses hold transaction data used to serve the billing system for individual accounts; that is, they are account centric. This transformation can be done in SQL during data extraction or performed by a business intelligence (BI) tool like Informatica. Some analytical tools have sufficient transformation capabilities to perform necessary operations. Often, however, the complexity and long execution time to perform these transformations can be awkward and time consuming to perform during the data extraction or BI operations. An alternative approach is to do much of the “heavy lifting” in the processing at night to fill a series of tables from which data can be extracted directly into the modeling tool. Such a separate data structure is called an *analytical data mart*.

The customer-centric analytical data mart

A data mart is a subset of a data warehouse that is focused on a particular part of the company’s operations (e.g., sales). Data in the data mart are (ideally) refreshed

each night by a series of SQL scripts. For example, a data mart could be built in a banking operation in which all banking data are contained in a relational or dimensional data warehouse (Fig. 5.1).

The importance of IT personnel in the modeling effort

An IT team member should sit in on many sessions of analytic tools training classes. The predictive analyst should sit in on an IT team meeting when an analytics project is being discussed so that appropriate questions can be answered. There are instances when the IT team should do some of the ETL. A good example is a big transposition operation in which the history of different coding schemes must be understood in order to perform this task properly.

What caused the shift to customer-centric data organization?

As information piled up in computer system databases, sales and marketing personnel wondered how this information could be used to help them sell more products and services. They turned to their IT department to provide information on the number of items sold or services provided, organized by geographic region. This information was used to track sales performances in different geographical regions of the company’s operations. These operations were given the name “Business Intelligence” (BI). The term business intelligence was coined by Luhn (1958) as “... the ability to apprehend the interrelationships of presented facts in such a way as to guide action towards a desired goal.” But, it was not until 1989 that the term was adopted as an “umbrella” to describe the combination of concepts and methods of using stored information to improve decision making (Power, 2007). The 1990s saw an explosion in the development of data warehouses in an effort to present “one version of the truth” across all parts of the company. For example, this mantra was preached

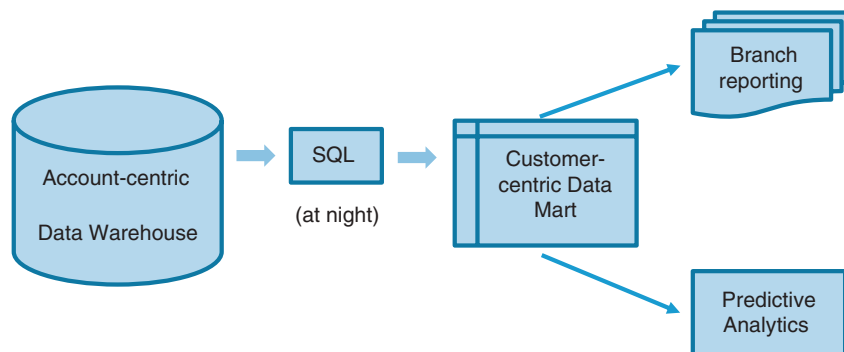


FIG. 5.1 A Customer-centric data mart in a bank, which serves both branch reporting and analytical operations.

very strongly by NCR Corporation Teradata data warehousing salesmen. Sales and marketers began thinking about how they could use BI reports of these “truths” to “target” specific geographical regions (markets) with different assortment of products and services. Gradually, this focus narrowed to the concept of “a market of one,” namely, the individual customer. In the mid-1990s, this heightened attention on the customer resulted in the application of analytical technology to find patterns of buying propensities of individual customers – the age of data mining was born.

Why is customer-centric data needed in analytical operations?

BI reports from the older account-centric systems in traditional data warehouses were focused on how well products and services were sold and vended. However, the rising interest in customer buying patterns required that data structures be reorganized on the basis of the customer, not the products and services they bought. This change required the analyst to extract data from a variety of account-centric tables and reorganize them on a customer basis. This means all information pertaining to a given customer had to be related to a common database key (the basis of information lookup), the customer number. One way to conceive of this organization is to consider the gathering of all of the information pertaining to a given customer into one row of an Excel spreadsheet in which the first data element (column in the spreadsheet) is the customer number, and successive columns records information related to products bought and other customer-related data (e.g., demographic information). The Excel spreadsheet form of data organization is exactly the format needed by analytical tools.

Previously, many different products or services could be purchased by the same customer, but the organization of the database made it very awkward and cumbersome to gather all of the information on the purchases by a given customer. In a customer-centric database, the product or service key was no longer the basis of information lookup; the customer number became the central organizing theme of the new database, and it became the new basis of lookup. This database existed in either the form of a flat file (analogous to an Excel spreadsheet) assembled (integrated) by the data analyst from many data elements extracted from account-centric database table. Alternatively, the data could be extracted to an analytical data mart. These data marts were developed initially on separate computer systems but eventually, some were implemented in “virtual data marts” on the main data warehouse system in the form of database “views.” A database view is a series of stored database

queries that access data elements in specific database tables and loads them into a temporary data structure in computer memory. This temporary data structure can be queried as if it were a table of data in the database. This is the form of the analytical data mart in the banking operations represented in Fig. 5.1.

The ETL process

This process consists of extraction of appropriate data elements from source systems and integrating them into a common physical or logical data structure. Data elements may have a variety of characteristics: different storage forms, different data formats, different levels of aggregation, different time-grains, and following different coding systems. Data elements from all sources must be conformed to the same data format, transformed to a common level of aggregation and time-grain, and converted to a common coding system. Only then will numbers be commensurable (suitable for mathematical operations) and categories be comparable. The final step is to build an ETL operation that produces consistent data for the analysis of customer-related questions; this is analytical ETL.

Analytical ETL: What is it and how does it function?

ETL technology was developed initially for the building of data warehouses. Data from internal and external data sources passes through three distinct data processing phases to the data warehouse. Fig. 5.2 shows these phases: (1) data extraction (E), (2) data transformation (T), and (3) data loading (L). Each of these phases consists of a body of operations in SQL or via some SQL code generator (i.e., the Informatica ETL tool).

Consider the case where much of the historical customer data is aggregated and stored in the enterprise data warehouse (EDW). Detailed information (e.g., billing data), however, is not uploaded to the EDW but is contained on the accounting/transactional computer system. Information from the accounting/transactional system is summarized and loaded onto the EDW, but the detailed transaction information resides only on the accounting/transactional system. A good example is call-level detail data in telecommunications systems. Some of the most powerful predictors in a model may be derived from transactional data elements in the switch system. Analysis of the ratios of incoming calls to outgoing calls in the call-level detail data collected at the switch level can be used to predict which customers have a home office. Also, data from marketing activities might exist only in Excel spreadsheet format.

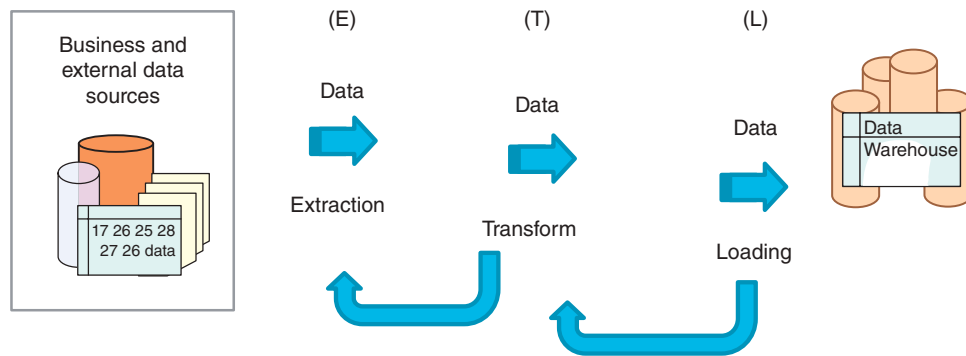


FIG. 5.2 The flow of ETL operations to install data from external data sources into the data warehouse.

Finally, other demographic data on customers might be available only from outside sources, such as Experian. The job of the analyst is to determine where the data elements reside, gain access to the information, and install it into a single customer record. This job is far more difficult than it appears. One or more of the data sources may be so poorly documented with a data dictionary that it is awkward to determine what data elements are present and what their codes mean. A common mistake is to short-cut this difficult process – the temptation is to use only what is readily available or what can be easily integrated. The thought process is that the result would represent a “quick win” or enable harvest of “low-hanging fruit.” The problem with that approach is that the model is likely to be weaker than it might be and would require data preparation operations twice (again after preliminary modeling). This approach is ultimately less efficient than doing a thorough job the first time.

Given the knowledge of what is available from a given data source, gaining permission for access to the information might be awkward and time consuming to obtain. With data extracted from all of the data sources by SQL scripts, it remains for the analyst to put them together (integrate them) to form a single customer analytic record (the “CAR”). Furthermore, some data might require aggregation or recoding to make it consistent with the final aggregation level or time-grain designed for the CAR. These activities may consume up to half of the time allotted to a PA project.

Analytical ETL activities should be done by the data analyst, because he understands much better than the IT SQL expert how the information will be used. This understanding is necessary, because unforeseen problems and glitches with extraction and transformation operations arise inevitably and must be solved along the way; knowledge of the codes in each field is crucial to doing the job right, and the data analyst is the best person to do it.

There are some operations that might be done more efficiently by the IT SQL expert, albeit in collaboration

with the analyst. Sometimes, a very large number of transactions are done by a smaller number of customers. For instance, imagine that 5 million customers engaged in 120 transactions. At the same time, the analyst may recognize that the modeling phase will require some assistance from IT. For example, the IT expert will also know how to schedule time and resource allocations efficiently for a computationally intensive operation like a transpose or merge of a large number of records. Once the resulting 5 million row flat file is created, the analyst can probably take the lead from there.

Analytical data flow

The processing sequence to move data from internal and external sources to either an external flat file or a series of tables in the Exploration Data Mart (an analytical data mart) is included in Fig. 5.2.

The data flow pathway shown in Fig. 5.2 appears to be a linear cascading process; it is not – there are loop-backs and feedbacks along the way, reminiscent of the feedback loops in the CRISP-DM analytics process shown in Fig. 3.2. For example, the transformation operation for one data element may suggest other data elements which should be extracted. Each of the three major ETL phases can be decomposed into several tasks, as shown in Figs. 5.3–5.5.

Analytical ETL – data extraction operations (E)

Fig. 5.3 shows the three major data extraction activities necessary prior to analytical operations.

Data extraction – data access

Data may be located in a variety of formats in a variety of systems in a variety of locations inside and outside the company. Getting access to these data sources is often awkward and difficult, presenting a number of challenges for the data analyst to meet.

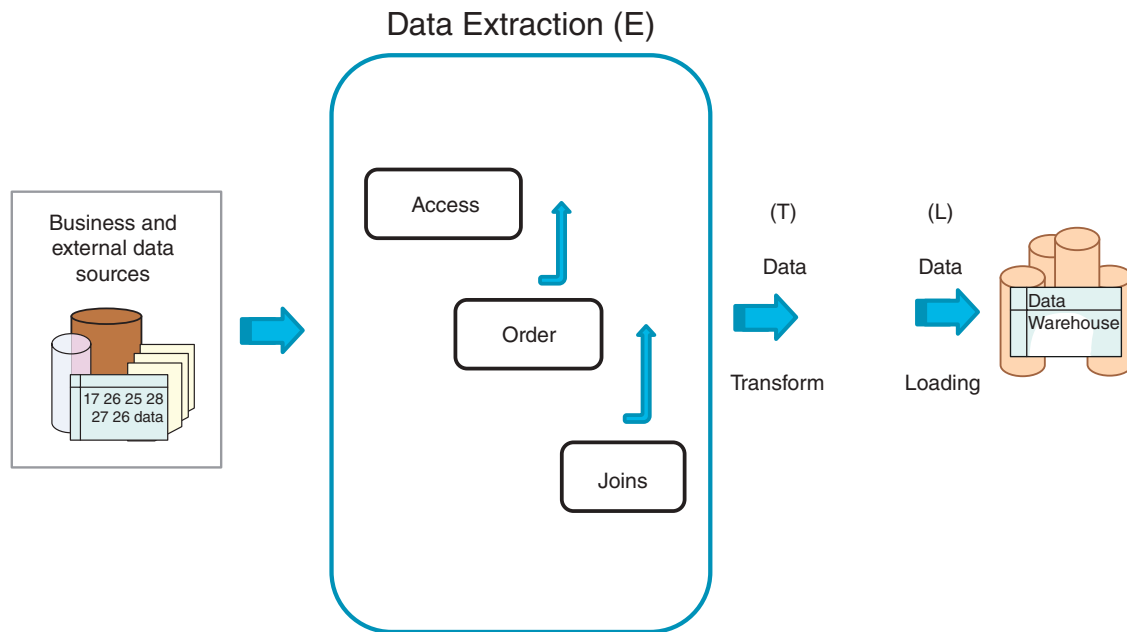


FIG. 5.3 Data extraction operations in analytical ETL.

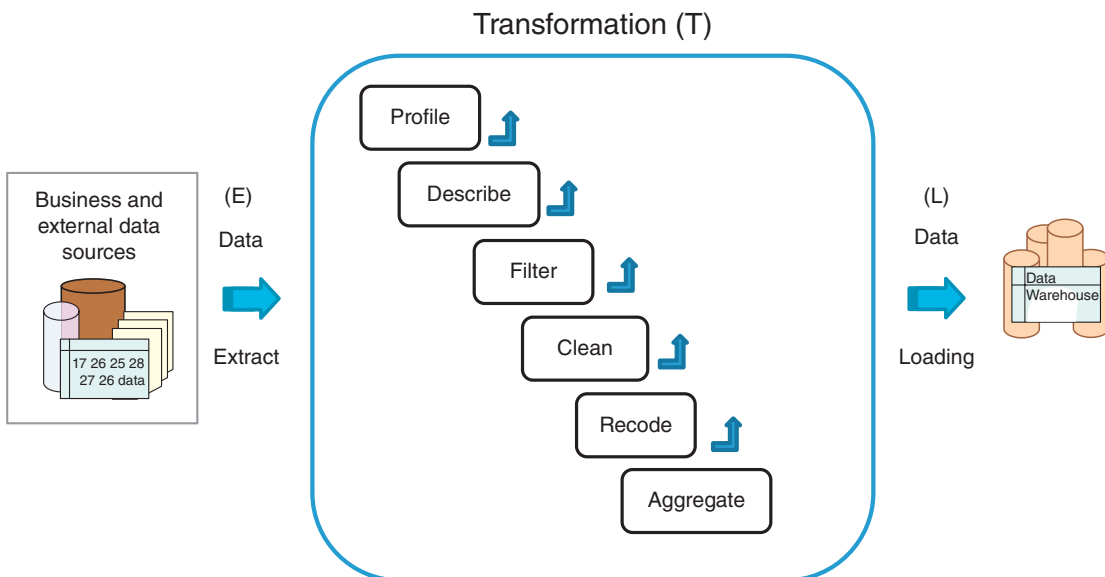


FIG. 5.4 Data transformation operations in analytical ETL.

Data inventory

The initial challenge for the analyst is to inventory these data sources. These sources might include:

- a. *Spreadsheets.* Each department of the company hosts a slice of the company's operations and may store, in electronic or paper spreadsheets, the results of internal analyses and management programs. This internal department data can be very valuable to the analysis. For example, the Sales department may store sales performances by

region (used for setting performance incentives), which can be mined to derive powerful predictive variables. Even if the data in the spreadsheets need not be incorporated into a larger data set, examining the tools used to manage processes can be very insightful for setting up the data set. For instance, what if a certain series of sales thresholds triggers higher commissions? The model may reveal these thresholds as being important, but without this knowledge, it would be difficult for the modeler to capitalize on them.

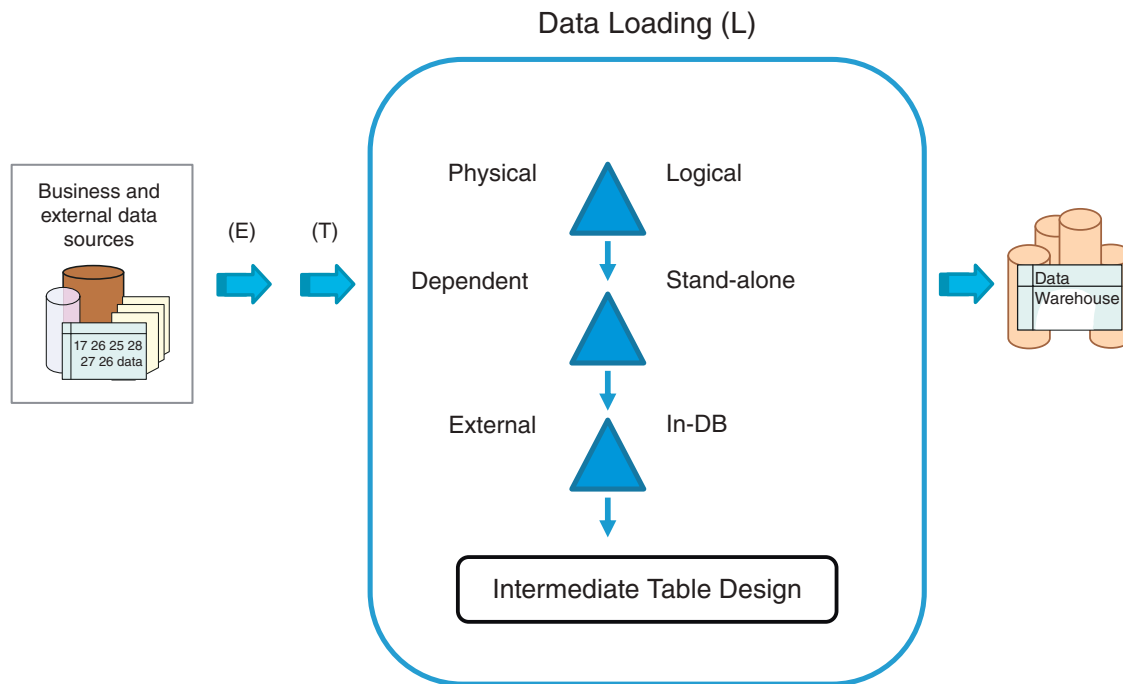


FIG. 5.5 Data loading operations in analytical ETL.

- b. *Databases.* Each department might have a data mart used to process summarized information to upload to the data warehouse. Commonly, raw data consists of detailed transactional data in Online Transactional Processing (OLTP) systems, such as retail sales databases such as Salesforce.com installations. The detailed data elements could be very valuable for modeling customer purchase behavior. If a company has a data warehouse, only summary data are extracted from the detailed transactional systems and loaded into warehouse tables. The problem with extraction of this summarized data for analytical purposes is that the summarization may obscure detailed information that can be valuable for modeling purposes. The more detailed point-of-sale data should be at least examined to investigate what might be lost at higher levels of aggression. IT teams may be surprised when analysts request these detailed data, but due diligence concerns require examining these data, even if they are never incorporated directly into the analysis.
- c. *Flat files.* Sometimes Excel spreadsheets are saved in comma-separated-value (CSV) flat files, because they consume far less storage memory than do Excel files. Many other applications can output CSV files, rendering this file type as a common means of accessing data processed in many other applications. Almost all model-building applications can input CSV files directly.
- d. *Text files.* The subject of text mining deserves a book by itself, and many are available (e.g., [Miner et al., 2012](#)). Textual information in an insurance underwriter's notes, for example, can be a valuable source of information related to insurance risk that is not uploaded to the data warehouse. For example, a simple word count of the term "inadequate" can be used to derive a predictor variable for modeling. In anticipation of a more complete analysis later, it can be valuable to simply mark the presence or absence of information like underwriter's notes as a binary variable. If that proves useful in the model, it helps to signal that a return to the raw text is likely to be valuable.
- e. *External data.* The most common source of external data is available from vendors like Experian and Dow Jones, providing valuable demographic and firmographic information that can be related to customer behavior patterns (e.g., what is purchased and when it is purchased). Beware of "off-the-shelf" modeling solutions that bet the success of the project on only these variables. By using standard external variables, the project variables are therefore useful for a "one-size-fits-all model." While it might be better than no model at all, it can't compete with a model that uses these variables in combination with the others.

Data access permission

The next challenge is gaining permission to access data from each source. This access is facilitated by granting a

specific user ID and password information to gain access to the specific data structures. This challenge can take a *long* time to complete, because each data source has its own bureaucracy and security infrastructures that must be negotiated before access can be granted. Expect these phases of the project to take more “calendar time” than “clock time,” in that the analyst may be idle awaiting this access. In some cases, medical or governmental privacy regulations may require the data to be “anonymized” (or “deidentified”) by removing all names, addresses, SSNs, and any information that could link the record to a specific person. This situation could occur when data is sent to a third-party vendor of data processing services. The customer ID number may be the only data element that could link the case to a specific customer, but that could happen only inside the company. An additional layer of security could be provided by using a sequential number as a customer ID, which could be mapped back to the internal customer ID via a mapping table stored only on a company computer system.

GDPR compliance

The new General Data Protection Regulation (GDPR) went into effect in the European Union (EU) on May 25, 2018. The primary elements of this new regulation relating to the use of personal data for PA are:

1. Individuals must give permission for personal data to be stored.
2. Individuals have the right to have their previously stored data deleted from all company storage structures.
3. Individuals have the right to be informed of the exact processes and decision rules driven by their stored data.

Stipulation 2 above means that some historical data for some individuals may be incomplete. Data analysts must be aware of that possibility when performing data preparation operations. Stipulation 3 above means that some modeling algorithms (e.g., neural networks) which function as “black boxes” cannot be used to build models on which decisions involving the individual are based. Decision tree models and many modeling tool sets do not violate this regulation of GDPR, because it is relatively easy to infer business rules from the decision trees. For more information, see: <https://www.forbes.com/sites/forbestechcouncil/2018/06/04/gdpr-the-checklist-for-compliance/#1c3df685bec7>.

While the GDPR regulations do not apply to data processing of individuals within the United States (at this time), it does apply to individuals residing in the EU. Therefore a product or service intended to be marketed outside the United States should be GDPR compliant.

ODBC configuration

The third challenge faced by the analyst is the need to “ream out the pathway” between all source system data structures and the destination data structure via the configuration of an appropriate ODBC driver. In addition to user ID and password provisions, any required details of the specification and configuration of the ODBC driver must be communicated by IT personnel to the PA specialist. The authority to provide this information must be granted by the IT department before it can be given to the PA specialist, and this process can be time consuming, depending on the priorities and workload volume controlling IT response time.

Data extraction – order

Record order

If data elements can be joined via a join key, order of the records in each data source extract is not important. The order of records from some data sources (e.g., electronic spreadsheets) may be important if the ODBC driver cannot treat the spreadsheet as a database (e.g., relative addressing is employed in the spreadsheet). Order of records also may be important if the intended integration technique is merging rather than joining. Joining requires a join key, which serves as an identifier of records with common elements to be integrated into the same output record. A merge operation does not use a join key, but rather, it assumes that records in both inputs are sorted in the same order, and the data elements (field) of each successive row in both sources are present in both data sources by the same name and the same order. The control of record order before integration may appear unnecessary or trivial, but it can corrupt the output data by integrating elements of rows with different customer IDs into the same output data record. Sometimes this error may not be apparent until some time later, much to the analyst’s dismay. Some tools (e.g., KNIME and IBM SPSS Modeler) use the same operations nodes to perform both merging and joining tasks.

Aggregation plan

The analytical modeler of banking data might want to work at the week level, but the data is stored at the transaction level associated with a time stamp. For example, the number of credits to an account must be aggregated to the week level with SQL (usually) and stored in temporary data structures. In a retail store, some customer data might be stored as transactions with time stamps, and some data might be stored in billing systems at the weekly or monthly level. Therefore the modeler must aggregate data at various time-grains and install the combined data in a data structure of a given time-grain. Some of the aggregation might be

done directly in the SQL data extraction scripts, and some of the aggregation might be in the modeling tool. An aggregation plan should be created to manage these diverse inputs and operations to ensure that all of the data elements are extracted and processed appropriately to serve the modeler purpose. You may even need to experiment with different levels of aggregation. Perhaps each line item in an order should be the level of analysis, and perhaps it should be analyzed at the order level.

Data extraction – joins

For the purposes of analytical modeling, joining is included under data extraction rather than data transformation, even though some transformation operations (e.g., aggregations) must be performed before joining occurs. In the analytical context, data transformation operations occur primarily after the various sources of data are joined to form an analytical record of elements (data fields) associated with a single entity (e.g., the customer).

With that thought in mind, you might wonder why the analytics technician and not the IT technician should be the one who performs the joining operations. Reasons why the analytics technician should join operations include:

1. The appropriate order of fields must be specified by the person who will do the modeling, because there are many steps in modeling that require variable selection. Similar groups of variables (from a modeling perspective) make it easier to select variables. For example, variable selection in some modeling tools separates variables with decimal numbers (continuous variables) from categorical variables. Output tables of intermediate modeling operations list variables in the order in which they occur in the analytical data structure. If the categorical variables are all grouped together, it is easier for the analyst to compare them.
2. The analytics technician understands how the modeling algorithm will “view” data in each data field in the analytical record as either descriptive and demographics data or candidate predictor variables.

Data integration

Consider two data sources with some data elements (fields) in common and some that are different. The Venn diagram (Fig. 5.6) represents the two data sources as two ellipses. The area of the left ellipse labeled as “A” represents the records in the first file that are not represented in the second file. The area labeled as “B” represents the group of fields that are not represented in the first file.

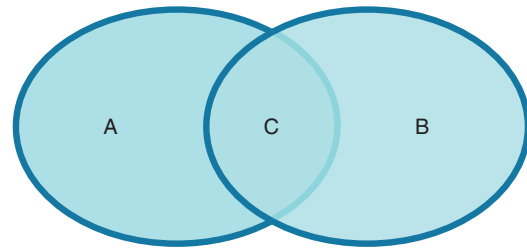


FIG. 5.6 Venn diagram showing three logical groupings of records in two files (ellipses).

The area of overlap between the ellipses (labeled as “C”) represents the group of fields contained in both files.

There are two general approaches to integrating data from two different sources.

1. Merge – Combine all of the fields in the first row (record) of the first data source with all of the fields of the first record of the second data source. Do the same thing with all other records in sequence.
 - a. The merge operation does not check to see if the rows combined from each data source are compatible. This means that there is no “join key” in the merge operation, which establishes a commonality between the groups of fields in the two data sources.
 - b. The records are merged according to the order of them in each of the two data sources.
2. Join – Combine some or all of the fields of matching records from two data sources. The common identity of records in each data source is evaluated according to a join key. This key consists of one or several fields in common among both data sources. The data rows need not be sorted in the order of the join key, because the keyed search is a random access operation. There are many types of joins possible, depending on the definition of which set of fields are combined.
 - a. Left outer join – combination of fields in areas A plus C in Fig. 5.6
 - b. Right outer join – combination of fields in areas B plus C in Fig. 5.6
 - c. Inner join – combination of fields in area C only in Fig. 5.6
 - d. Full outer join – combination of fields in areas A plus B plus C in Fig. 5.6
 - e. Antijoin – Selection of fields in A only, and not in B in Fig. 5.6

The inner join, left outer join, and right outer join are called also “partial joins,” because only some of the fields of one or more data sources are included in the output data structure.

The ETL process shown in Fig. 5.2 appears to represent a linear flow of operations from data sources to data

output. But this one-way flow of operations is rare. As mentioned above, it is usual that some transformation (T) operations are mixed in with the data extraction (E) operations. For example, in banking operations, some of the most powerful predictors are derived from low-level transactional detail, such as the total amount and total number of credit transactions during a week. These weekly figures must be derived from daily records, but daily is too fine a time-grain to use for modeling. The numbers and amounts of daily credit and debit transactions might be aggregated to the weekly level for analysis. These loop-back operations are symbolized by the left-pointing curved arrows in Fig. 5.2. This operation can be done in the same SQL program as the joining operation. The combination of extract and transform functions in the same operation illustrates the relative complexity of ETL operations and belies the apparent linear flow of information and processes from source to destination data structures.

Data transformation (T)

The second major ETL activity is data transformation.

Data transformation activities in ETL overlap to some extent the phases of the predictive analytic process. Like Fig. 5.3, Fig. 5.4 infers that data transformation is a linear cascading process. For the most part, this is true, but not entirely. There are loop-backs between each activity, as shown by the up-pointing arrows in Fig. 5.4. For example, changes in the codes in the Recode activity may direct editing of the filter scripts. Some of the recoding tasks might be included in the Data Extraction SWL stream for purposes of convenience.

Data profiling

The first step in transforming data elements in the combined data set is to profile each data element. In the development of data warehouses and BI activities, many data description elements are included in the data profiles of each data element (see Data description below). Some elements of data profiling are more complex than simple descriptive statistics. For example, data trends, data ranges, and data correlations can provide valuable characteristics of specific data elements, which can affect the kinds and extents of transformation operations necessary to be performed. Data elements that are very highly correlated can require that decisions be made to avoid integrating certain data elements. Profiling can also include the amount of missing data.

Data description

We must document the central tendency of the data set (the average or the “mean”), the general form of dispersion (standard deviation), and other measures of the data distribution, before we can consider changing (transforming) any of it. We must document the coding systems of

the data elements before we can fill any missing values. Why fill missing values? We don’t have to do that with standard parametric statistical analysis modeling algorithms, because they all depend on the calculations of the mean and standard deviation. The reason we must fill missing values of data elements we expect to use in PA is that many of the mathematical algorithms require it! We will discuss this topic at greater length later in this book.

Data filtering

Some data is not useful or appropriate for PA operations. For example, training data may be inadvertently left in a table, which is accessed in data extraction operations for PA. This training data should not be used for prediction and model evaluation, because it would set up a tautologous operation. For example, a tautology is formed when a word is defined by the word itself (e.g., “the blue sky is blue”). For the same reason, we cannot use data that participated in training the model in the evaluation of the model. Training data should be excluded from any data used to test the performance of the model.

The next three task groups are treated in greater depth later in the book. They are included here also because it may be most efficient to perform them here rather than wait until the analytical data set is complete.

Data cleaning

Data cleansing during the ETL activity of PA can include filling of missing data for some data elements (most of which will be done later during data preparation of the CAR).

Recoding

Some simple recoding might be done in the SQL scripts. For example, access to banking trust information might require complete anonymity, requiring that certain ID fields be recoded to “unknown.”

Data aggregation

Data in many transactional databases might be stored on a daily basis, and landing all of the data on the modeling platform would be very inefficient in terms of time and storage space. Data extraction scripts could aggregate daily data to create weekly summary data elements for later use in modeling and thus greatly reduce the number of records and storage space required by the output data structure.

Data loading (L)

The third ETL operation is the installation of data into the analytical data structure, which is performed in the loading (L) process in ETL operations. Fig. 5.5 shows three types of choices that can be followed in the loading operation.

Data loading operations include the definition of the nature of output data structure. The elements of this definition include the answers to three questions: (1) will the data mart be physical or logical? (2) will the data mart function as a dependent or stand-alone system? and (3) will PA be conducted on data in tables (located somewhere) or on data in existing data warehouse tables (in-database modeling)?.

Issues in data mart design

Physical versus logical

The first element in the definition is the selection of whether the customer-centric analytical data mart will be physical or logical. A physical data mart is hosted on a separate computer system, physically separate from the data warehouse system. A logical data mart consists of a number of tables or database views (logical tables) in the data warehouse system. These tables or views can be associated with relationships between them to function as a data mart wholly contained within the data warehouse database management system.

Dependent versus stand-alone

The second element of the definition of the output data structure is the selection of whether the data in the analytical data mart are loaded only once during the project or whether data are updated regularly (nightly?).

Analysis outside or inside the database

Some modeling algorithms can function to access data inside the database structure via ODBC or via a call-level interface for faster access. These algorithms can train models without requiring data to be moved from the database to the modeling system. This modeling architecture is called in-database modeling or “in-DB mining.”

Intermediate table design

Once the dimensions of the data mart are defined, the ETL scripts can be configured to point to tables(s) or logical data structures designed to hold the collected and aggregated data to be used for further modeling operations.

Current trends in data access and data structure design

Analytics is a fast-moving field. Without caution, practitioners can become so distracted by contemporary trends that they lose sight of the big picture. There are

four trends in particular that will continue for frame the way data is stored and processed: (1) Big Data, (2) Hadoop, (3) Distributed computing, and (4) Cloud computing.

Big Data

For decades analysts have struggled with computing power keeping up with data volume. So for many veterans of data mining, Big Data seems like the continuation of long-standing trends. For those who build PA models, this is largely true. For those in the IT area whose role is to gather and store these increasingly large volumes of data, it truly does reflect a change in approach and technology. Some of those changes will inevitably change the way modelers perform their tasks.

It is also important to keep in mind the fact that model building requires a substantial amount of data but rarely involves all the data. Model building should be performed only on historical data that is consistent with the data set that will be scored by the model at deployment. Model scoring, on the other hand, sometimes is performed on a single row of data in some deployment strategies. Therefore it is imprudent to embrace too quickly a Big Data strategy before considering the model deployment strategy. The reason is that modeling and deployment are separate events and may be associated with different issues of scalability and data access. Some modelers believe that all of the data for given customer or patient over many years is needed to build the model. In practice, however, modelers should focus on how much data will be needed to build the model; sometimes, that will be only 10 – 20% of the total records available.

Hadoop

The rise of the Hadoop ecosystem concept begs the question, “What implications does this have for data preparation and modeling?” Either data integration must be performed to assemble the data in a single location or the modeler has to know enough about these technologies to model data stored across multiple machines. Modeling algorithms have been modified in some cases to support operation across multiple machines. For instance, we must modify our concept of a regression trend line if the data set is stored on 100 computers. How does one combine the 100 trend lines into a single trend line? Perhaps surprisingly, however, the modeler is somewhat sheltered from these concerns. Many software packages (e.g., KNIME and SPSS Modeler) provide functions for pulling data directly from these systems and in many cases, multiple versions of common modeling techniques are provided – versions that run on the client machine, versions that run

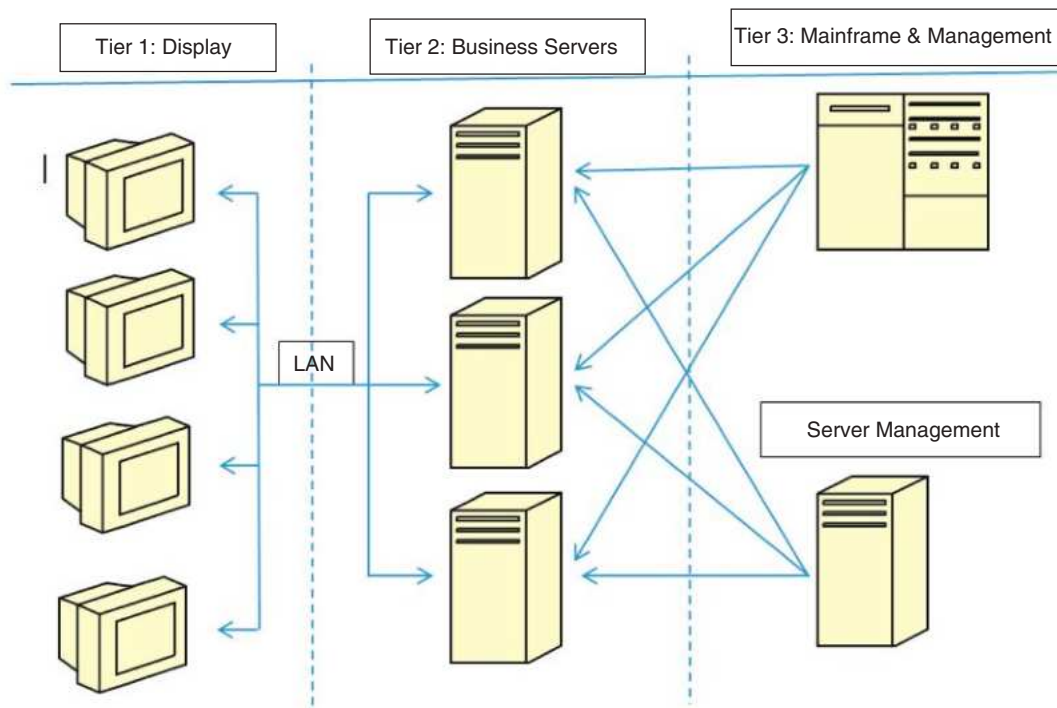


FIG. 5.7 A three-tiered architecture of a distributed computer system.

on database servers, and versions that run on Hadoop. The Hadoop “ecosystem,” including numerous spin-off technologies, involves storing and retrieving large amounts of data distributed across multiple machines.

Distributed computing

Distributed computing consists of running elements of a software system on different computers, possibly remote from each other. All of the component computers are run as a system. This system can be readily expanded by including more computers as needed. Computer redundancy can be built into the system by running the same software components on multiple machines. A distributed computing system can be configured as a three-tiered client-server architecture, as shown in Fig. 5.7.

Cloud computing

Cloud computing is the delivery of computing resources over the Internet. These resources include servers to perform data processing operations remotely. Remote access can also be provided for storage and analytics software. Cloud computing platforms eliminate the need for organizations to buy their own computing resources; they pay for only the resources they use.

The big three providers of cloud computing platforms are IBM, Microsoft Azure, and Google. Many companies have moved their IT operations to the cloud. It may make sense for many companies to move their analytical operations to the cloud also.

Keywords and phrases

ETL, Database, Prediction, Transformation, Joining, Aggregation, Account-centric, Customer-centric, Data mart, Data extraction, Data structure, Data warehouse, Data preparation, SQL scripts.

References

- Luhn, H.P., 1958. “A business intelligence system,” *IBM Journal*, 2(4), 314. <https://doi.org/10.1147/rd.24.0314>
- Miner, G., Delen, D., Elder, J., Fast, A., Hill, T., & Nisbet, R. (2012). *Practical text mining and statistical analysis for non-structured text data applications*. Academic Press, p. 1053.
- Power, D. J. (2007). A Brief History of Decision Support Systems, version 4.0. DSSResources.COM. <http://dssresources.com/history/dsshistory.html>.
- Sherner, M. 2012. How Obama’s data crunchers helped him win. CNN, Nov. 8, 2012 broadcast.

Data Understanding

Prelude

The Data Understanding phase of the CRISP-DM process model is considered as the first step in data preparation, even though the Data Preparation phase follows it. The reason for combining the two phases is that you cannot prepare a data set properly without knowing *what to repair* and *what variables* (data elements) are available for deriving new variables in the Data Preparation phase.

Four Data Understanding tasks in CRISP-DM

- Collect the initial data set
- Describe the data set
- Explore the data set
- Verify data quality

Collect the initial data set

The first task includes activities that were addressed in [Chapter 5](#) – Data access. The fourth task, Verify Data Quality, will be discussed in more detail in the next chapter, but in this chapter, we will refer to some missing data considerations to look for when exploring data. In this chapter, we will focus on the second and third tasks: Describe the data set and Explore the data set.

The Describe initial data set

CRISP-DM defines Describe Data as “Examine the ‘gross’ or ‘surface’ properties of the acquired data”. Perhaps no aspect of this task is more important than assessing the level of measurement of the variables in the data set. Software requires the proper definition of variable type, and this ‘surface’ level assessment drives many of the decisions that must be made later. We are not interested at the moment in how the data are *stored* – that is, whether as a string, an integer, a date, or a real value. Data format is indeed important and will be discussed in the next chapter. In this chapter, our focus is

on the behavior of data in a model and the verification that our data is suitable for use with the modeling algorithms we intend to use. One of the most important aspects of a data set is its *level of measurement*.

The top-level property we will explore is whether variables are nominal, ordinal, or continuous. We will also use a somewhat more granular list of categories in order to be more specific and, in the examples section, we will see how software can refer to these categories in slightly different ways. The notion of level of measurement dates back decades, long before the contemporary software packages that we discuss in this book. The use of the terms has changed relatively little over the years, but some software will use more than three categories.

Why is the proper definition of level of measurement important? Software will often require it, but our understanding is even more important than that. The modeling algorithms will treat different levels of measurement in different ways automatically, so our model’s effectiveness relies upon the correct declaration of the measurement level. Statistical analysis software will rarely enforce the declaration of measurement type, but we might generate nonsense if we fail to take it into account. For instance, it is meaningful to speak of pounds (of weight) per inch of height, but it is not meaningful to speak of pounds per zip code. Even more mundane tasks are more effective when this concept is taken into account. For instance, there are graphics types that are designed specifically to display different combinations of variable types. Statistical tests can be organized into a kind of taxonomy built around level of measurement. Software routines designed to automate basic reports or to recommend chart types automatically usually rely upon these definitions of measurement type.

[Table 6.1](#) shows one form of such a taxonomy of measurement type for the KDD Cup – 1998 competition data set (available in the UC-Irvine Machine Learning Repository at: (<https://archive.ics.uci.edu/datasets?search=KDD%20Cup%201998%20Data>)).

Much of the discussion presented in this chapter refers to this data set.

TABLE 6.1 Solution design considerations associated with various measurement types in the KDD Cup – 1998 data set.

Type	Examples in case study data	Description	Data prep considerations
Unique IDs/keys	Customer_ID StreetAddress Transaction_ID TelephoneNumber EmailAddress	These fields have a unique value for every row. This information is useful for data integration.	Check for data format problems (leading zeros). Do not use as a predictor variable.
Categorical:			
Only one value	Telephone Country code Country name	Only one possible value.	Variables that do not vary are never useful. Do not use as an input variable.
Binary	Homeowner Phone_Cust Target_B	Has only two values – often 0/1 or T/F or Y/N. Also referred to as a Boolean. Some tools call these variables <i>Flags</i> .	Sometimes only the “Y” is present, with the “N” represented only with blanks.
Low-order nominal	Gender ProductCategory PaymentMethod	Nominal variable with a modest number of categories – below 10–15 or so.	Some techniques can convert them automatically into binaries.
High-order nominal	Domain Cluster Phone Model State Title	Nominal variables with more than 15–20 categories.	Can cause problems if they are allowed to be converted automatically into binaries. Usually, it is necessary to remove them, group them into fewer categories, or use them as a lookup table.
Very high order nominal	Product_Code/SKU City ZipCode	Nominal variables with several dozen or even hundreds of possible values.	Do not use them directly in the model.
Ordinal	INCOME WEALTH1	Categorical variable with meaningful ordering/ranking.	Declare as ordinal. Possible candidate for imputing with the median.
Scale/continuous	Age Sales Amount	Distinguish between ratios and intervals.	
Date/timestamp	PromoDate DOB TransactionDate	Temporal variables like dates and time stamps.	Very useful for deriving temporal abstractions in historical data sets.
Text			

Table 6.1 shows a descriptive statistics report for sample variables in one of the practice data sets. Notice that more than three categories are described. In particular, notice that the high-order nominal category has been divided into five subcategories. Some of the examples presented in this chapter will show these declarations.

Understanding level of measurement and variable types

Unique identifiers and IDs

Variables like Social Security number, customer ID, or residential address are not explored per se but are important for assessing sample size and the structure of the

data. For example, you might want to make a random sample of only those records with customer ID values above a threshold, because lower numbers pertain to customers who joined after the financial crisis in 2008. In [Chapter 8](#), we will consider errors in a high-order nominal variable, ZIP, because we must clean up these errors to permit successful mailing operations during model deployment. If you include unique identifiers or IDs as inputs to a modeling algorithm, you are likely to generate an overfitted model in which the pattern in the data set would be “memorized” and may cause you to think that the model was very good, but yet it may fail utterly when applied to predict new data.

Variables with one value

Variables with only one value in all rows can play no useful role in the predictive modeling part of the analysis. Some modeling algorithms will issue an error when they encounter them in the input data set. They are almost always indicative of data access queries designed to define the universe of data that is to be analyzed. For instance, The RFA_2R variable (recency code as of the most recent campaign) has the code = L (lapsed donor) in every row; all of the donors in this data set are lapsed donors, because the selection criteria of the organization providing the data set extracted only lapsed donors for further analysis. Therefore this variable must be removed.

Variable “pairs”

For example, the variable ODATE is documented in the data dictionary as the date of the first gift, and the variable FISTGIFT is also defined as the date of the first gift but at the monthly level instead of the yearly level of ODATE (the second two digits for all rows are all “01,” referring to January of the year as a placeholder). ODATE should be removed, because its information is contained in FISTGIFT. The variable names are constrained to eight characters to be consistent with the addressing rules of the Fortran language.

Binary variables

A binary variable has one of two possible codes in each row. While a two-category variable like gender would count in the strictest interpretation, these variables are usually “Boolean” in nature – that is, they are true/false variables. The name derives from George Boole, the English mathematician. Multi-categorical variables can also be expressed as binary variables by deriving a new variable for each category, as in Marital Status in [Table 6.2](#). Statisticians may derive similar variables from each class in a categorical variable; this operation is called “dummy coding” or “one-hot coding.” Dummy variables are essentially the same as binary variables. Binary and dummy variables can be problematic at times, because data input routines treat them often as integers (1 and 0), but classification model algorithms must operate on categorical target variables. If you want to use a binary variable as a target variable in a classification algorithm, you may have to change the type to a categorical variable before submitting it to the modeling algorithm. It is important to understand that these dummy variables will almost never be normally distributed; therefore they will not satisfy the normality requirement for parametric statistical algorithms. You can always treat them as categories, however. This type of variable is very frequently produced during feature engineering operations discussed in [Chapter 10](#).

Low-order nominals

Text variables can be used directly by some algorithms (like Decision Trees) but not by other algorithms (like Linear Regression). Some text variables have only a few classes (like Gender), and some have a relatively large number of classes (like car model). Both examples are often treated the same, but to do so is a serious mistake. There is no hard and fast rule to distinguish between low- and high-order nominals, but it is considered to be relatively safe to stay in the single digits for the number of classes in a categorical variable. The reason that this mistake is so important is that many algorithms convert the text classes to binary variables; it is OK to have ten or fewer binaries, but it is not OK to have hundreds of binaries.

TABLE 6.2 Marital status categories converted to binary variables.

Customer #	Original variable	Single	Married	Divorced	Separated	Widowed
1	Single	1	0	0	0	0
2	Married	0	1	0	0	0
3	Single	1	0	0	0	0
4	Widowed	0	0	0	0	1
5	Separated	0	0	0	1	0
6	Married	0	1	0	0	0
7	Divorced	0	0	1	0	0
8	Married	0	1	0	0	0

High-order nominals

If you declare zip code as a nominal and use it as a candidate predictor variable to input to a modeling algorithm, you are making a serious mistake. Many analytical and data visualization routines will fail if there are too many classes in a categorical variable (the threshold for KNIME is 200). The automatic processing of zip code could generate as many as 100,000 binary variables, very few of which would have any predictive value.

The algorithm might produce some obscure rules describing a relationship between the outcome for white cars on Friday, which may be simply the result of sampling bias that created the training data set. Biased rules like this lead to overfitting of the model by the algorithm on the training set but fail miserably on the validation data set. The only way to handle a variable like zip code is to draw upon geographic commonalities in zip code that might be predictive. For example, a particular set of zip codes might include a relatively wealthy residential area like Hope Ranch in the Santa Barbara area. Another commonality that might be used directly is that the first number of the zip code refers to the geographical region of the United States, ranging from low numbers in the Northeast Atlantic areas to high numbers in the Pacific Northwest areas.

Ordinal variables

Some variables are low-order nominals, but they have a meaningful sequence. Therefore GENDER doesn't qualify, because there is no meaning in the sequence of GENDER codes. But if we derive SES (socioeconomic status) from the second byte of the DOMAIN variable (indicating Highest, Average, Lowest), it would matter. Why is the distinction important? Some modeling algorithms will treat SES rather differently from nominals like GENDER. In Decision Trees, for instance, any nominal category can be "collapsed" or combined with any other. People in different SES levels, however, might have different propensities to donate money, and a tree algorithm might combine them automatically. This variable is numeric, but it is not a scale variable (see below); they are a special kind of categorical variable. Proper declaration is important.

Scale variables

Scale variable values are continuous numbers with equally spaced units. For example, a pound of weight increment is the same, no matter where it falls on the continuous scale. Therefore the calculation of means (averages), medians (middle value), and standard deviations makes sense for use as metrics of data distribution. It would not make sense, however, to perform these calculations on variables like zip code.

Dates

Dates are essentially a kind of scale variable, albeit a special kind. They are on a continuum, and they are

equally spaced, but they are treated differently for a number of reasons. They are stored inside computers in the form of time since an arbitrary date. For instance, Excel uses January 1, 1900 as a reference point and calculates the number of days since then and a given date in a variable. If you set the data type of a variable as a date, the modeling algorithm will assume the proper relationship between the date in terms of the internal data representation. If you type a date variable as an integer (e.g., 20120301 representing March 1, 2012, or 20160101 representing January 1, 2016), the algorithm will assume that the relationship between the two dates is represented by the numbers themselves, not their representation as dates.

The best use of date variables is to derive other variables from them. Such variables that can be derived from date variables include (1) time in days, months, or years between two date variables (e.g., TIMELAG = months between the first and second donations); (2) time since the reference date for each of several date variables; (3) amount of the last donation (e.g., LASTGIFT).

Text

Text variables require special treatment which is beyond the scope of this discussion, but we will define it and describe it briefly. Raw text gets generated in the forms of tweets on the platform X, Facebook comments, call center transcripts, open-ended answers to survey questions, healthcare notes, and a host of other data sources. While "mining" the text is a book-length topic (see [Miner et al., 2012](#)), there are some very basic calculations that can be performed. For example, you can derive a binary variable for the presence or absence of a doctor's notes. Also, you can derive a new variable to express the length of the comment when it is present. Text management operations like this might require a lot of work, but they might prove to be worth the effort. Some data analytics tools provide text search capabilities to derive binary variables representing the presence or absence of a given word or phrase in a body of text, like physician's notes.

Exploring data

Often, the Data Understanding phase of CRISP-DM is confused with discovering the most promising predictors or other more direct insights in the data set, because it is partly associated with data visualization and with generating reports from our data. Neither of those associations is correct. This phase is far too early in the CRISP-DM overall process to perform either of those activities. You can gain insights that may contribute later to these activities, but we should not equate these insights with predictive modeling operations. With the rise of dedicated powerful data visualization software (like Tableau), the temptation arises that one can uncover predictive insights visually and forgo any predictive modeling activities.

The major task in data exploration is to generate a “to-do” list for all of the following operations. In the process, we can build a list of variables that require reformatting, cleansing, and imputation (filling) of missing values. These activities will provide insights into how various legacy data sources will fit together during data integration to form the final modeling file. Therefore our primary task is to gain a sufficient understanding of our data elements to guide our subsequent data preparation activities. This is the reason that the Data Understanding phase of the CRISP-DM process model is considered to be a part of the overall data preparation process.

Another source of confusion can stem from the many feedback loops in the CRISP-DM process. Data Understanding as a phase comes between Business Understanding and Data Preparation in the formal process, but the large circular arrow in the process diagram is intended to show that the flow of information in the overall modeling operation is not a waterfall process. Rather, the overall effort may circle back to a previous phase or may iterate back and forth between two phases. For example, some Data Understanding tasks (e.g., identification of recoding needs) must be performed on data sets from legacy sources before these data sources are integrated. Also, the recoding

activities (normally a part of the Data Preparation phase) must be completed before data integration is performed. Because the focus of this book is on Data Preparation, we will focus on the earlier of those two Data Understanding waves of activity in this chapter. We will focus on identifying those operations that must be performed before the data set is submitted to the modeling algorithm.

Once we have assigned the proper data type to each variable, we can explore them one at a time. This exploration is often referred to as running a “univariate” analysis. You can only get a limited insight into your data by analyzing one variable at a time, but it would be a mistake to leap too quickly to multivariable type analyses. In that regard, you should be focused on potential errors and problems which are obscured in complex visualizations and reports. Keep your Data Understanding tasks simple and be methodical in performing them. The potential for uncovering exciting insights can wait – ensure first that the data is trustworthy and ready for analysis. Table 6.3 provides some advice on how to analyze each type of potential problems that might arise.

A bivariate analysis is simply an analysis of two variables. The trick to doing this effectively is to

TABLE 6.3 Appropriate univariate analysis for each variable data type.

	Appropriate descriptive statistics and visualizations	Potential problems to anticipate
IDs, keys	Simple counts, sorts, and aggregations	• What is the maximum number of transactions or rows of data assigned to each ID? • Do there appear to be any duplicate IDs? • Are there null or blank IDs, including special codes for missing like 999-999-9999?
Categorical variables (binary, nominal, ordinal)	Mode, bar chart	• What is the most common category (mode) – does one category predominate or are there a lot of categories that are barely represented? • Do there seem to be any incorrect categories (misspellings, case sensitivity, redundant categories)? • Are there “catch-all” categories that are overly used, like “Other” or “Misc.”? • Are there potential opportunities to combine categories without losing information?
Continuous variables (scale and ratio)	Min, max, mean, median, percentiles, boxplot, and histogram	• What are the mean and median – are they similar or different to each other? • Is the data shaped like a bell curve or is it skewed? • Are there zeros, nulls, or blanks? Is there perhaps a confusing mix of all of them? • Are there values that are outside the range that make sense? Are negative values relevant and/or present?
Dates	Min, max, mode, simple counts, and sorts	• What are the min and max dates? Are those ranges the direct result of how the data was queried, or are they meaningful? • Is there one (or more) dates with a very large number of values? Might it be a default date of some kind representing missing data? • Are there any null or blank dates? • Are the dates in the required format? Are any of them being stored as string fields or in a nonstandard format?
Text	Simple counts and sorts	• How many rows are completely blank including rows that are populated only with “none,” or “blank,” or “NA”? • What is the typical length of the text: a word or two, a sentence or two, a paragraph, or more? Is there tremendous variation in length? • Does it appear to be machine generated or populated by drop-down menus?

understand that it is driven by the level of measurement of each of the two variables. The most common bivariate analysis is to calculate the linear correlation coefficient between the two variables. This analysis could be done in this phase, but we choose to wait until [Chapter 11](#) – Feature selection, to do it, because the results will be used there as one of the bases for variable selection.

KNIME examples in Data Understanding

Example 6.1. Descriptive statistics

This example shows the KNIME facility to display summary statistics. The completed KNIME workflow is shown in [Fig. 6.1](#).

Steps in this example:

- Complete Example KNIME 4_1
- Connect a Histogram node to the Table Reader node
- Generate a histogram plot of AGE by bins
- Connect a Pie Chart node to the Table Reader node
- Generate a pie chart for the HOMEOWNR variable distribution

Detailed directions for configuring all the nodes shown in [Fig. 6.1](#) are available on the book Web page.

IBM SPSS modeler examples

Example 6.2. Descriptive statistics

This example demonstrates the use of the Examine functions of the Statistics node. The complete modeling stream is shown in [Fig. 6.2](#).

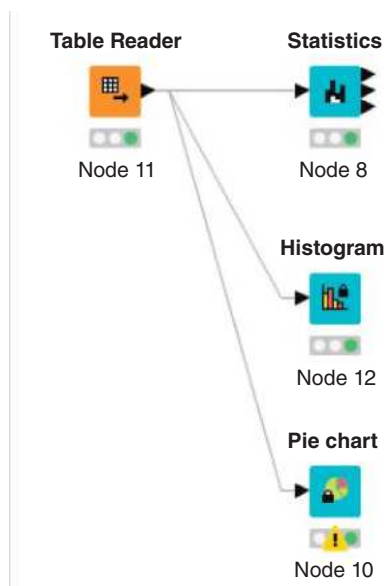


FIG. 6.1 A data visualization KNIME workflow.

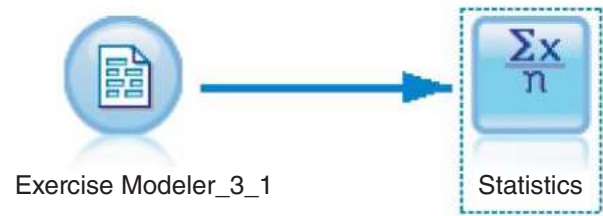


FIG. 6.2. Use of the statistics node in modeler.

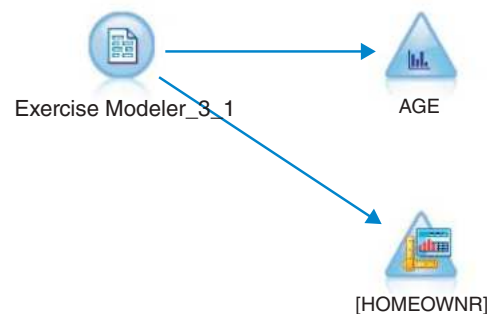
Steps in this example:

- Add source node to a new stream and load the Example 3_1.CSV data file into it
- Connect a Statistics node to the source node
- Generate a descriptive statistics report

Example 6.3. Basic data visualization in modeler

This example demonstrates some of the options in Graphs palette.

The completed Modeler stream is shown in [Fig. 6.2](#).



Screenshot Modeler 4.2 The completed Example_4_2 stream.

Steps in this example:

- Create a Histogram of scale variable, Age
- Create a Pie Chart of a categorical variable, HOMEOWNR

Detailed directions for configuring all the nodes in these workflows are available on the book Web page.

Keywords and phrases

Process, Code, Automated, Categorical variables, Date variables, Binary variables, Scale variables, Descriptive statistics, Univariate analysis, Data preparation, Data visualization.

Reference

Miner, G., Delen, D., Elder, J., Fast, A., Hill, T., Nisbet, R., 2012. Practical test mining and statistical analysis for nonstructured text data applications. Elsevier, 1053 p.

Visualizing data for understanding

Foundations of data visualization: John Tukey

Arguably, John Tukey could be called the “father of data visualization.” The history of visual data analysis can be traced from John Tukey to Leo Brieman to CRISP-DM and ultimately, to the development of “traditional” machine learning. This book is focused on that development path. John Tukey was one of the most important statisticians of the 20th century. Although [Tukey \(1977\)](#) discusses the subject of exploratory data analysis through a 1970s lens, the ideas in his book are still pertinent and influential in data analytics today. In particular, his famous “box-and-whiskers” plot is a graphics option available in almost every example of data analysis software available today. A metaphor that he uses to describe data exploration is useful in achieving our goals for this chapter posits that:

A detective investigating a crime needs both tools and understanding. If he has no fingerprint powder, he will fail to find fingerprints on most surfaces. If he does not understand where the criminal is likely to have put his fingers, he will not look in the right places. Equally, the analyst of data needs both tools and understanding.

Most advice on data visualization focuses on reporting, dashboarding, and business intelligence. While these aspects are important, the data scientist also must perform other types of visualization. In the early stages of a modeling project, you may produce data visualizations in high volume for your own consumption. For instance, the initial goal is to produce a visualization for every variable. The visualization task will be very different when you put the task of regular reporting of model results into production, or when you are writing a milestone report.

Importance of the audience of visualizations

A helpful way to approach documenting the progress through the states of a modeling project is to consider the four possible audiences of your visualization outputs:

- Yourself
- Subject matter experts (SMEs)

- Your internal customer(s)
- General business stakeholders

The amount of detail, comprehensiveness, and granularity of data visualization will decrease dramatically for these audiences as you move through the stages of a modeling project.

Yourself as the audience

For yourself as the audience, you might create visualizations for every variable (e.g., box-and-whisker plots). This task may seem arduous, but most data visualization tools can facilitate it. These charts need not be “pretty”; only you will see them. The goal is to spot unusual patterns (like very long “whiskers” on a box-and-whisker plot) and other issues that must be addressed during data preparation (e.g., handling of outliers).

Subject matter experts as the audience

A subset of some odd or quirky patterns might be referred to as specific subject matter experts (SMEs), who can contribute their expertise to help you fix specific issues in the data set.

Internal customers as the audience

Internal customers are those stakeholders who will use the modeling results directly (e.g., sales and marketing staff). These stakeholders should be given only those visualizations that support their management operations and decision making to help you identify the most useful patterns to support those concerns.

General business stakeholders as the audience

Finally, members of your general business audience (e.g., management) should be shown only carefully crafted graphics showing strong relationships that you have verified are both valid and meaningful.

This chapter will focus on the earliest stages of that process, where you seek to understand the patterns. We must do that with the appropriate audiences in mind and prepare to serve them.

Seeing is believing

"‘I see,’ said the blind man to his deaf wife as he picked up his hammer and saw,"

This humorous aphorism points to the act of “seeing” that implies understanding, rather than visual perception with the eyes. Stephen Few (2009) focuses on the distinction between showing and seeing. He relates “seeing” to “Sensemaking,” which is a good concept to use for describing the goals of this chapter. The basis for this term is expressed by two quotes from Few (2009):

On the presentation of quantitative information:

“Most presentations of quantitative information are poorly designed—painfully so, often to the point of misinformation. ... Even the best information is useless, however, if its story is poorly told. This problem exists because almost no one has ever been trained to design tables and graphs for effective and efficient communication.”

On visual data sensemaking:

“Now You See It does for visual data sensemaking what Show Me the Numbers does for graphical data presentation: it teaches simple, fundamental, and practical concepts, principles, and techniques that anyone can use—only this time they’re exploring and making sense of information, not presenting it. These techniques rely primarily on something almost everyone has: vision. They use graphs to display

data in ways that make meaningful patterns visible to reveal the stories that reside within. These techniques also involve interacting with data in particular ways to tease out relevant facts and their meanings.”

Case Study: visualizations of the Ames Housing data set

In this chapter, we will be using the KNIME Analytics Platform to visualize the Ames Housing dataset. This is a well-documented data set developed by Dean De Cock of Truman State University for statistics education. You can download this data set here: <https://www.kaggle.com/datasets/prevek18/ames-housing-dataset?resource=download>

Initial descriptive data analysis and level of measurement

Most software packages provide some sort of semiautomated way to perform an initial descriptive analysis of an input dataset. The KNIME Analytic Platform offers the “Data Explorer” node to perform this task shown in the KNIME workflow (Fig. 7.1).

Numerical variable description

It is critical that the software identifies the level of measurement correctly, as we discussed in Chapter 6. Note that Table 7.1 contains only numeric variables, and Table 7.2 contains only categorical variables. The output

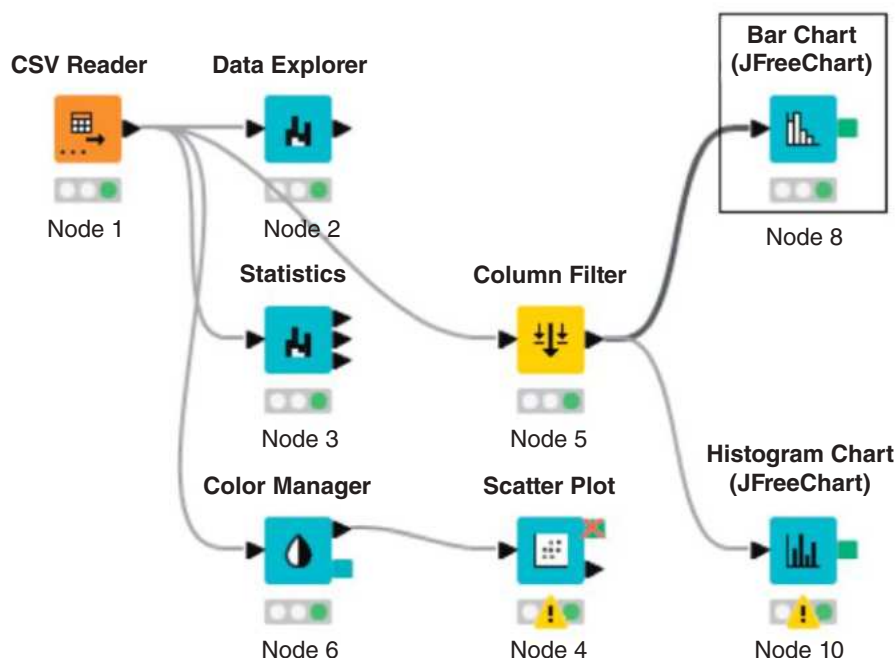


FIG. 7.1 The KNIME workflow visualizations of the Ames Housing dataset.

TABLE 7.1 Selected numerical variables in the Ames Housing data set.

Numeric										
Nominal										
Data Preview										
Search:										
Column	Exclude Column	Minimum	Maximum	Mean	Standard Deviation	Variance	Skewness	Kurtosis	Overall Sum	No. zeros
Order	☐	1	2930	1465.500	845.962	715652.500	0	-1.200	4293915	0
MS SubClass	☐	20	190	57.387	42.638	1818.001	1.358	1.387	168145	0
Lot Frontage	☐	21	313	69.225	23.365	545.939	1.499	11.235	168908	0
Lot Area	☐	1300	215245	10147.922	7880.018	62094679.889	12.821	265.024	29733411	0
Overall Qual	☐	1	10	6.095	1.411	1.991	0.191	0.052	17858	0
Overall Cond	☐	1	9	5.563	1.112	1.236	0.574	1.491	16300	0
Year Built	☐	1872	2010	1971.356	30.245	914.782	-0.604	-0.502	5776074	0
Year Remod/Add	☐	1950	2010	1984.267	20.860	435.152	-0.452	-1.342	5813901	0

TABLE 7.2 Selected categorical variables in the Ames Housing data set.

Numeric

Nominal

Data Preview

Search:

Column	Exclude Column	No. missings	Unique values	All nominal values	Frequency Bar Chart
PID	☐	0	>1000	0527375100, 0534275060, 0908225290, 0528458020, 0902106010, [...], 0906226080, 0909427200, 0923225390, 0528292070, 0528235090	Not all nominal values calculated,
MS Zoning	☐	0	7	RL, RM, FV, RH, C (all), A (agr), I (all)	
Street	☐	0	2	Pave, Grvl	
Alley	☐	0	3	NA, Grvl, Pave	

of the KNIME Data Explorer node provides three tabs: Numeric (to screen for numerical variables); The Nominal tab, and; the Data Preview tab. The PID variable in Table 7.2 is a number, but KNIME assumes that it is only a character, not a number, and renders it as a string.

Although this is just the first few rows, you can see the kind of information that is provided by the Data Explorer for numerical variables. Consistent with our discussion in Chapter 16, things that we would look for would include minimums and maximums that were the same (no variance), unexpected negative values, and

unexpected zeros. The KNIME report has been designed with these needs in mind. This second tab shows the report for nominal variables. Note the “very high order nominal” variable PID (as we discussed in [Chapter 16](#)). This is an ID field and should not be used in any graphics or analyses. Note the “exclude column” feature shown in the image. Again, the KNIME team anticipates that an exploration at this level might result in filtering out variables from “downstream” operations in the workflow. (A histogram can be viewed by clicking on the blue plus symbol in the report for numeric variables.)

Categorical variable description

Categorical requires different variable description techniques that are used for numerical variables.

Note: the pattern in MS Zoning, Street, and Alley. All three show some categories that are much more common than others. A possible solution might be to dichotomize MS Zoning (residential versus “all other”).

A comprehensive univariate analysis

A visual “univariate” analysis is nothing more than an appropriate graphic showing the data distribution for each variable in the data set. We explored some of the options for each variable type in [Chapter 16](#) when we listed the various levels of measurement, including the more comprehensive list that we discussed. In this chapter, we’ll look at some examples and give some guidance as to what to look for.

The Data Explorer provides its graphic in the form of a simple bar chart of the data distribution for each nominal variable. There is sometimes little added value in repeating this. However, there would be a possible exception in that if the Data Explorer (or some other automated report) is not making clear the missing value pattern, we want to take a closer look. For instance, the Garage Finish variable has a “NA” category, because it doesn’t apply if there is no garage.

A more comprehensive report for each variable is provided by the Statistics node, which is shown in [Table 7.3](#).

The bar chart for the Garage Finish variable is shown in [Fig. 7.2](#).

Naturally, a bar chart would be a poor choice for a numeric variable. For numerical variables in the dataset, you can generate a histogram. [Fig. 7.3](#) shows the histogram of Lot Frontage. Notice the dramatic outliers (two of them) at 313 feet frontage. While these frontage numbers are plausible, they may be viewed as outliers compared to the other values. We should check to determine that these values are correct. If they are correct, these might not belong in the data population being analyzed. For instance, a very large value of lot frontage could be a warehouse and not a

residence. If we are conducting a residential property analysis, we should delete them during analysis of this variable. Otherwise, the population of other values would be associated with a mean value that is not characteristic of the data population of residential properties.

Bivariate analyses

It may seem unrealistic to examine every variable pair. This moderately wide dataset would generate 82×82 (or 6724) pairs to analyze. That would be hundreds of variable pairs. However, visually reviewing each pair is surprisingly quick with practice. When one takes into account the extremely time-consuming problems that can result from rushing into modeling (especially when using “black box” modeling techniques) without proper data exploration, one will conclude that the upfront effort is worth it. Nonetheless, as the number of input variables increases, the value of that number squared starts to seem daunting. Make an effort to look at *every input against the dependent variable at a minimum* (81 pairs). Then supplement that with an effort to look among the inputs at variable pairs that seem likely to interact. Also, if all input variable pairs are not possible, consider the “top” variables, perhaps the top 10 or so.

Since our dependent variable is a scale variable, sales price, we will need a visualization technique for two scale variables (for our numeric inputs) and a technique for one scale and one category (for our nominal inputs). In the first situation, we’ll need to look at scatterplots. [Fig. 7.4](#) shows the scatterplot of lot area and Sales price.

In this scatterplot, we have lot area on the x-axis and our dependent sales price on the y-axis. Note the four data points on the medium and far right of the scatterplot. These data values might be considered as outliers to focus the analysis on the vast majority of properties below 60,000 square feet. A bit of detective work reveals that these properties are indeed residences, but it is still important to explore further. The object of this analysis is to prepare for an eventual meeting with an SME. If you are fully prepared for the meeting, you will get more benefit from their expertise. Adjusting the specifications of the scatter plot reveals that the most extreme outlier (and another one near it) belongs to the Timber neighborhood ([Fig. 7.5](#)). Is there something distinctive or special about this neighborhood? The SME will likely know, and we can determine together if this data point belongs in our analysis. Note that we should not decide to delete these properties until we can determine that they are related to our business problem.

Tukey’s box-and-whisker plot

Visualization of the combination of a scalar and a nominal variable is well served by Tukey’s box-and-whisker plot shown in [Fig. 7.6](#).

TABLE 7.3 A portion of the Statistics table output from the Statistics node sorted on No. missings in descending order. Notice the numbers in the No. missings column in the top 6 rows. This report can be used later on to impute missing values for those six variables.

Row ID	S Column	D Min	D Max	D Mean	D Std. deviation	D Variance	D Skewness	D Kurtosis	D Overall sum	I No. missings	I No. NAs
Lot Frontage	Lot Frontage	21	313	69.225	23.365	545.939	1.499	11.235	168,908	490	0
Garage Yr Blt	Garage Yr Blt	1,895	2,207	1,978.132	25.528	651.7	-0.385	1.827	5,481,405	159	0
Mas Vnr Area	Mas Vnr Area	0	1,600	101.897	179.113	32,081.327	2.607	9.287	296,214	23	0
Bsmt Full Bath	Bsmt Full Bath	0	3	0.431	0.525	0.275	0.617	-0.748	1,263	2	0
Bsmt Half Bath	Bsmt Half Bath	0	2	0.061	0.245	0.06	3.941	14.922	179	2	0
BsmtFin SF 1	BsmtFin SF 1	0	5,644	442.63	455.591	207,563.013	1.416	6.859	1,296,462	1	0

SalesPrice is the scalar (numerical) variable, and MS Zoning is the nominal (categorical) variable. Fig. 7.6 plots the box-and-whisker diagrams for six MS Zoning categories. This chart is helpful indeed because we can make several observations at a glance:

- I (industrial) and A (agricultural) are too rare to establish a pattern.
- RL (low-density residential) and RM (medium-density residential) have a large number of cases, a substantial amount of variance, and are the source of most of our outliers.
- While the FV category (Floating Village Residential) does not have the most expensive outliers, it has the highest median. It is also noteworthy that the FV category seems to lack any inexpensive houses. It has a substantially higher 25 percentile (indicated by the bottom of the box) than any other.
- RH (high-density residential) represents the less expensive homes and consistently does so. It has a relatively low variance.

Visualizations with multiple variables

As your exploration progresses and your understanding deepens, it is helpful to produce visualizations with several variables at once. This is because our ability to detect patterns visually is superior to our memory. Stephen Few, the data visualization author and expert, offers this advice:

We should avoid fragmenting information that we're examining by placing it on separate screens in locations that we can't see without scrolling. For instance, if we see a pattern in a graph and then try to compare it to a pattern in another graph that is on another screen, we will no longer remember much of what we were looking at previously once we bring up the new screen.

(Few, 2009)

The trick in making this advice work is to allow our graphics to become a bit more complex. But to make them coherent, we have to identify the strongest relationships. In this example, some of the best predictors of our dependent variable, sales price, are gross living area

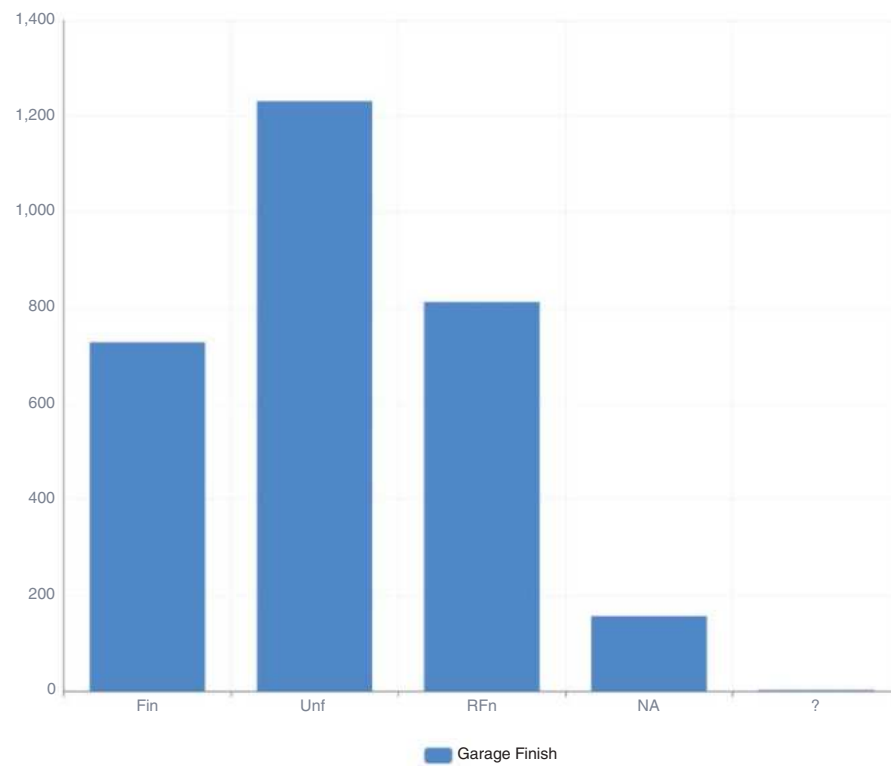
Bar Chart

FIG. 7.2 The frequency bar chart for the categorical variable Garage Finish.

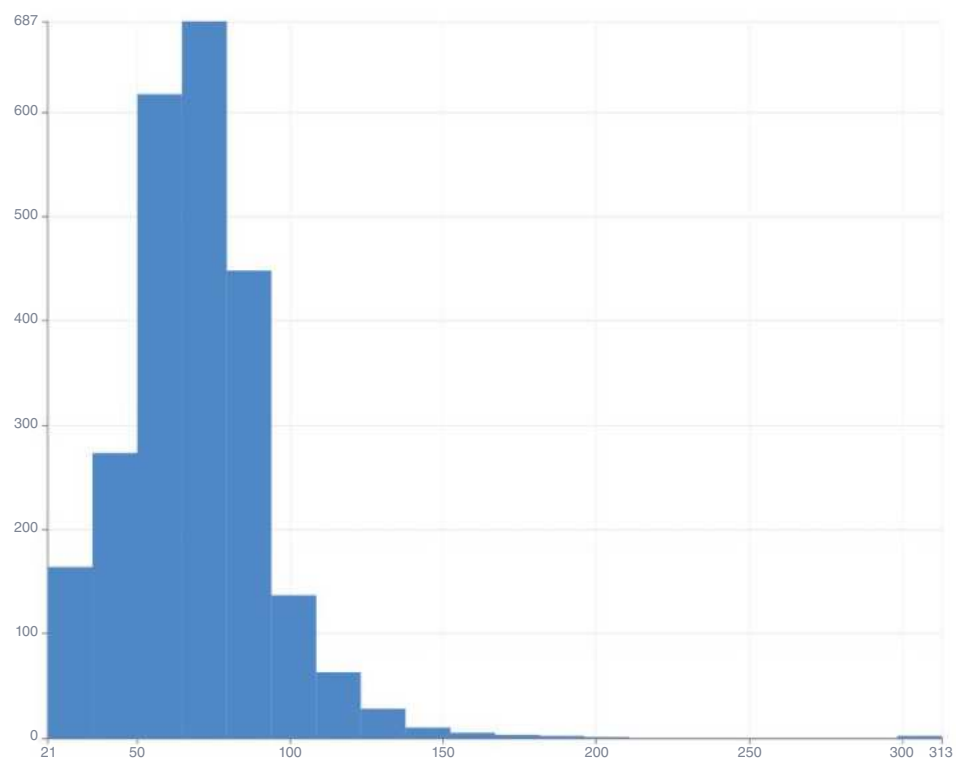
Histogram

FIG. 7.3 A histogram of the Lot Frontage variable in the Ames Housing dataset.

Scatterplot of Lot Area and Sales price

Ames Housing Dataset

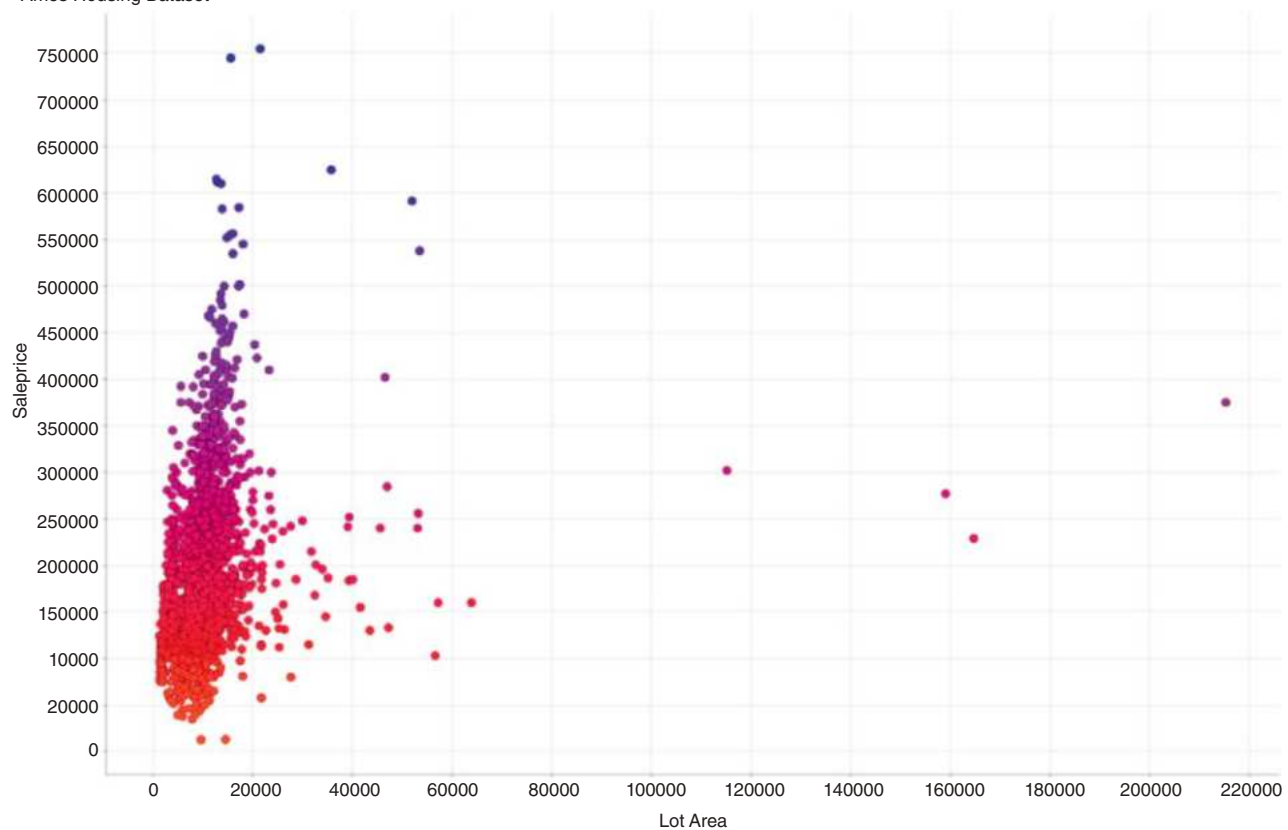


FIG. 7.4 A scatterplot of the relationship between sales price and lot area. Red circles indicate relatively low sales prices and blue dots represent relatively high sales prices.

and neighborhood. When this is done, however, the resulting graphic is difficult to read. A couple of data points have much larger bubbles (indicating higher lot size in this case), but that dimension doesn't seem to help us much. There is a very strong linear relationship between gross living area and sales price, but there are so many neighborhoods involved that color is very difficult to identify. This chart might be useful in helping us confirm high-level patterns, but it wouldn't be very useful to share with others. As we start to attempt a smaller number of curated graphics, we will be increasingly focused on graphics that would be useful in a milestone report. Let's proceed by first exploring a first attempt, then discussing how it can be improved, and then producing a second attempt, as shown in [Fig. 7.7](#).

There are a number of visualizations that might be even better to show complex relationships among a set of variables. We could:

- Aggregate our data by neighborhood so that we are looking at a few dozen data points and not 3000
- Clean up our labeling system

- Replace lot size (since it wasn't a sufficiently strong enough relationship) with the more useful sample size
- Try to identify why neighborhood matters by describing the neighborhood using the dominant zone (a derived variable)
- Eliminate a couple of neighborhoods that had fewer than 10 houses each

[Fig. 7.8](#) shows a scatterplot that relates four variables: Mean SalesPrice, mean Gross Living Area, Dominant Zoning, and Sample Size.

What patterns can we perceive in this graphic? The strong relationship between gross living area and sales prices is obvious. We can now read the neighborhood names clearly. Three relatively small neighborhoods (Stone Brook, Northridge Heights, and Northridge) have the highest prices and the largest homes. It is noteworthy that Northridge is not more expensive on average than the other two despite having the largest average gross living area. It is easiest to spot the largest neighborhood, North Ames, which appears to fit the overall pattern. Two small,

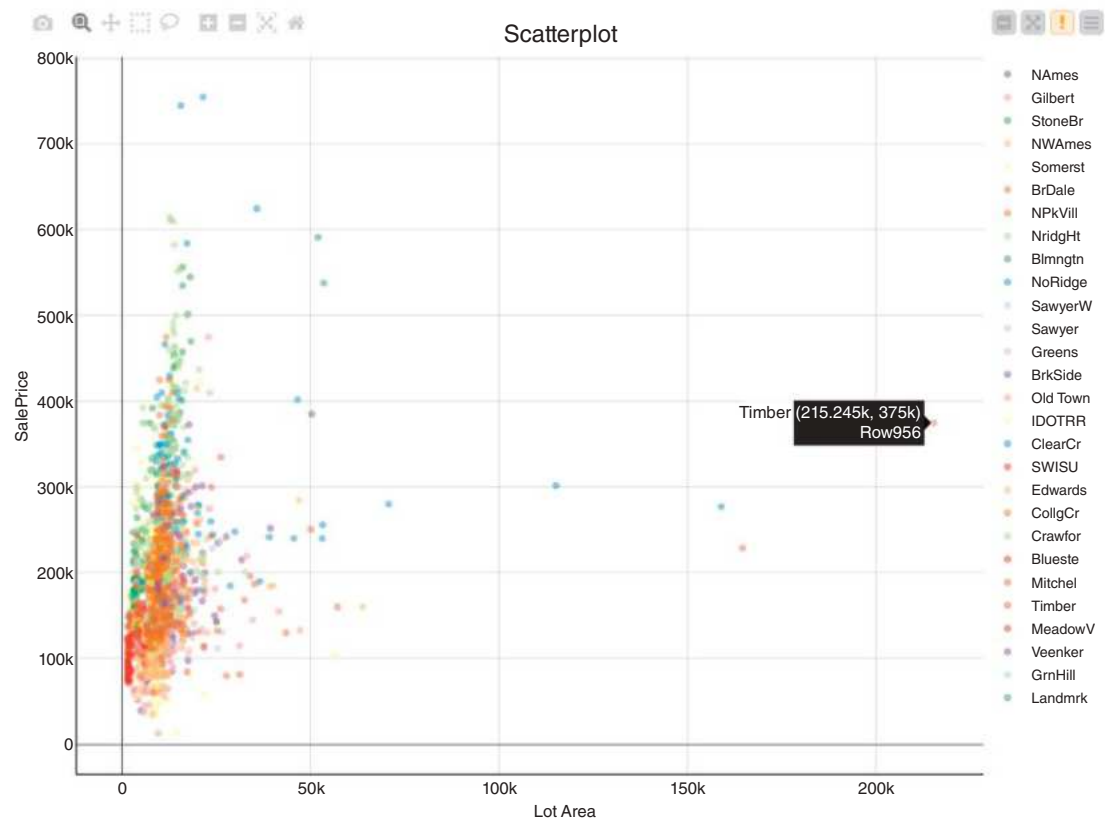


FIG. 7.5 The Scatter plot for the relationship of SalePrice to LotArea color coded for community names.

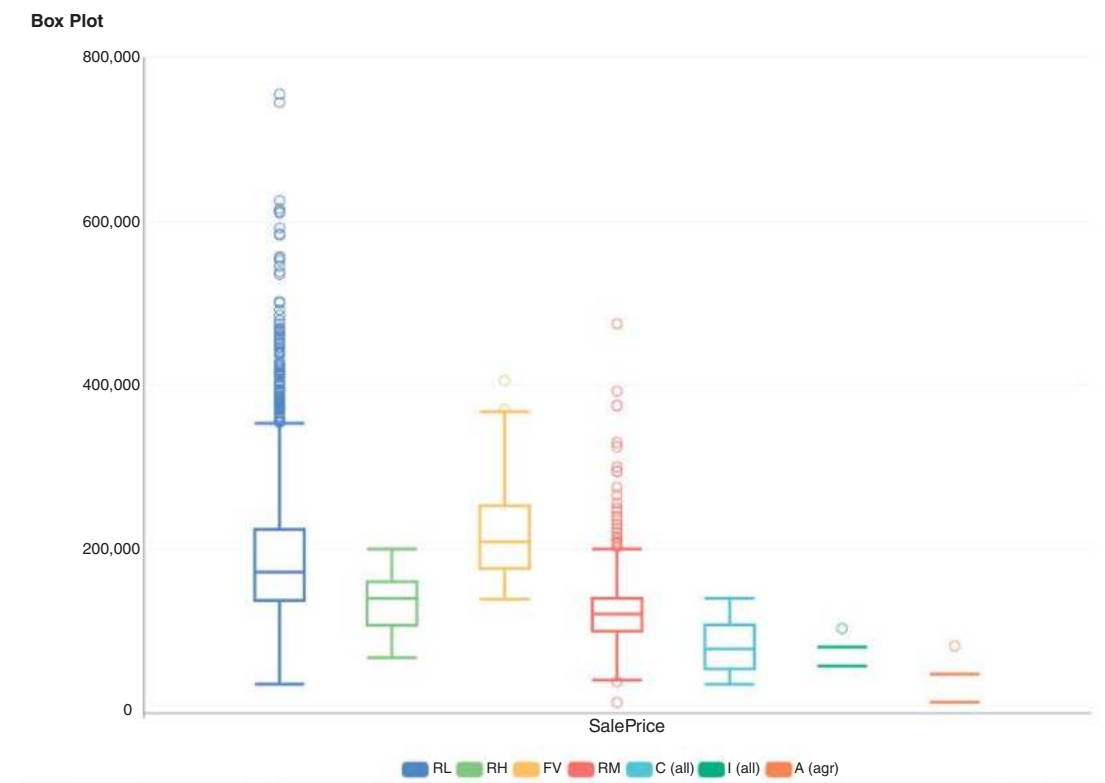


FIG. 7.6 Tukey's box-and-whisker plot.

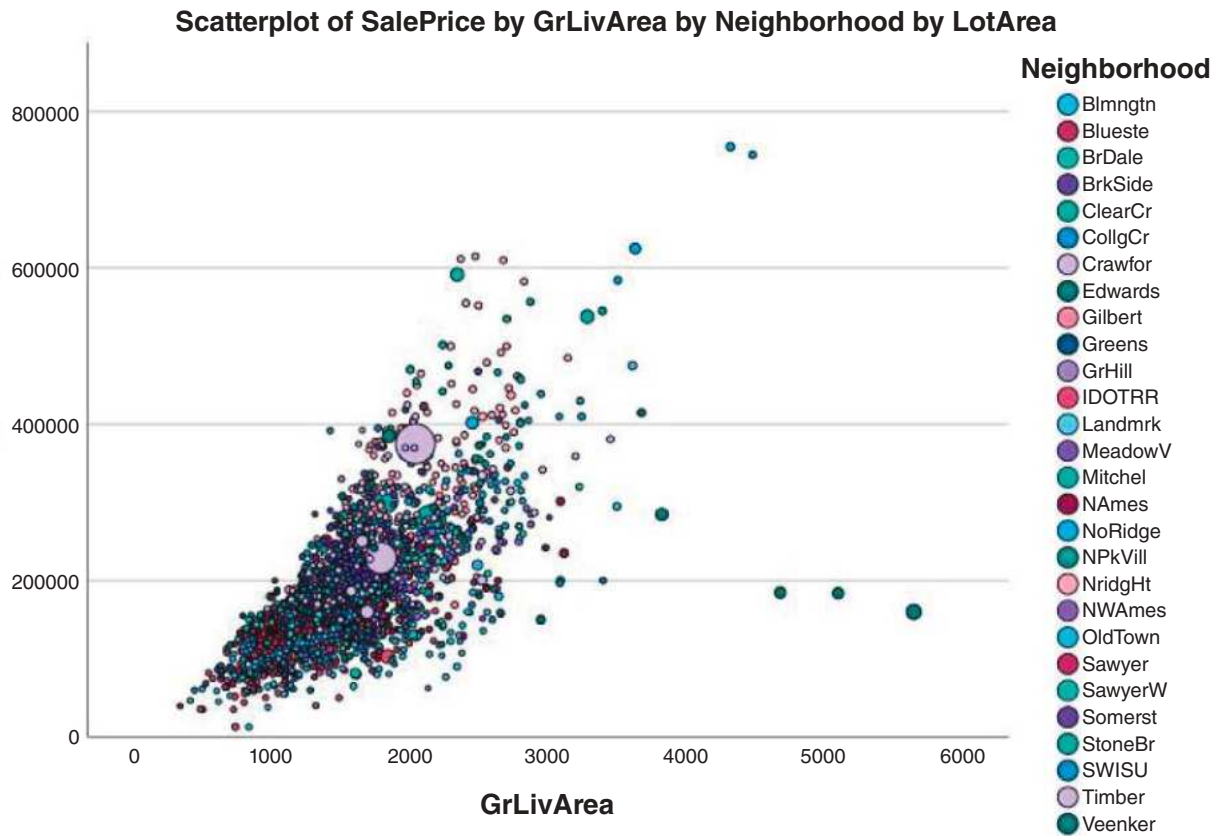


FIG. 7.7 Relationship among three variables, Saleprice, GrLivArea, and Neighborhood.

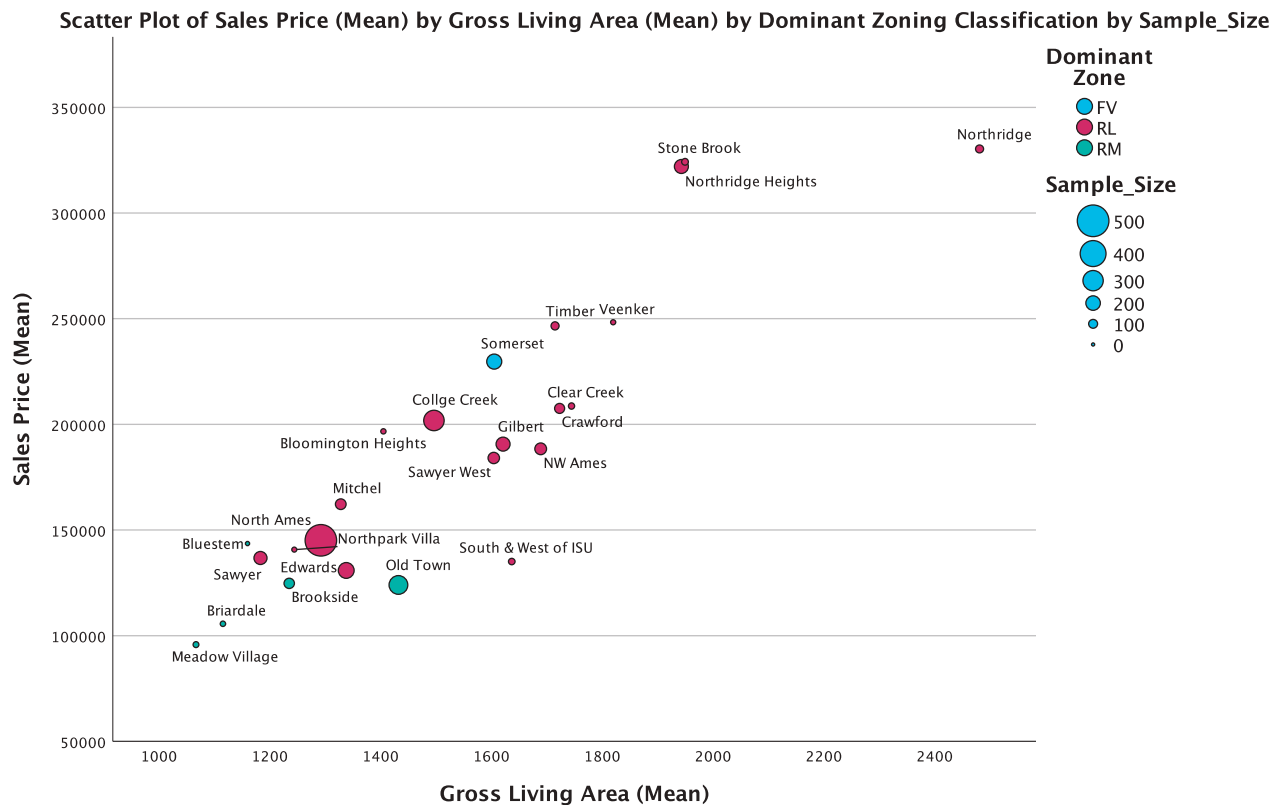


FIG. 7.8 A scatterplot to relate mean SalesPrice, mean Gross Living Area, Dominant Zoning, and Sample Size.

medium-density residential neighborhoods (as identified by the RM zone), Briardale and Meadow Village, have patterns similar to each other: smaller, less expensive homes.

Postscript

This case study shows how to explore combinations of input variables to reveal relationships. Descriptive information like that gathered in this analysis can be used as a basis for engineering new candidate predictor variables in [Chapter 11](#).

Keywords and phrases

Histogram; Scatterplot, Numerical, Categorical, Nominal, Patterns, Visualization, Audience, Outlier, Bar chart, Box-and-whisker plot, Subject matter expert.

References

- Few, S. Now you see it. 2009 <https://hub.knime.com/knime/extensions/org.knime.features.js.views.labs/latest/org.knime.base.node.stats.dataexplorer.DataExplorerNodeFactory/>
- Tukey, J., 1977. Exploratory data analysis. Addison-Wesley, 688 p.

Data cleaning

Preamble

Chapter 5 described the common tendency to charge ahead to modeling without doing sufficient data preparation (the “Chargers”) or to avoid it altogether (the “Avoiders”). Chapter 5 discussed the challenges of data access. A major challenge in analytics projects is being diligent to define clearly what data preparation operations must be done. This chapter will discuss some of the common data cleaning operations and explain why they must be done before modeling begins. Some analysts liken these operations to “cleaning the toilet.” This chapter will challenge you to liken these operations to “detailing the car” rather than doing the “grunge work” of data cleaning.

Data cleaning operation classification

There are several ways to classify data cleaning operations:

- a. Focusing on the source of data errors (single versus multiple sources), for which different versions of corrective measures are prescribed for each source type
- b. Classifying data cleaning operations according to steps in a logical sequence of processes, beginning with error identification and moving through error correction of existing information to imputing new information to fill missing values (van der Loo, and de Jonge, 2018)

In this chapter, we will follow the second approach to classification – organized by the sequence of logical processes of data cleaning.

The approach of (van der Loo, and de Jonge, 2018) directs raw data through several processing steps, each of which increases the information value of the data for use in analysis. The process sequence depicted in Fig. 8.1 is a very pragmatic one in that it expresses the need to do “first things first.”

Types of data cleaning operations

- Technical data correction
- String manipulation
- Data filtering
- Time-series filtering
- Outlier handling

Technical correction of data

If the time available for data cleaning is very restricted in a given application, the most important operations to perform are those that correct data errors from a technical standpoint (e.g., fixing wrong codes and data types). If more time is available, errors in data consistency can be addressed and fixed. With technically correct and consistent data to use, the analyst can be comfortable moving to the next important operation in data preparation – deriving new variables. It is crucial that the data input to data derivation operations be correct and consistent, or the quality of the derived variables will be compromised. The old maxim of “garbage-in, garbage-out” certainly applies here. The final stage in the value chain is analytical operations. The primary concern of de Jonge and van der Loo was to provide correct and consistent data for statistical analysis; the same holds true for predictive analytics.

The first activity in data cleaning must check the validity of a number of attributes of the data elements in a data record. In the data management world, these data elements were called *fields*. In predictive analytics, these data elements are called *variables* or *features*. Some practitioners distinguish between variables as raw data and features as transformed variables. In this book, these terms are used interchangeably, because that is the case in both the KNIME and IBM tools used in this book. Five types of validity checking are discussed below:

- Data type
- Data format
- Uniqueness (if appropriate)

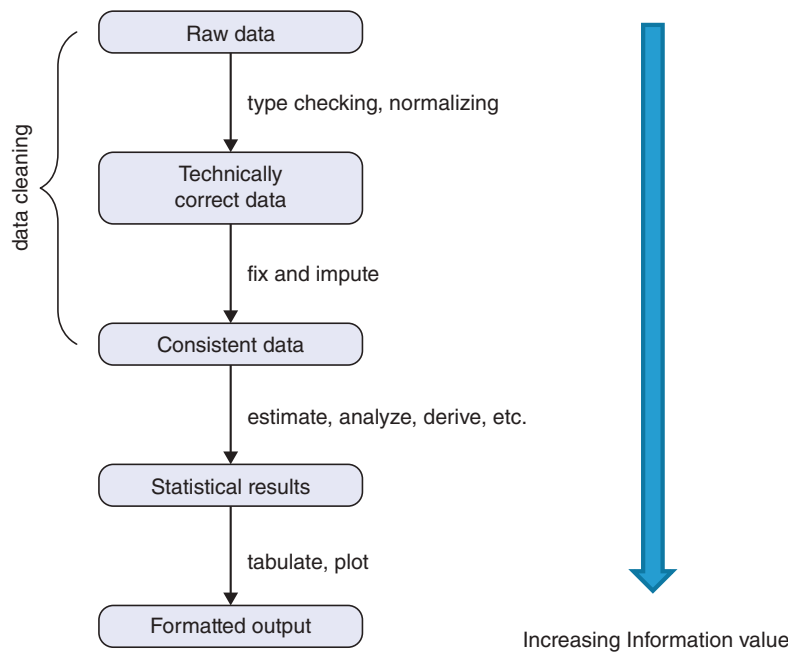


FIG. 8.1 Stages in the value chain of data cleaning operations for analytical purposes. Source: Modified after van der Loo, and de Jonge (2018).

- Check for appropriate codes
- Checks for arithmetic consistency

The process flow between the data understanding and data preparation phases in the CRISP-DM analytical methodology shown in Chapter 1 (Fig. 1.3) appears to be unidirectional; it is not. Many problems discovered during the data understanding phase might be corrected on the spot, but they belong logically in the data preparation phase. For this reason and for the sake of coordination, corrective measures for some problems will be discussed after the problems are described below.

Data type checking

The most important variable in the analytic file to check for data type is the target variable. Software packages will differ as to the exact terms used, but the thorough review in the previous chapter should help. If a classification model is planned, you must make sure that the target variable is typed as a category.

Three of the common type changes are (1) number to string, (2) string to number, and (3) nominal (category) to number.

String to number type change

Some algorithms accept only numerical values as predictor and/or target variables. For example, we will see in Chapter 11 that one method of feature selection is to

remove one member of a pair of highly correlated variables. Before correlation coefficients can be calculated for the many binary variables (1,0) that a data set might contain, a type change from string to number is required. KNIME provides a node, String to Number, to perform this operation. The node is configured to process a selected list of string variables. To build a classification model, the user must omit from that list the string variable to be used as a target variable, and all nominal variables with string contents (e.g., Marital_Status with contents like “Married”).

Number to string type change

If a classification data set with a binary target variable (with values of 0 or 1) is loaded from a comma-separated-values (CSV) file, the auto-typing routine in the data tool will set the data type to a number (probably an integer). After loading the input file, the data type must be changed from a number to a string. KNIME has node to do this operation also – the Number to String node.

Category to Number

Some nominal variables (with string contents) have one to several different string contents representing categories of information. Some analytical tools provide nodes like the KNIME Category to Number node, which can assign integer numbers of each category automatically for a list of variables. This operation can save a large amount of time processing each variable individually.

Why change string variables into numbers?

There are several reasons why numbers rather than strings are required in some analytical operations.

- Classification algorithms of most modeling tools accept only string target variables; a number won't work. This is a good example of performing a data engineering operation with the modeling algorithm requirements in mind.
- Parametric statistical modeling algorithms (e.g., Linear Regression) accept only numerical inputs; strings won't work. The reason is that the mathematical operations to find the best predictions must use calculations that require numbers (e.g., the mean and the standard deviation).
- Continuous numbers are required by Regression Tree algorithms (a form of decision tree to predict a number).
- Neural network algorithms require numerical data inputs. Some analytical tools permit the initial input of categorical variables, but they are converted internally to numbers before training begins.

Caveat with changing strings to numbers

The major drawback to changing strings to numbers is related to the way in which the numbers are used. Modeling algorithms will function as if there is a true value relationship inherent in the numerical list, when no such relationship might exist. For example, decision tree algorithms will optimize the cut-point along the numerical range of a variable to use in defining two different branches in the binary decision tree. This cut-point assumes that there is a relationship among values on one side of the cut-point that distinguishes them from the values on the other side. In most cases, no such relationship exists. This problem may increase the difficulty in interpreting results of data rows predicted by the trained decision tree algorithm.

Examples

For tools like KNIME, a categorical variable must be typed as a string, not an integer. For example, Table 8.1 shows the variable properties screen for the TARGET_B variable in KNIME, after the KDD-Cup 1998 dataset was imported from a CSV file. KNIME auto-typed the variable as an integer, because it contained only 1s and 0s. In order to use TARGET_B as a target variable in a classification model like a KNIME decision tree, the type must be retyped to string. In SPSS Modeler, a distinction is made between a flag (a binary) variable and a nominal variable (a string), but both will work in the Modeler classification algorithms.

Table 8.1 shows that the initial variable properties are "sensed" by the load process in KNIME, where the data

TABLE 8.1 A portion of the Donors2 input table showing that TARGET_B was input as an Integer.

Row ID	TARGET_B	TARGET_D	ODATEDW
1	0	0	8901
2	0	0	9401
3	0	0	9001
4	0	0	8701
5	0	0	8601
6	0	0	9401
7	0	0	8701
8	0	0	9401
9	0	0	8801
10	0	0	9401
11	0	0	9601

type is set by default to integer (I) rather than a string (S). This error required a change of data type from integer to string before the classification model can be trained.

Example 8.1 Data type changes

This example is described briefly as an example of data type changes. The complete KNIME workflow and directions document are available on the book Web page.

You may discover that there are type changes that must be made before modeling can begin. The TARGET_B variable is input as an Integer by the CSV Reader node in Example 8.1. We must change the data type to a string. Fortunately, there exists a specialized node designed for just this purpose – the Number to String node.

Steps:

- Read in input data with a Table Reader node
- Convert the data type from a number to a string with a Number to String node
- Save the output with a Table Writer node

The completed workflow is shown in Fig. 8.2.

Complete directions for completing Example KNIME 8.1 are contained on the book Web page.

Another problem can arise by loading a CSV file into an analytical tool. If the data tool that generated the CSV file does not enclose string values in quotes, the input data tool will interpret the incoming stream of digits (like Zip Code) as an integer and strip off any leading zeros. But the data type of the Zip Code value is not an integer data type, it is a *string of digits*, with leading zeros (e.g., 04101, a Zip Code in Portland, ME). To correct this input error, the data type of the Zip Code must be changed to a string data type, and leading zeros added to 4-character Zip Codes to produce the proper 5-character string of digits. Now the imported Zip Code can be used for mailing purposes.

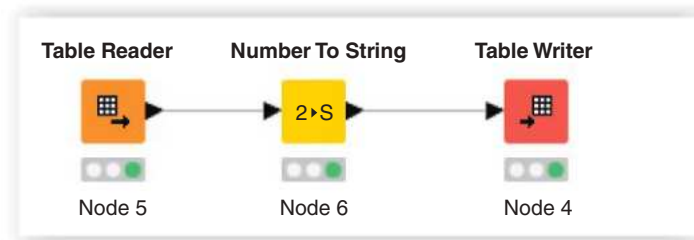


FIG. 8.2 The completed workflow for Example KNIME 8.1.

TABLE 8.2 Different data formats for Date and Dollars in four different source systems.

	Source 1	Source 2	Source 3	Source 4
Date	8901	1/1/1989	1-Jan-1989	1989/01/01
Dollars	\$100	100.00	0100.00	\$100.00

Data format checking

If a given data element in the analytical file is to be populated with values from different sources, the respective formats of the data in the source files must be consistent. For example, Table 8.2 shows some various formats for Date and Dollars that might be encountered in different sources.

Many data integration and data analytics packages can transform from and to many date formats. The only data format that might cause some problems is the date in Source 1. In this case, the data format checking will identify the need for some manual transformation of the date data. This task might be performed immediately, using capabilities in the tool package. Both the KNIME and IBM SPSS Modeler packages employ an internal expression language that can transform the Date values in Source 1 to a common date format. Failing to do so could possibly result in confusing month with day, as in May 12th and December 5th. Alternatively, the transformations can be done in Excel, rendering it as a collateral tool very popular with predictive analytics practitioners (for use with data sets with less than about 1 million rows and 256 columns).

Data code uniqueness checking

For analytical purposes, codes in a categorical variable must refer to unique states of the variable. Sometimes the meanings for codes will change. For example, the data sets used in this book contain a number of variables that express the recency of each previous donation. Codes in these variables are (1) C = Current, (2) L = Lapsed, (3) I = Inactive, and (4) D = Dormant. It might be the case in this data set that the distinction between Lapsed and Inactive might change as a result of some business

decision. This type of change is not uncommon in some data domains like banking, where customer relationships are treasured and managed closely. The change does not affect current reports, because current definitions apply to them. However, this change *does* affect the analysis of the historical data set, particularly if any of the variables will be used as predictors. In this case, the same logical state of a variable can be expressed in different parts of the historical record with different codes. This situation can add a significant amount of “noise” to the signal of any relationship that might exist between the predictor variable and the target variable. The analyst must determine when the code change was made and update the codes in the previous historical data accordingly.

Checking for appropriate codes

In addition to uniqueness, codes must be checked to see if they are appropriate for the variable. Data entry mistakes happen. Sometimes, negative signs are coded in a ZIP Code variable; these must be removed. Other times, ambiguous codes are entered (the “J” code in GENDER).

Table 8.3 shows two problems with the ZIP variable sorted in ascending order. Notice that there are many negative ZIP values and that some of the ZIP numbers have only four, rather than the standard five, digits. The negative ZIP values represent data entry errors, but the 4-character ZIP values are caused by the auto-typing feature of the KNIME file input nodes. KNIME assumes that ZIP values are integers and imports them as such, with no leading zeros. Zip Codes, however, may have one or two leading zeros. Therefore ZIP is actually a string (text) data type that is mistyped by the KNIME auto-typing feature upon input to a workflow, using either a File Reader node or a CSV Reader node. Table 8.1 shows both negative ZIPs and those with only four digits.

Example 8.2 Recoding

Workflow KNIME 8.2 shows the KNIME workflow for performing Example KNIME 8.2. The KNIME workflow and documentation is available on the book Web page. Fig. 8.3 shows the KNIME workflow used to correct the ZIP variable.

TABLE 8.3 A portion of the KDD-Cup 1998 competition data set, showing negative ZIP code numbers and some ZIPs with only four digits.

Row ID	TARGET_B	TARGET_D	ODATEDW	OSOURCE	TCODE	STATE	ZIP
12996	1	50	9501	FRC	0	AE	-9645
18610	0	0	9001	PV3	1	AE	-9182
6170	0	0	8901	DUR	0	CT	-6484
16578	0	0	9501	HAM	0	VT	-5663
11632	0	0	9601	CHT	0	ME	-4438
10681	0	0	8701	DRK	0	MA	-1754
7834	0	0	9501	ARG	0	MA	1876
6031	0	0	9401	MBC	0	MA	2038
2981	0	0	9501	AVN	0	MA	2124
12944	0	0	9201	SSS	1	ME	4078
368	0	0	9101	BHG	0	ME	4364
3296	0	0	9101	AML	1	VT	5033
7756	0	0	9601	HHN	1	CT	6335
16543	0	0	8601	DRK	2	CT	6460
6219	1	11	8601	DRK	2	CT	6468

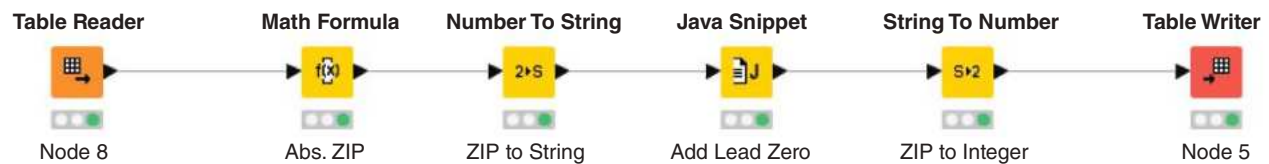


FIG. 8.3 The workflow for Example KNIME 8.2.

The primary steps in Example KNIME 8.2 include:

- Use of the Math Formula node to remove negative ZIP signs
- Use of the Number to String node to change the data type of ZIP
- Java Snippet node to add leading zeros to ZIP values with less than five characters

Validity checking for arithmetic correctness

Consider five fields in a table representing four quarterly income amounts and the fifth field holding the yearly total. A random check of the sum of the quarterly amounts will validate the yearly total. If mistakes are found, all of the values in the yearly total field should be recalculated with an expression formula.

Checking for other consistency problems

Other consistency problems might include:

- Text field misspellings: spellings should be corrected.
- “Orphan” codes (not used anymore) should be recoded for consistency.

- Occasional corrupted data in a given variable: corrupted codes should be recoded to some appropriate code.
- Missing values or corrupted codes in many variables appear anomalous in comparison with adjacent records. In this case, it is best often to just delete the entire row.
- Different numerical units must be standardized.
- Some alphabetic codes might be changed to numerical codes to be consistent with requirements by parametric statistical algorithms.
- Duplicate records should be removed to ensure that each case is unique.

String manipulation

String manipulation refers to the process of changing, parsing, matching, or analyzing strings. String manipulation can be performed in Python using the KNIME Python script node, or with the Column Expressions node in KNIME. Some common string manipulation operations include:

- Change case
- String length constraint (e.g., Zip Codes)

- Joining of strings
- Replace characters
- Remove characters
- Reverse strings
- Substring creation
- Data type changes (e.g., character numeral to integer)

Data filtering

In many cases, only data from within a specific range of values of a variable is needed for analysis. For example, analysis of data from teenaged persons requires a filter to be defined in a way similar to the logic in a KNIME Rule-based Row Filter node set to include all true matches:

```
$person_age$ >= 13 AND $person_age$
<= 19 => TRUE
```

Data filtering may be necessary when analyzing time-series data. And it is in the analysis of time-series data that we see the closest analogies to signal processing of transmission and image signals.

Time-series filtering

The practice of signal processing provides a series of data filtering methods that can be cross applied to data analytics. Signal filters remove high-frequency signal fluctuations (“jitter”), at either at the top of the range, the bottom of the range, or both. [Masters \(1995\)](#) describes concisely several types of signal processing filtering methods that can be adapted to limit the analysis to specific ranges of data prior to training predictive algorithms.

Low-pass filter

A low-pass filter passes data below a specified highest level of acceptability. For example, the higher part of the fluctuations in the 3-month T-bill rates might be eliminated from the data set.

A low-pass data filter can be implemented simply by eliminating cases with values below a certain threshold value. The effect of this operation will be to remove trivial inputs to the modeling algorithm. On the other hand, a high-pass filter can be used to remove cases above a threshold (outliers). One of the arts practiced in data mining is picking the right thresholds for these operations. To do this correctly, you must know the data domain very well. For example, data for telephone minutes of use each month (MOU) show that average call durations are about 3 minutes in length. But the curve tails off to the right for a long time! Back in the days of analog modems, connect times could be days long,

presumably because modems were left on and forgotten. Retention data sets using MOU as a predictor variable for attrition (“churn”) should be filtered with a high-pass filter to remove these outliers. The resulting model will be a much better predictor, just like a radio signal passed through a digital high-pass filter in a radio set will be much clearer to the hearer.

High-pass filter

A high-pass filter does the opposite of a low-pass filter; it passes data above a specified lower level of acceptability.

Band-pass filter

Finally, the band-pass filter passes only data above a low value and below a high value.

At this point, you might be wondering why we emphasize time-series data processing, when most data sets we must prepare for modeling *not* time-series! The reason is that signal processing techniques can be applied very effectively to data sets other than time-series data. We can use these techniques to analyze time-series data from billing systems and purchase records also. When you view a predictor variable in the context of its contribution in predicting a target variable, you can think of that contribution as a “signal” of the state or level of the target variable. Some variables might provide a stronger signal (be more predictive), and other variables may be less so. There will be a certain amount of “noise” (confusion in the target signal) in the values of the predictor variable. We can selectively remove some of this noise in ways very analogous to time-series signal filtering. The basis for our data filtering operations is well grounded in engineering theory.

Outlier handling

Values of a given variable may appear to be very far from the mean; nevertheless, they can still be part of the natural data distribution. Our task in predictive analytics, however, is to train a model based on usual data values and not based on unusual values. The resulting model will be a much more accurate predictor of the usual values than a model trained on a data set with unusual values in it. In this regard, this effect is like separating the “signal” of a radar ping from the surrounding noise in the input data. Outliers are one cause of that “noise” in the process of training a machine learning model. In deployment, the model may be very predictive of the target variable in a record (row) when usual values are present in all of the predictor variables,

but perform much poorer when outliers exist in one or more of the predictor variables. The failure of the model on a few records is acceptable when the model performs relatively well on records with usual values in the predictor variables.

KNIME provides the Outlier Removal node, which will remove all records in which a given variable has outlier values as defined in the node configuration. Some values may be so far from the mean, however, that they are more likely to be the result of input error than to be part of the natural data distribution. The problem with using this node is that it removes the entire record that has outlier values for a specified variable. Another way to handle outliers is to recode them with a specified value, either the highest allowable value, the mean value (the average value), the median value (the middle value), or the mode value (most common value).

What are outliers?

Data in many databases are stored often in forms and formats that may obscure the pattern we want to analyze with our analytical tools. Sometimes all we have to do is modify certain information items (fields) in a specific table in the database. One of the most common problems of this nature is the presence of a few values that do not fit the pattern of the vast majority of the data element. These are called *outliers*.

Hawkins (1980) defines an outlier as “...an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.” Hawkins discusses four kinds of outlier detection algorithms:

- Those based on critical distance measures
- Those based on density measures
- Those based on projection characteristics
- Those based on data distribution characteristics

Some data mining packages have special routines for identification of outliers. For example, Statistica Data Miner provides a Recipe module, which permits automatic checking and removal of outliers beyond a given range of value or proportion of the frequency distribution (in this case, 95%). Invoking the distributional outlier option in this tool will trim off cases in the tails beyond the 95% confidence interval for distance from the mean value.

Why worry about outliers?

In predictive modeling, our challenge is to find patterns in the input data set that can serve as a basis for making predictions. The data patterns we define don't have to account for all of the input data, just the vast

majority of the data. A clear pattern that permits the correct prediction of a target variable 90% of the time may fail to predict properly a small number of extreme values – outliers. It is better to handle these outliers before training a model rather than to allow the extreme values to unrepresentatively affect the form of the model.

Ways to handle outliers

- List-wise deletion
- Row-wise deletion
- Recoding
- Filtering
- Imputation of outliers

List-wise deletion

The simplest way to handle outliers is to remove the variables that contain outliers (list-wise deletion).

Row-wise deletion

If more than several variables have outliers in them, another option is to delete entirely the rows that contain them (row-wise deletion).

Sometimes you want to keep the outliers (abnormal values). In fact, some outliers are of primary interest to the modeling of credit risk, fraud, and other rare events like network intrusions. For models of “normal” responses, it might be a good idea to remove the extreme outliers by deletion of the row or imputation of the value with a constant or the mean or median. The reason for this is that you want to model the data which help to define the normal response. If you leave the outliers in the data set, they will just inject noise, which will reduce the predictability of the model. But you might object that we should keep all values in the data, because we have to score values like this in our production operations of the model. Well, yes, you have to score outliers, but you can afford to be wrong in your predictions 5% of the time (for example) for the sake of being very predictive on the other 95% of the data.

But how do you decide what is an outlier and what is not? Chapter 1 will discuss some of the properties of the Normal distribution and provide us with a technique to help us define outliers – the Z-value. In the Z-distribution, 99.7% of the values are between -3 and +3 Z-units from the mean value of a variable. We can choose to classify a data value as an outlier if its Z-value is outside of the -3 to +3 range. Our model of the signal in the data that are left will be more predictive for those data points within the range of exclusion of outliers. Yes, we may have to accept a few prediction errors at the extreme of the input data distribution, but the vast majority of the predictions will be stronger than if we left the outliers in the modeling data set.

TABLE 8.4 HIT values sorted in descending order to show 14 rows with a value of 240.

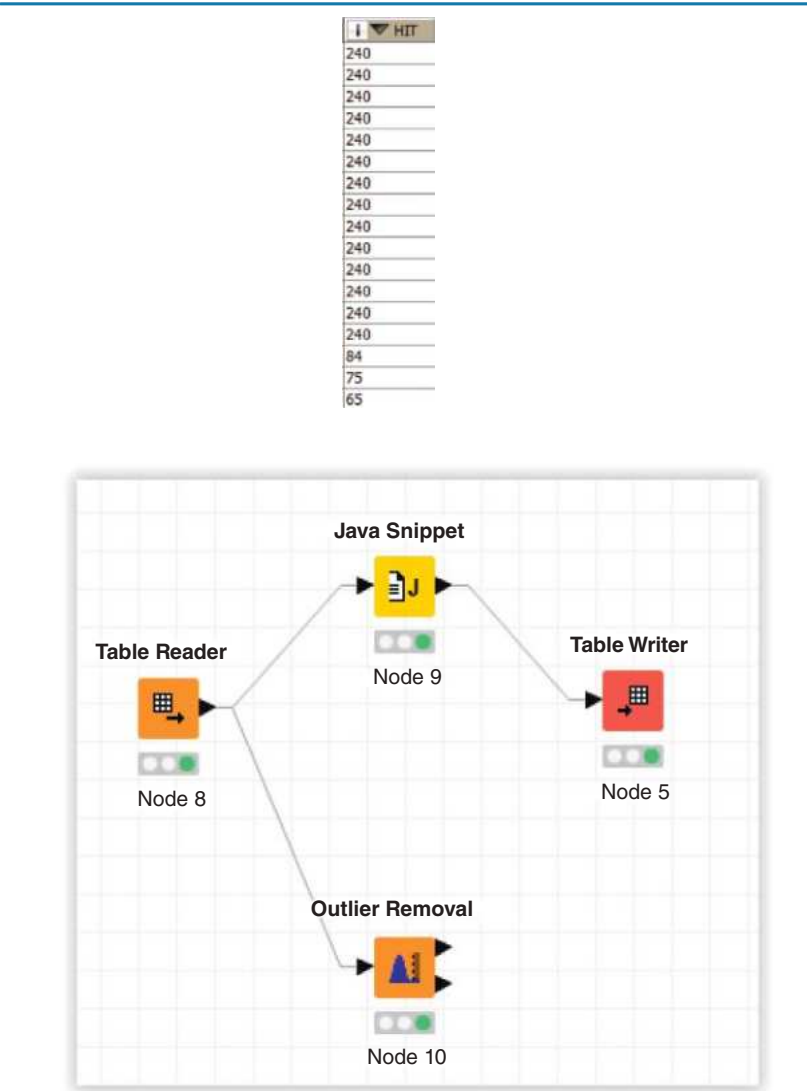


FIG. 8.4 A KNIME workflow showing two procedures for handling outliers.

Recoding outliers

Another way to handle outliers is to recode the extreme values to the next highest value. Table 8.4 shows original values for HIT in the KDD 1998 PVA data set, sorted in descending order.

Table 8.4 shows that the HIT variable in our original modeling data set has 14 rows with a value of 240, where the next highest value in the data set is 84. Those very high data values are very suspicious, because they are much higher than 84, and because they are all identical. It is very probable that these 14 values of HIT are data entry errors. We should not use these error values to define the signal of the response. We could recode the 14 values with the mean HIT value, or we could use the next highest value (84) to represent the true values.

The decision of which strategy to follow depends on the business domain knowledge and expertise of the analyst.

Example 8.3 Handling outliers

Fig. 8.4 shows the KNIME workflow that performs several outlier handling operations.

Primary Operations in the KNIME workflow shown in Fig. 8.4 for outlier handling are:

- 1. Reading in the input data with a Table Reader node
- 2. Using the Outlier Removal node for removing records with outliers in a given variable, HIT
- 3. Using the Java Snippet node to recode outlier values in HIT to a specified value

Complete directions for completing Example 8.3 are contained on the book Web page.

Postscript

This chapter focused on those operations that may be necessary to solve problems associated with the data set that would prevent the modeling algorithm from building a model to answer a given question. As in any task to request information, our initial task is to frame the question properly. No source of information can be expected to read our minds; we must state clearly and concisely the request for information before we can expect an appropriate answer. Data cleaning is the first step in preparing our data set patterns to be perceived by the modeling algorithm to permit it to answer the question properly. In [Chapter 9](#), we will discuss the second step in this process, various data

conditioning operations necessary to support the data format and computational requirements of the chosen algorithm.

Keywords and phrases

Signal, Outliers, Filtering, Correlation, Information, Pattern, Noise, Classification model, Type changes, Time-series data.

References

- Hawkins, D. (1980). *Identification of outliers* (pp. 188). Boston, MA: Chapman and Hall, Kluwer Academic Publ.
- Masters, T. (1995). *Neural, novel & hybrid algorithms for time-series predictions* (pp. 514). New York: John Wiley & Sons.
- Van der Loo, E., De Jonge, E. (2018). *Statistical data cleaning with applications in R* (pp. 320). New York: John Wiley and Sons, Inc. ISBN-13: 978-1118897157.

Data conditioning

Preamble

All of the data preparation operations discussed in previous chapters involved modifications to rows or columns of data. Data conditioning operations perform some column-related operations (e.g., filling of missing values in specific rows), but they focus more on modifications to the entire data set to prepare it for modeling. These data conditioning operations are often omitted, which may affect the predictive accuracy significantly. The best approach is to investigate your data set to determine if any or all of these data conditioning operations are necessary.

Data conditioning operations that generate consistent data sets

Data conditioning means to transform existing columns and rows to be consistent with the input requirements of various analytical operations. If we submit data to a modeling algorithm that violates its assumptions regarding the range of relative numbers of examples for each target category, the algorithm will perform, but it will perform much poorer than it would if those assumptions were met.

Three types of data conditioning operations are discussed in this chapter

1. Standardization and normalization
2. Balancing of data sets with imbalanced target categories
3. Filling of missing values

Standardization and normalization

Standardization and normalization are techniques used to rescale all columns (variables or features) in a data set to eliminate any bias to algorithm function caused by large differences in data ranges.

Normalization sets an upper limit to the range of a variable (e.g., 1.0) and rescales all values proportionately within that range. Normalization makes the range of values in each column to be equal.

Standardization divides each value in the column by the standard deviation of the column. The range of values associated with each column is a function of the standard deviation. Standardized values work better in algorithms based on the standard deviation value (e.g., Linear Regression).

Why standardize or normalize variable ranges?

Parametric statistical algorithms (e.g., Linear Regression and Logistic Regression) estimate the coefficient of each predictor variable with a process that uses the Gradient Descent technique in the error minimization process. Variables with much larger ranges will bias the optimization procedure in favor of variables with relatively large ranges compared to those with much lower ranges. The other reason is practical. Standardization removes from the coefficient assigned to each predictor variable any differential effects of scale such that the coefficient values represent the relative contributions of each predictor variable in the determination of the prediction. Thus the coefficients serve as expressions of variable importance in the solution.

Machine learning (ML) algorithms that use the Euclidian distance between variables in a multidimensional space (e.g., K-means) will be biased in the calculation of similarity values by variables with relatively large ranges. Decision tree algorithms and their ensembles (e.g., Random Forest) also use some procedure (usually Gradient Descent) to optimize the cut-point along a variable range in building a branch of the decision tree. Range bias will affect the cut-point optimization also.

A general rule of thumb of statisticians and data scientists is to standardize before using these algorithms:

- Algorithms that use the Gradient Descent optimization technique
 - Linear Regression
 - Logistic Regression
 - Neural Networks
 - Many Decision Tree algorithms

- Algorithms that use distance between points to determine similarity
 - Kohhonen Neural Network
 - Support Vector Machine
 - K-means clustering

Decision Tree algorithms (like Random Forest) are relatively insensitive to differences in ranges among predictor variables. Random Forest and other ML algorithms are sensitive, however, to significant differences in the proportions of examples of the target variable classes. Next, we will consider the dynamics of the performances of the Random Forest and Neural Network algorithms related to the degree of imbalance in the proportion of the target classes.

Balancing of data sets with imbalanced target categories

Parametric statistical modeling algorithms (e.g., Multiple Linear Regression and Logistic Regression) are based on calculation of the average (the mean) value and the standard deviation (based on the mean difference between a data value for a variable and its mean value) for all variables in the data set. Actual deviations from the mean are calculated and squared (to remove any minus signs) for each variable in each row of the data set to produce a cloud of squared deviations. The algorithms attempt to find the line of best fit through that cloud of data points. The function of the parametric statistical algorithms is not sensitive to the relative frequency of the target classes. This is *not* the case, however, with ML algorithms.

ML algorithms seek to build a pattern in the data set for each target class. If there are more rows of the positive target class than for the negative class, the data pattern for the positive class will be defined better than for the other class. The same problem occurs with multiple target classes. This situation causes a bias in the classification operations in favor of the target class with the highest frequency in the data set. Imbalance in data sets has been reported that range up to 100,000 to one (Fawcett and Provost, 1996).

Two general ways have been used to address this issue of class imbalance: (1) assign difference costs to the target classes of each training example (Pazzani et al., 1994; Domingos, 1999), and (2) by either undersampling the majority class or oversampling the minority class (Chawla et al., 2002). The form of oversampling employed by Chawla et al. (2002) was named Synthetic Minority Oversampling Technique (SMOTE).

Consider three popular classification algorithms: Decision Trees, Random Forests (RF), and Neural Networks (NN). Consider also the Adult data in the UCI Machine Learning Repository. This data set consists of a

target variable (class) and 14 predictor variables, of which 8 variables are categorical. Initial data preparation consisted of generating binary “dummy” variables for all classes in the categorical variables. The target variable (class) is the median income per year coded as ≤ 50 K (22,801 rows) or > 50 K (7199 rows). This data set is highly imbalanced (76% vs. 24%). This imbalance could cause the classification algorithm to be biased strongly in favor of the data pattern of the majority class (≤ 50 K). The number of the minority target class might be sufficient to define the minority data pattern, in which case there might be no effect of the imbalance.

We can evaluate the effectiveness of three balancing techniques (undersampling, manual oversampling, and SMOTE) to train models of the Adult median income data set, available in the UC-Irvine Machine Learning Repository.

Using the imbalanced data, results of the first model are shown in Table 9.1.

The overall prediction accuracies of all three algorithms appear to be relatively high, but the sensitivity accuracies are quite low. It can be inferred from these results that all three algorithms do a better job predicting the ≤ 50 K target class (reflected by the relatively high specificity accuracy) compared to the > 50 K target class (reflected by the relatively low sensitivity accuracy).

There are three common strategies employed in previous studies for removing this prediction bias before building the final classification model.

- Undersampling the common class to equal the rare class;
- Oversampling the rare class to equal the common class;
- Simple Minority Oversampling Technique (SMOTE).

Undersampling the common class to equal the rare class

Many analytical tools have routines that perform an undersampling operation of the common class until the total sampled number equals the total rare class number. This approach does generate a balanced data set and is useful for preliminary work, but it deletes a large number of rows that would be used otherwise to define the

TABLE 9.1 Performance accuracies of the Decision Tree model vs. the Random Forest model with the imbalanced data set.

Algorithm	Sensitivity accuracy	Specificity accuracy	Overall accuracy	AUC
Decision Tree	0.590	0.885	0.814	0.771
Random Forest	0.602	0.938	0.857	0.890
Neural Network	0.599	0.925	0.847	0.899

TABLE 9.2 Performance accuracies of three algorithms using a manually undersampled data set.

Algorithm	Sensitivity accuracy	Specificity accuracy	Overall accuracy	AUC
Decision Tree	0.775	0.753	0.758	0.787
Random Forest	0.845	0.804	0.814	0.901
Neural Network	0.856	0.771	0.792	0.895

TABLE 9.3 Performance accuracies of three algorithms using a manually oversampled data set.

Algorithm	Sensitivity accuracy	Specificity accuracy	Overall accuracy	AUC
Decision Tree	0.605	0.872	0.808	0.738
Random Forest	0.827	0.817	0.820	0.892
Neural Network	0.599	0.925	0.847	0.899

data pattern of the common target class. Using this undersampled balanced data set, results of the second model are shown in [Table 9.2](#).

Oversampling the rare target class number to equal the common target class number

Oversampling involves duplication or synthesis of data, not deletion of data – no information is lost. This process can be performed by duplicating enough rows of the rare target class to equal the number of rows of the common target class (manual oversampling). Alternatively, the additional rows can be synthesized (see SMOTE below). [Table 9.3](#) shows the much-improved results of a manual duplication operation using oversampling of the rare target class to generate equal numbers of both target classes.

Synthetic minority oversampling technique (SMOTE)

The SMOTE technique was introduced by Chawla et al. in 2002. SMOTE has become a common balancing technique to apply to imbalanced data sets in many data domains. SMOTE uses a Nearest-Neighbor analysis to synthesize values for each variable in enough rows of the minority target class to equal the number of rows in the common target class. [Table 9.4](#) shows the comparative performance of 3 algorithms trained with the SMOTE technique: Decision Tree, Random Forest, and Neural Network.

TABLE 9.4 Performance accuracies of three algorithms using the SMOTE technique to balance the data set.

Algorithm	Sensitivity accuracy	Specificity accuracy	Overall accuracy	AUC
Decision Tree	0.648	0.854	0.804	0.783
Random Forest	0.751	0.858	0.830	0.889
Neural Network	0.817	0.804	0.807	0.894

Discussion of the effects of balancing on model performance

The SMOTE has become very popular as a balancing technique recently. However, there is some recent evidence that SMOTE is not always the best way to balance imbalanced data sets. For example, [Mishra \(2017\)](#) found that in some data sets, random undersampled models were more accurate than models balanced with SMOTE, while for other data sets, SMOTE won.

Using the Adult median income data set, it appears that the undersampled models were consistently more accurate for all three algorithms with all three balancing techniques. We see therefore that the effectiveness of a given balancing technique may depend on the data set and the modeling algorithm used.

Filling of missing values

Almost every data set has missing data (MD) in it. To the database manager, missing data is data all by itself; the missingness might be indicative of some pattern, or it might have been caused by some random error. The presence of missing is not a problem for the database manager, but it can be huge problem for the data analyst. This chapter discusses some of the causes of missing data and several ways to fix this problem.

Why are missing data important?

In parametric statistical analysis (e.g., Regression), the condition of missing data in a variable is not a computational problem. The mean and standard deviation of each variable is calculated from whatever data are available for each variable in the data set. Subsequent mathematical operations are performed using the mean and the standard deviation; available raw data values are used only for error calculation purposes subsequently. This is not the case with ML algorithms. ML algorithms will fail if there is even one missing value in a variable among all records in the data set. To prevent failure, most ML algorithms will perform a deletion operation by default, either to delete the entire row when it encounters a missing value in a variable in the row (List-wise deletion), or to delete the variable with missing values

(Pair-wise deletion). If List-wise deletion is selected to handle missing values, it may reduce the number of rows in a data set prior to analysis to such an extent that the rows left no longer represent the original pattern in the target variable appropriately. It can even result in no rows being passed to the algorithm at all. But the massive reduction of rows is perhaps more ominous because at first, the rookie analyst might not notice that the model was based on 10% of the data. Therefore many predictive analytics applications provide several methods for handling missing values. If the Pair-wise deletion method is selected for default processing, it might delete a variable that could serve as an important variable in the ensuing models. If ML algorithms will be used for modeling, it is preferable therefore to handle missing values directly.

For the purposes of this book, we will assume that the analyst plans to use an ML algorithm. Therefore the analyst must determine the appropriate methods of handling missing data for each variable. But before launching into a discussion of missing data handling techniques, we have to consider (1) the consequences of missing data and (2) the kinds of “missingness” that exist in the data set – the choice of methods for handling missing data depend on the kind of missingness.

The consequence of missing data

McKnight et al. (2008) cited three general consequences of missing data: (1) *construct validity*; (2) *internal validity*; and (3) *causal generalization*. The choice of what codes should be used to impute missing data must be guided by considerations of errors that might be caused by ignoring these consequences. For the purposes of this book, we will adopt McKnight’s classification of missing value-handling operations.

Consequences of missing data on construct validity

A construct is a derived measure composed of several raw data elements. For example, values of Income, Education, Homeowner, and Age might be combined to define a number of bins in a new derived variable, Socio_Economic_Status. It is reasonable that this derived variable might be more predictive of the outcome of a model than any of the component raw variables. If there are missing data in any of the component elements of this new variable, they might compromise the validity of which bin to use to characterize an individual. The validity of any relationship between the combined effects of the three raw data elements may be affected by error caused by imputing the wrong values for the missing data. The risk of this error is greatest when single measures are used to represent a complex response. For example, the variable Smoker might be characterized as one who uses cigarettes, while other forms of smoking are possible (cigars and pipes). If there are missing data

in the smoker variable, they might be imputed with a code for nonsmoker. This imputation might be wrong if the person smokes a pipe. In this case, the safest imputation for missing data in the Smoker variable would be “U” for unknown, not “N” for nonsmoker.

Construct validity errors can be minimized by using multiple expressions of the construct, composed of different raw variables. For example, the construct Rural might be defined in terms of population density or Zip Code. McKnight et al. (2008) note that an inherent problem with multiple expressions of a construct is when there is a systematic reason for the missing data in the construct. Consider that case when missing data for the variable Drug_User may be caused by the failure of drug users to contribute data compared with nondrug users.

Consequences of missing data on internal validity

This effect of missing data concerns the validity of relationship involving the variable with missing data and any other variable, including the target variable. Missing data may either increase or decrease the strength of this relationship significantly, which in turn might generate some spurious inferences. The effect of missing data in a variable upon its relationship with the target variable is a direct effect, but there may be many indirect effects. In some situations, the very missingness of data in a variable is significant. Missing data can affect the strength of interaction of a predictor variable with some other predictor variable in a model. These interactions can be important sources of prediction of the target variable. Some modeling software packages provide interaction analysis capabilities, which can generate interaction variables at various levels. One of these packages (STATISTICA Data Miner) can generate interaction variables with up to four factors. All elements of interaction variables used in a predictive model should be analyzed to determine the best method for handling missing values in them. For some factors with much missing data (>50%?), deletion of the variable might be best. For other factors, various methods of missing data imputation might be used, depending on the nature of the missing data (see below). After any factors are deleted and missing values are filled for other factors, the interaction analysis should be rerun.

With that short introduction to the consequences of missing data, we are faced with the problems of what to do about them. But before we can decide which procedures to follow for handling missing data, we must know the nature of the “missingness” among the data elements in our data sets.

Classification of missing data types

Since 1976, most statisticians have referred to the classic paper by Rubin (1976), which describes several types of missing data. Based on that early work, later authors

have defined three categories of missing data: (1) MCAR – missing completely at random; (2) MAR – missing at random; and (3) MNAR – missing not at random (Allison, 2002; McKnight et al. 2008). Those terms are somewhat confusing at first glance, because the last two categories imply that there are several levels of missing at random, depending on what the basis is for determining the randomness. The choice of among procedures for handling missing values depends on the nature of the missing data category.

MCAR

For data to be MCAR, the incidence of missing data is not related to any other value in the data field itself or to values in any other data field. For example, missing data in the Age field is not related to the value of age, nor are they related to values in any other field, such as Gender.

MAR

For data to be missing at random, it is not related to the values in the Age field, for example, but they may be related to other data elements, like Gender.

MNAR

For data to be missing not at random, there must be some function or mechanism that causes data to be missing, such as the propensity for older people to fail to report their ages. In this case, the mechanism is forgetfulness.

Diagnosis of the nature of missing values

A common strategy to follow in the process of classifying missing data is to derive “missingness” variables for each variable that has missing data. A missingness variable is coded with the value 1 if a missing value exists for a given record, and it is coded with the value 0 if a data point is not missing for that record. The missingness variable reflects the pattern of “missingness” among the records for a given category in the variable and can be used to look for relationships between this pattern and any other variable. Consider the variable Test1 with missing values in Table 9.5. The value of 1 is entered for a given row in the new variable Test1_Missing if the value for that row is missing in Test1; otherwise, a 0 is entered.

Classification of missing data as MNAR

If a relationship is found between the dummy variable INCOME_MISS and its source INCOME, the missing data in INCOME can be classified as MNAR (missing not at random). For example, higher values of INCOME might tend to be missing, in which case there is a mechanism related to INCOME responsible for the missingness in INCOME. Proper filling of MNAR data requires

TABLE 9.5 Missingness coding of test scores.

Test1	Test1_Missing
26	0
28	0
–	1
32	0
21	0
–	1

that the mechanism be understood so that appropriate values can be selected to fill missing values. For example, if Gender data is missing for some people in an area known for the presence of numbers of transgender missing data, it may be caused by the fact that a transgender person could be classified either as a Male or as a Female. The best way to prevent MNAR data in Gender is to prevent it from happening by installing additional categories. In European documents, a third category, Other, is used. In some locations with a lot of gender diversity, several additional categories might be used. For example, the Sex and Gender Guidelines of the City and County of San Francisco include five categories (City and County of San Francisco, 2013).

- Male
- Female
- Transgender Male
- Transgender Female
- Not listed

Classification as MAR

A relationship might exist between missing data in the INCOME variable and some other variables, such as the CREDIT_CUST as candidate predictor variables. To test this hypothesis, a simple correlation analysis can be run between the values in INCOME_MISS variable and the candidate predictor variables. A correlation analysis of the missingness of the two variables suggests that there are two different mechanisms controlling the missingness.

KNIME examples for filling missing values

The following three examples for filling of missing values are described briefly in the text, but complete directions for completing the example are available on the book Web page. In Example 9.1, we will fill missing values with constants in all three major types of input variables (Integers, numbers, and strings) using the Missing Value node. We will exclude the GENDER and INCOME variables, however, because we will fill missing values in GENDER by recoding them with a logical script in a Java Snippet node in Example 9.2, and we will fill missing values in INCOME with a model in Example 9.3.

Example 9.1**Steps**

- Read in the input data source
- Filter out inappropriate variables
- Use the Missing Value node to fill all missing values except in GENDER and INCOME

Your final KNIME workflow for Example 9.1 should look like [Fig. 9.1](#).

The input data and detailed configuration of each node are available on the book web page.

This example appears to be rather simplistic, but the power of the Missing Value node is very great. It can specify the imputation value for missing values in strings, integers, and decimal numbers automatically. A data set may have hundreds or even thousands of variables (as some pharmaceutical data sets have). The KNIME Missing Value node can fill each of the variables in the data set automatically. Alternatively, the user can specify different imputation actions for individual variables in another dialogue screen.

Using correlation analysis to prepare for imputing missing data that are MAR

The previous analysis gives us a lot of information to guide our choice of the missing data handling method to use for INCOME. We could choose to run a regression between INCOME and predictor variables WEALTH1, NUMCHLD, LIFESRCE, DATASRC, and AGE. Example 9.2 will perform a Linear Correlation operation to prepare data for using polynomial regression analysis to impute missing values in INCOME. Polynomial regressions are curvilinear in nature, rather than linear, which is more appropriate for business data.

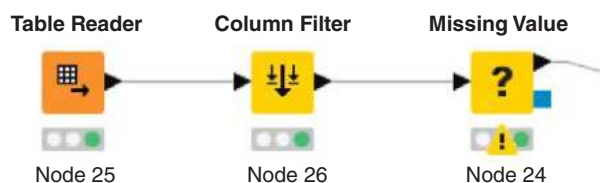


FIG. 9.1 Completed workflow for Example 9.1.

KNIME Example 9.2 – Using Linear Correlation to prepare MAR missing data to be imputed

In this example, we will use a Polynomial Regression algorithm to impute missing data in INCOME. Polynomial regressions are curvilinear in nature, rather than linear, which is more appropriate for business data.

Steps in this Example:

1. Use a Java Snippet node to recode GENDER
2. Perform a linear correlation analysis to identify potential predictors for INCOME
3. The final workflow screen should look like [Fig. 9.2](#).

The input data and detailed configuration of each node are available on the book web page.

KNIME Example 9.3 – Using regression to fill missing data that are MAR

Example 9.2 provided us with some valuable information to use for further processing of the INCOME variable. We will use a Polynomial Regression algorithm to impute missing data in INCOME. Polynomial regressions are curvilinear in nature rather than linear, which is more appropriate for business data.

Steps in this Example:

- Use a Normalizer node to normalize values in all numeric variables to a Z-score
- Use the candidate predictor variables identified in Example 9.2 and train a Polynomial Regression algorithm to impute missing values in INCOME:
 - If INCOME = > 0, output the original data
 - If INCOME = 0 (former missing value), output the imputed value

The final workflow screen will look like [Fig. 9.3](#).

Classification as MCAR

The most well-known test for determining whether or not data are MCAR is Little's test. This algorithm is simple in concept, but it requires some careful data processing to run it. Some analytical tools provide Little's test (e.g., SPSS, R, and Python). If your tool package does provide Little's test, use it initially to classify your missing data.

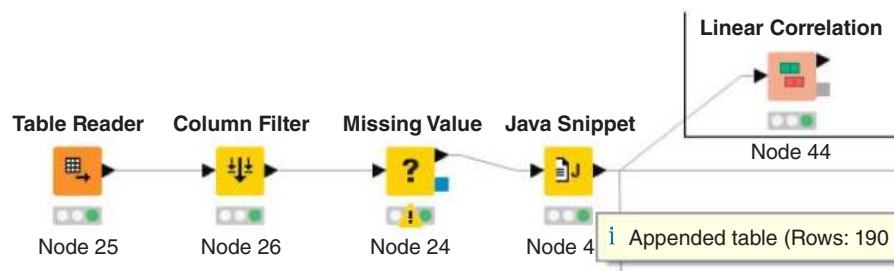


FIG. 9.2 The workflow showing the use of a Java Snippet node for recoding and a Linear Correlation node for correlation analysis.

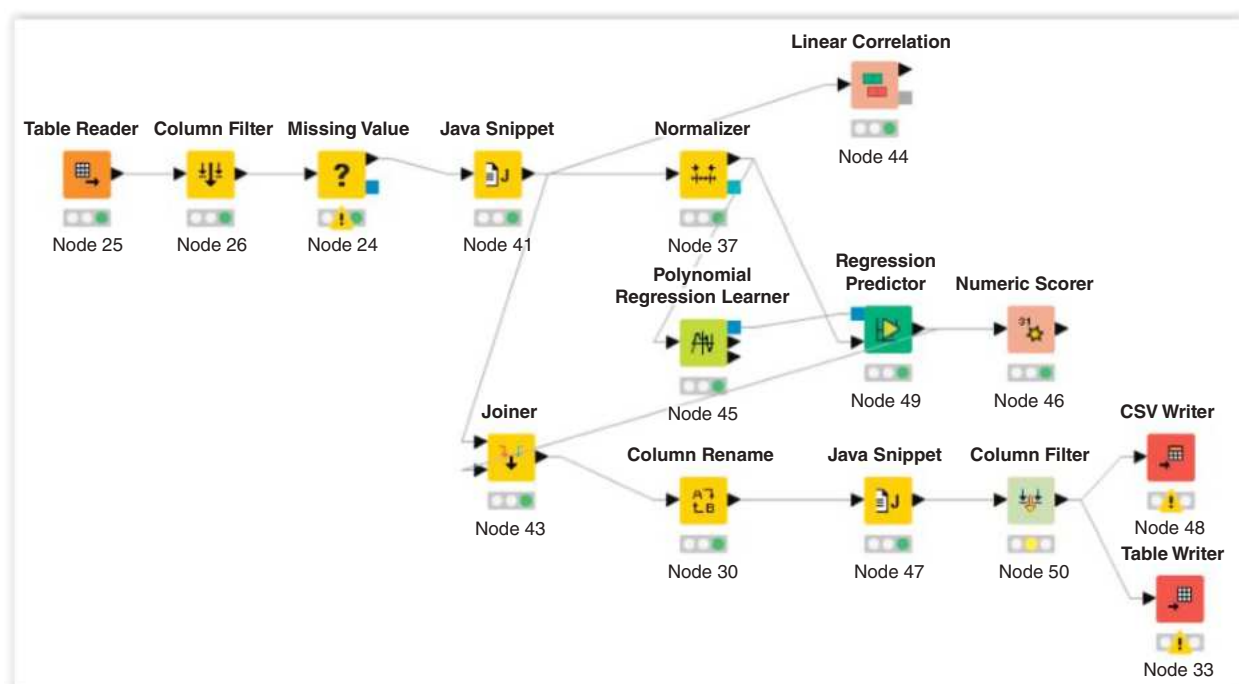


FIG. 9.3 KNIME workflow showing the use of a Polynomial Regression algorithm to impute missing values in the INCOME variable.

If you have used Little's test for MCAR or have reason to believe that your missing data are not MNAR nor MAR (according to the previous discussion), numerous strategies are available for filling missing data that are MCAR.

1. List-wise deletion-Deleting the entire record
2. Pair-wise deletion-Deleting the variable
3. Imputing a mean or a median for numeric data
4. Imputing the most common categorical value
5. Imputing a category for "Other" or "Unknown"
6. Imputing the value with Nearest-Neighbor analysis

Postscript

Now that the original data set is prepared properly for modeling, we can refer to the modified variables as "features": it is time to do the interesting stuff – Feature engineering! Some of the most predictive features in many ML models are those that the modeler derives from the input data. Chapter 10 will describe a number of feature derivation strategies to build the final data set to input to the modeling algorithms.

Keywords and phrases

Imputation, Regression, Undersampling, Oversampling, SMOTE, Imbalanced, List-wise deletion, Row-wise deletion, Missing at random, Minority case, Bias.

References

- Allison, P.D., 2002. Missing data. Sage Publications.
- Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W., 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357.
- City and County of San Francisco. 2013. Sex and gender guidelines. <http://www.sfhiv.org/wp-content/uploads/081513-SFDPH-Sex-and-Gender-Guidelines.pdf>
- Domingos, P., 1999. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 155–164. ACM Press.
- Fawcett, T., Provost, F., 1996. Combining data mining and machine learning for effective user profile. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 8–13. AAAI.
- McKnight, P. E., McKnight, K., Sidani, S., Figuerdo, A. J. (2008). *Missing data: A gentle introduction* (pp. 251). New York, NY: Guilford Press.
- Mishra, S., 2017. Handling Imbalanced data: SMOTE vs. random under-sampling. *International Research Journal of Engineering and Technology* 4 (8), 317–320.
- Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., Brunk, C., 1994. Reducing misclassification costs. In *Proceedings of the Eleventh International Conference on Machine Learning*. Morgan Kaufmann.
- Rubin, D.B., 1976. Inference and missing data. *Biometrika* 63 (3), 581–592.

Data and feature engineering

Prelude

It is almost never the case in most data domains that all columns (aka fields in a database row) contain information in the proper format or that can be used individually as predictors in a modeling algorithm. In a modeling algorithm, these database fields are called “variables.” When components or combinations of those variables transformed to create new variables, they often are called “features.” That subset of those features and original variables used by the modeling algorithm to train a model is called “predictors.”

Before we can consider the contribution of each feature engineering activity to the modeling effort, we must define some terms. These terms will be used throughout this book to refer to these entities.

Definitions

- **Source data:** information from any source in any format
- **Analytical file:** a set of information items from (possibly) multiple sources; that information is composed into one row of information about some entity (e.g., a customer)
- **Record (aka Case):** one row in the analytical file
- **Attribute:** an item of data that describes the record in some way
- **Variable:** an attribute installed into a column (field) of the entity record
- **Target variable:** a variable in the entity record to be predicted by the model
- **Predictor variable:** a variable in the entity record that is a candidate for inclusion in the model as a predictor of the target variable
- **Numeric variable:** a variable with only numbers in it; it is treated as a number
 - Continuous numeric variable: a number with decimal point in it, either real or implied
 - Discrete numeric variable: an integer without a decimal point, real or implied

- **Categorical variable:** a variable with any character in it; the character may be a number, but it is treated as text
- **Feature:** a modified variable
- **Dummy variable:** a feature created for each member of the list of possible contents of a categorical variable (e.g., “red,” “green,” “blue”)
- **Surrogate variable:** a variable that has an effect on the target variable very similar to that of another variable in the record. We will discuss this variable type in [Chapter 11](#) – Feature selection.

Many practitioners of predictive analytics agree that some of the most powerful predictors are those features derived by the analyst from components or combinations of existing variables. Various strategies for deriving new features will be discussed, evaluated, and illustrated in this chapter.

Topics of discussion in Chapter 10

1. Why Data Engineering?
2. Data type changes
3. Data Lookups
4. Variable (column) filtering
5. Row filtering (selection)
6. Grouping and Aggregation
7. Feature Transformation
8. Feature Extraction.
9. Feature Derivation
10. Data Abstractions
11. Temporal Data Mining
12. Discretization

Why data engineering?

Previous names for the topics listed above include data preparation, data preprocessing, and data transformation. These activities are performed with the data as the primary focus of interest. Rather than focus on the actions themselves, it is more effective to focus on the end result – a data set that includes patterns with a

composition and format useful by the modeling algorithm to build a predictive model. The same approach is used in manufacturing an airplane. The airplane must be able to fly and perform specific tasks (e.g., serve as a transport or as an attack vehicle). In the case of data to be used with a predictor algorithm, we must design our data processing activities to permit the algorithm to build the most powerful predictive model possible. That approach is essentially an engineering process.

Data type changes

Often, data elements (e.g., variables) are input in the wrong format to support an intended operation. Users must change the data type to one compatible with a given operation. Sometimes a given data element must be changed several times to serve in various operations.

Data lookups

Looking up data in one file or table and moving it to another is grouped most often with data integration. We discuss it here because it provides a source of newly derived variables. In Excel, the lookup operation is provided by the VLOOKUP and HLOOKUP functions. In KNIME, the lookup operation can be performed by the Cell Replacer and the String Replacer nodes. The lookup operation uses a common key between two data sources.

The Cell Replacer node in KNIME

Fig. 10.1 shows the KNIME workflow flowchart to perform a lookup operation with the Cell Replacer node.

The common key for each data source is the T_CODE variable, which exists in the Node 33 data input file and the T_CODE data file, but the description of the code exists only in the T_CODE file. The lookup operation scans through the Node 33 input file and looks up the T_CODE in it that matches the code in the T_CODE file, then installs the Description field in the T_CODE file into the data stream flowing out of the input Node 33 file. This operation can be performed at any time in the data preparation process, not just in the initial data integration process. This KNIME workflow is available with documentation on the book Web page.

The String Replacer node in KNIME

The String Replacer node functions like the Cell Replacer node, except it allows only part of the cell contents to be replaced that matches a pattern (definable with wildcard characters).

Variable (column) filtering operations

The integrated input data stream may contain some variables that are not relevant to the intended analysis; these variables should be screened out. Every data and

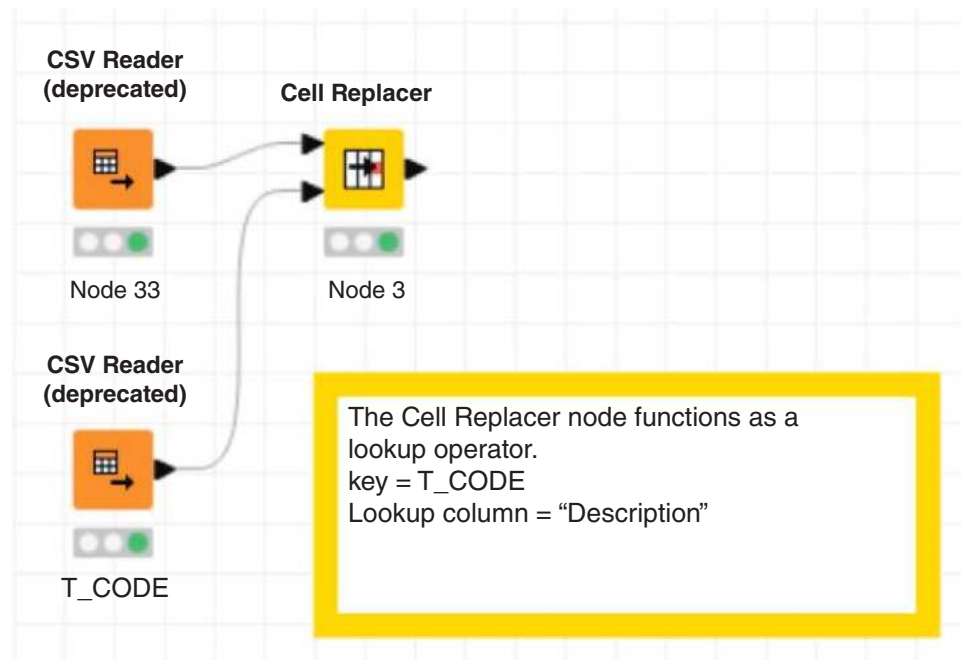


FIG. 10.1 Use of the Cell Replacer node to look up information in the data dictionary to define the T_CODE variable in the KDD-1998 PVA data set.

analytics tool has some facility for screening variables output to the data stream. KNIME uses the Column Filter node to do so. The KNIME Column Filter node is also very useful for tracking which variables exist in a data stream with hundreds of variables by the use of a partial matching capability in a variable name search string in the configuration screen. The column filtering operation is illustrated in the KNIME workflow shown in Fig. 10.2.

Row filtering operations

Row filtering is necessary to pass only some rows but not others. This type of filtering operation must be done to limit the rows entering a given analysis to only those pertinent to the analysis. For example, the training of a model to predict prostate cancer must not include rows for female patients, whose data patterns would only confuse the data pattern the algorithm optimizes to define the target variable. Sometimes rows must be filtered to feed the data plot for specific variables.

Fig. 10.2 shows how to use the variable filtering capability and the row filtering capabilities of KNIME to display graphics for specific data subsets.

Figs. 10.3 and 10.4 show the time course of the 7-day moving average of cumulative cases and new cases analyzed in the workflow shown in Fig. 10.2.

The KNIME workflow for the application of accessing the online data source and plotting these graphs is located on the book Web page, along with documentation describing its building and use.

Grouping and aggregation

It is very often the case that two data sources to be integrated contain data records (rows) at two different levels of detail. For example, one medical data source may contain many records of billing codes (e.g., ICD-10 codes) for the same patient ID recorded on different dates, while another data source contains demographic information for each patient. The need is to group the patient records with demographic data from each patient and output a file with one row per patient for analysis. Then both files to be integrated will be expressed at the same level of detail (the patient level).

A common grouping operation requires aggregation of contents of specific variables (columns or fields of data) in one of the data sources. The example described below groups patient records in a data source. Most grouping operations require some sort of aggregation operation to express results for detailed variables in the grouped output data stream. In many cases, the aggregation is the summation of the detailed values. In other cases, specific values for a given variable in the patient record data source are selected (e.g., the first or the last).

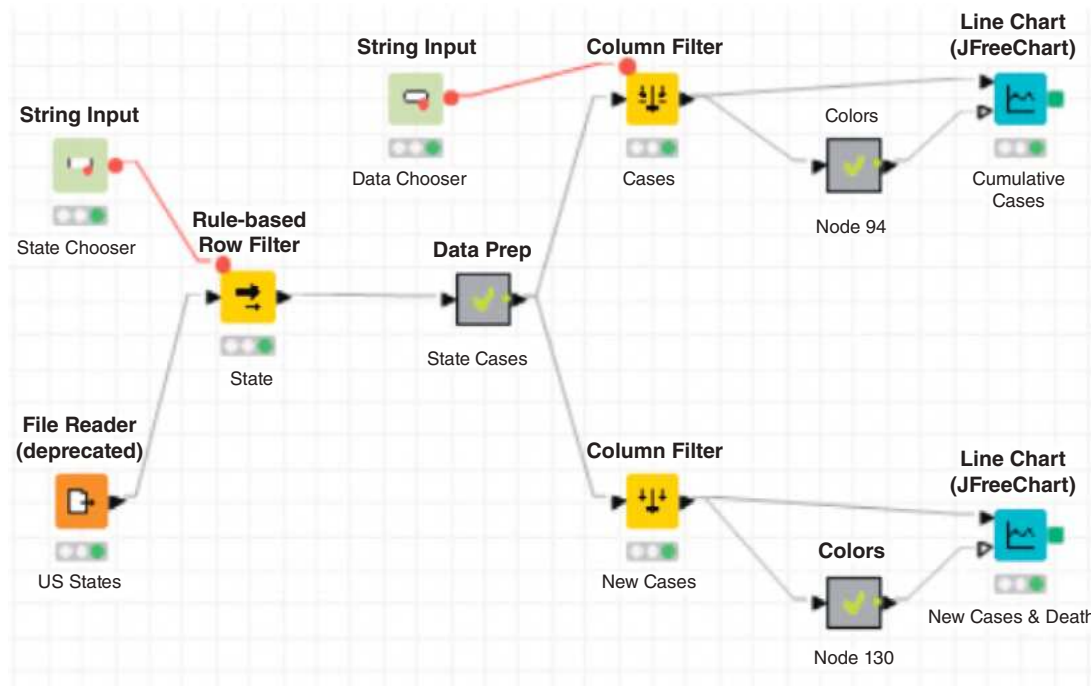


FIG. 10.2 Plotting of COVID-19 cases in California using variable (column) filtering and row filtering operations from 1/1/2022 to “Yesterday” (11/1/2022). Note: The date for Yesterday is redefined when the user refreshes the online data set accessed by the US States File Reader node.

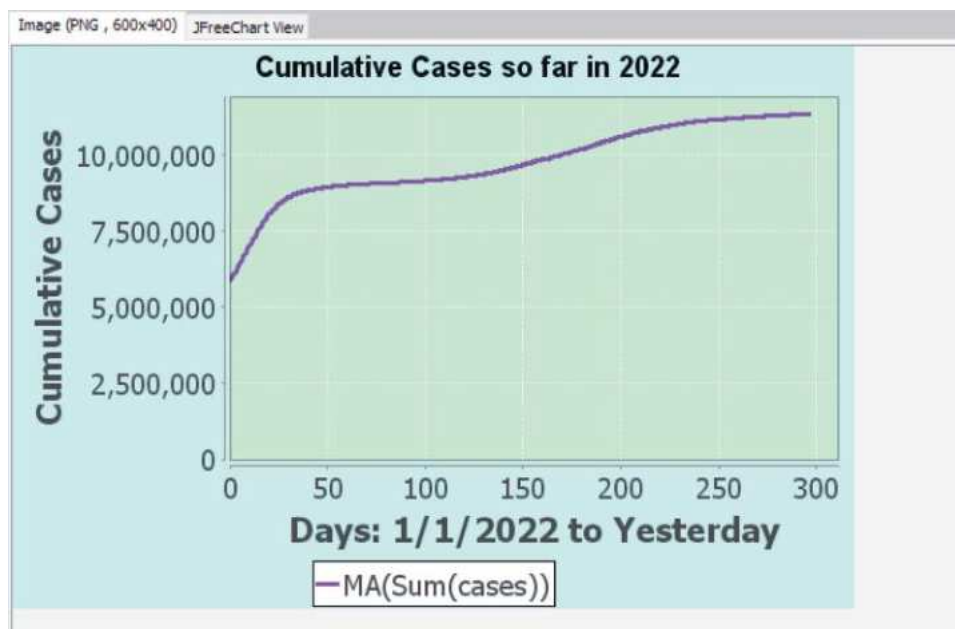


FIG. 10.3 Cumulative cases of COVID-19 in California (Yesterday = 11/1/2022).

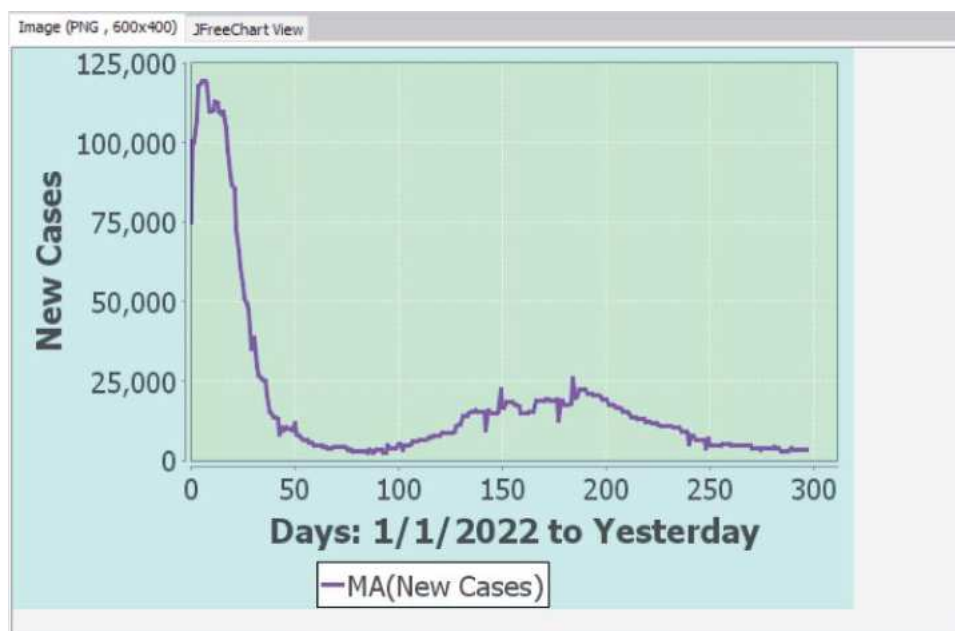


FIG. 10.4 7-Day moving average of new cases of COVID-19 in California (Yesterday = 11/1/2022).

Most analytics tools provide capabilities for grouping and aggregation of data.

Fig. 10.5 shows a KNIME workflow (data flowchart) for grouping of data in the input data set (Table 10.1) to output data at the grouped and aggregated level shown in Table 10.2. Most data and analytical tools provide the grouping and numerical aggregation facilities similar to those in KNIME used in the workflow.

Fig. 10.5 shows the KNIME workflow which inputs Table 10.1 and outputs Table 10.2 below it.

The grouping key specified in the configuration of the GroupBy node in Fig. 10.5 was the PatientID, which generates Table 10.2 with one row per patient. The GroupBy node is further configured to aggregate the #Polyps variable in Table 10.1 by summing the polyp numbers for each patient. KNIME provides an

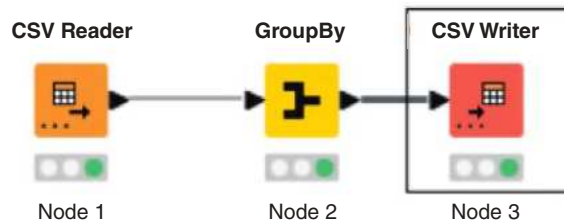


FIG. 10.5 KNIME workflow for grouping of input data in Table 10.1.

TABLE 10.1 Patient records for the colonoscopy procedure, showing the location in the colon and the number of precancerous colon polyps found.

PatientID	Date	Location	#Polyps
1	1/3/2022	Ascending	1
1	1/3/2022	Descending	2
1	1/3/2022	Transverse	1
2	2/17/2022	Ascending	3
2	2/17/2022	Descending	2
3	3/26/2022	None	0
4	7/14/2022	Ascending	3
4	7/14/2022	Descending	4

TABLE 10.2 Grouped and aggregated results of the GroupBy node shown in Fig. 10.5.

PatientID	Sum(#Polyps)	Unique concatenate(Location)
1	4	Ascending, Descending, Transverse
2	5	Ascending, Descending
3	0	None
4	7	Ascending, Descending

aggregation option to output a list of unique values among the input detailed records. In our case, the Unique concatenate(Location) output variable contains a comma-separated list of each Location type listed among the records for a given patient. The Unique-concatenate aggregation operation in the GroupBy node permits the flow of the list of Locations among the records for a given patient. This list can be queried in the downstream data flow to provide location-specific information for downstream analysis.

The workflow shown in Fig. 10.2 is available on the book Web page, together with the input data file and a description of the configuration of the GroupBy node.

Variable transformations

Various functions can be used to transform data from one form to another. The use of these functions for changing the data distributions of one to several variables is covered in Chapter 8 – Data conditioning. Sometimes you may want to derive a variable as a function of another.

Parametric statistical (PS) modeling algorithms (e.g., regression) assume that the data distributions of each predictive variable are normally distributed. If you want to use a PS modeling technique, you may have to transform the data distribution of some variables to satisfy this assumption.

If you want to use standard parametric algorithms for modeling (e.g., regression), then you must obey the “rules.” These rules are the assumptions of those algorithms:

1. The data of all variables is normally distributed (see Fig. 10.6). Notice that 95% of all of the values lie between +2 and –2 standard deviation units from the mean, and 99.7% of the values lie between +3 and –3 standard deviations units from the mean. All statistical significance metrics are based on this property of the data distribution. If the data of even one of the variables departs significantly from random, it can cause significant error in conclusions based on the analysis. This is the assumption that is broken most commonly in business data sets.
2. Effects of all predictor variables are linearly related to the dependent variable (the quantity predicted). This assumption is broken very often in business data sets too.
3. The effects of all predictor variables upon the dependent variables (the quantity predicted) are independent of the effect of any other variable. This assumption is also broken often in business data sets.

There are several other assumptions made in the definition of this theoretical “parametric model,” but the assumptions listed above can cause the greatest errors in analysis by far.

Mentioned in Chapter 9 was the use of the Z-transform to standardize numerical data before submitting them to the modeling algorithm. This transform function outputs numbers that range from + infinity to – infinity, with 99% of the values lying between +3 and –3 standard deviation units from the mean. This pattern of values corresponds to characteristics of the normal curve shown in Fig. 10.6.

Regression models are very useful in some data domains, because they output equations, which can be implemented very easily in SQL on the model

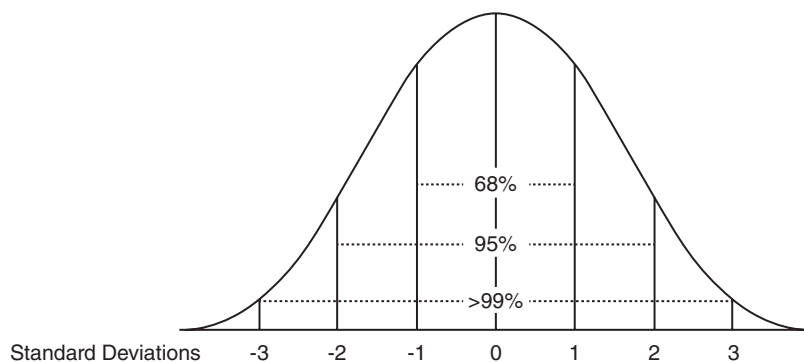


FIG. 10.6 A normal distribution curve.

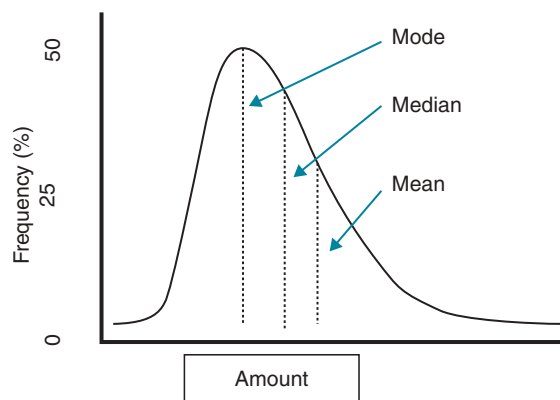


FIG. 10.7 The relationship between the Mean, Median, and Mode of a data set.

deployment systems. The only way to use these parametric modeling algorithms properly in most business data domains is to transform the data distributions.

Fig. 10.7 shows a distribution that is *skewed* (leans toward the left). In a Normal data distribution shown in Fig. 10.6, the Mean (or average), the Median (middle value), and the Mode (most common value) are all at the same place in the distribution (in the middle of it). In the skewed distribution of Fig. 10.7, the Mean, Median, and Mode values occur at different points on the distribution curve.

The ratio of the Mean to the Median value (Mean/Median) can express the degree to which the data distribution departs from normality.

Common functions can be used to transform various data curve shapes to be more normal in distribution: natural log, square root, and inverse square root.

Using a log transform to generate a normal distribution

The natural logarithm can be used to transform nonlinear and nonnormal raw data to render it more linear and normal in frequency distribution.

To see the effects of a natural log transformation on data distribution, we can consider a study of the number of fish (eastern mudminnow [*Umbra pygmaea*]) reported by McDonald (2014), as displayed in Fig. 10.8A and B.

Notice that the natural log transform did not generate a perfectly normal data distribution, but it is close enough to satisfy the normality assumption of the parametric algorithm (e.g., regression).

Using a log transform to satisfy the normality and linearity assumptions of parametric statistical algorithms

Consider the data set shown in Table 10.3.

Fig. 10.9 and 10.10 show the raw data with a trend line and the transformed data with trend line, respectively. The natural log function transformed the exponential trend line in Fig. 8.5 to express a linear trend line in Fig. 8.6.

Fig. 10.10 shows the dual action of the log transform: (1) to satisfy the normality assumption of the Parametric Model and (2) to satisfy the linearity assumption of the Parametric Model. Experience has shown that the appropriate transformation of variable distributions can increase predictability of a regression model by 10% to 20% and even more in some cases.

The application of a natural log transform can be performed in the KNIME Math Formula node or the Column Expressions for each variable. If you have a lot of predictor variables, this process can take a significant period of time. KNIME provides a multiple-column form of the Math Formula node to apply the same transform to a list of variables.

Some forms of regression account for the nonlinearity aspect of the data set by incorporating either the natural log transform (Logistic Regression) for numerical data sets or the Poisson transform for count data (Poisson Regression).

The problem with using log-transformed data in parametric statistical algorithms

Feng et al. (2012) described the common practice of transforming exponential data distributions with a log

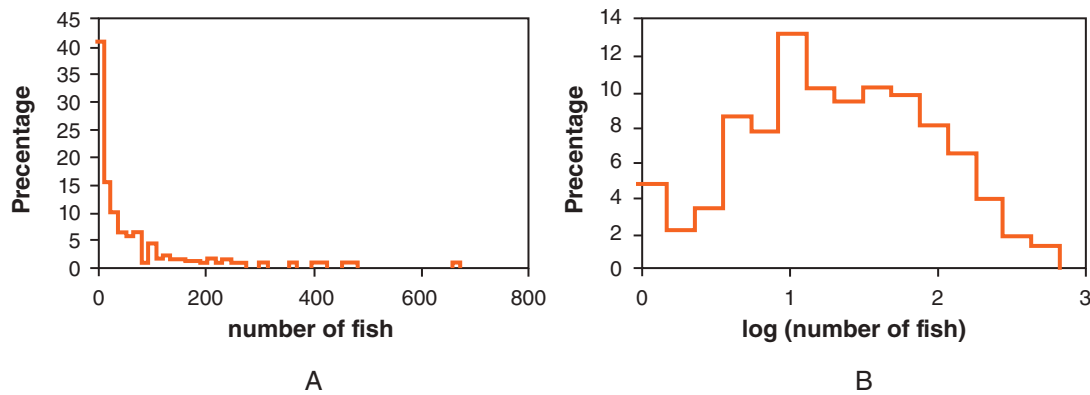


FIG. 10.8 Histograms of the data distribution of eastern mudminnows (*Umbra pygmaea*) per 75-meter section of stream. (A) shows the untransformed data and (B) shows the transformed data. Modified from McDonald (2014).

TABLE 10.3 Raw (X and Y) and transformed data (Ln Y)

X	Y	Ln Y
45	33	3.50
99	72	4.28
31	19	2.94
57	27	3.30
37	23	3.14
85	62	4.13
21	24	3.18
64	32	3.47
17	18	2.89
41	36	3.58
103	76	4.33

Modified from <http://www.real-statistics.com/regression/exponential-regression-models/exponential-regression/>

transform to convert them to a normal-like data distribution. The log-transformed data could be used in a PS algorithm like Linear Regression, without violating one of the important assumptions of the algorithm. The sample mean of the log-transformed data is often used to estimate the population mean of the original data using an anti-log function. Feng et al. (2012) reported that this process can generate inaccurate estimates of the true population mean of the original data.

Parametric statistical or machine learning algorithms?

With all of these characteristics of log transformation in mind, the analyst may decide that machine learning (ML) algorithms are preferable to PS algorithms in some cases. In practice, the analyst might choose an ML algorithm to generate the predictions of the outcome, and then choose to use the Logistic Regression algorithm to

generate the relative importance values of each predictor variable. Another benefit of this plan is that most implementations of the Logistic Regression algorithm include a valuable capacity to analyze the residuals (differences between observed and predicted values). To follow this plan in KNIME, data should be prepared for use with both ML and PS algorithms. This plan requires that all missing values should be filled (required by ML algorithms but not for PS algorithms), and predictor values should be standardized (required for PS algorithms but not for ML algorithms).

Different data sets might require different transformation functions to satisfy the assumptions of PS algorithms.

KNIME provides a number of transformation functions for use with appropriate data sets:

- Log_n (the natural logarithm)
- Log_{10} standard logarithm to the base 10
- Square root
- e (natural exponential power $\cong 2.71828$)
- $\exp$ (e to the power of x)
- Square
- Absolute

Feature extraction

Some features can be derived by splitting the contents of one or more existing variables. For example, a variable with contents containing information in a variable Status for single and married men and women could be split into two features, MaritalStatus and Gender, using the string processing logic of a data tool to compare partial string matches.

The unique-concatenate aggregation function available in the KNIME GroupBy node was introduced in the section “Grouping and aggregation.” The output of the operation contains a column holding a comma-separated list of locations in the colon where precancerous polyps were found during a colonoscopy exam (e.g., “Ascending,

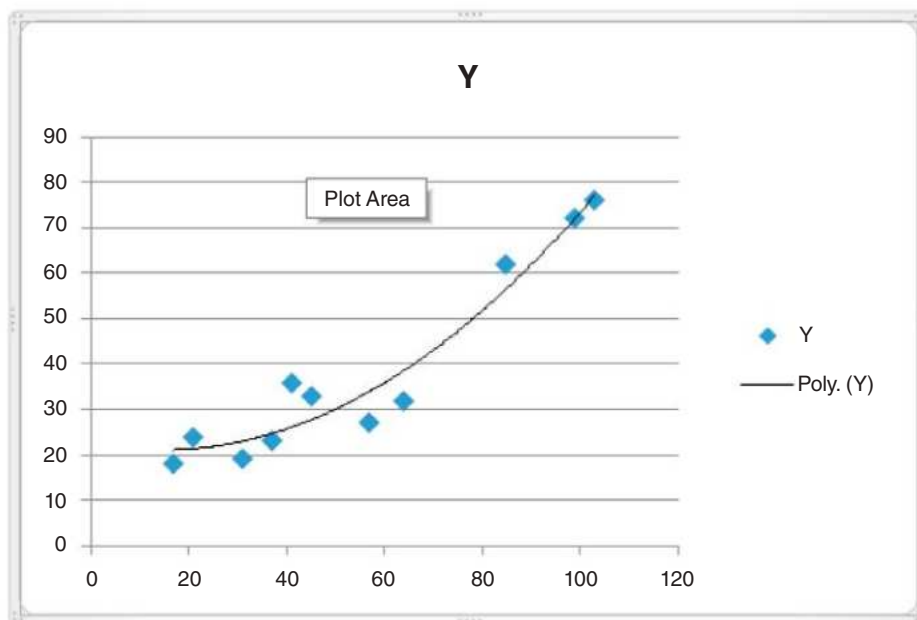


FIG. 10.9 A raw distribution of data showing its curved (exponential) trend line.

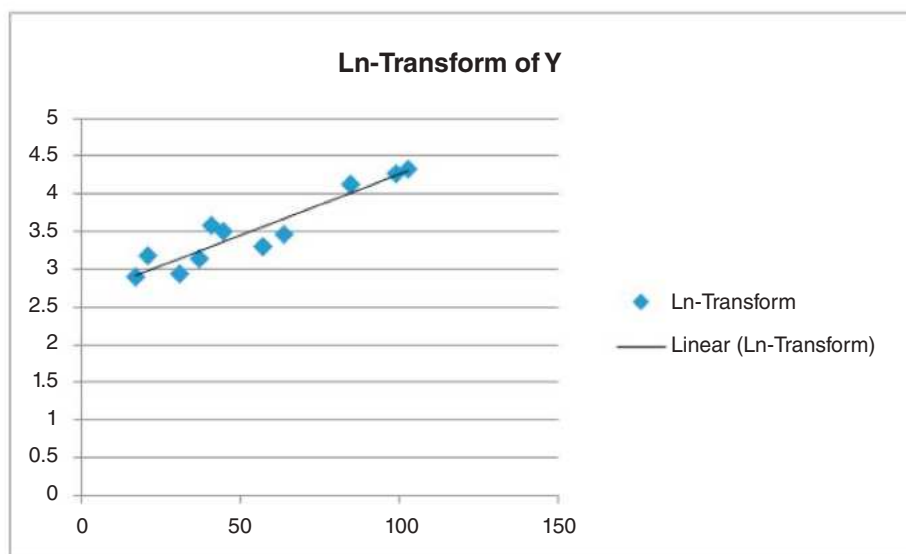


FIG. 10.10 The transformed distribution of data from Fig. 10.9 showing its linear trend line.

Descending"). These two locations can be processed to generate new binary variables for Ascending and Descending locations. In this way, data can be extracted from the pregrouped data set to generate new candidate predictor variables in the grouped data set.

Feature derivation using mathematical combinations of existing variables

Sums

The KDD-CUP 1998 data set (PVA_Data_All.csv, available on the book Web page) contains a series of

variables RAMNT_3 through RAMNT_24, which hold the dollar amounts donated to a charity in campaigns 2 through 24 during the previous 3 years. Notice also a variable, RAMNTALL, which the data dictionary describes as the sum of the amounts donated for all time. We could perform a sum of the columns RAMNT_3 through RAMNT_24 with a Column Aggregator node and compare the sum with RAMNTALL.

Fig. 10.11 shows the processing data streams for this section and the next.

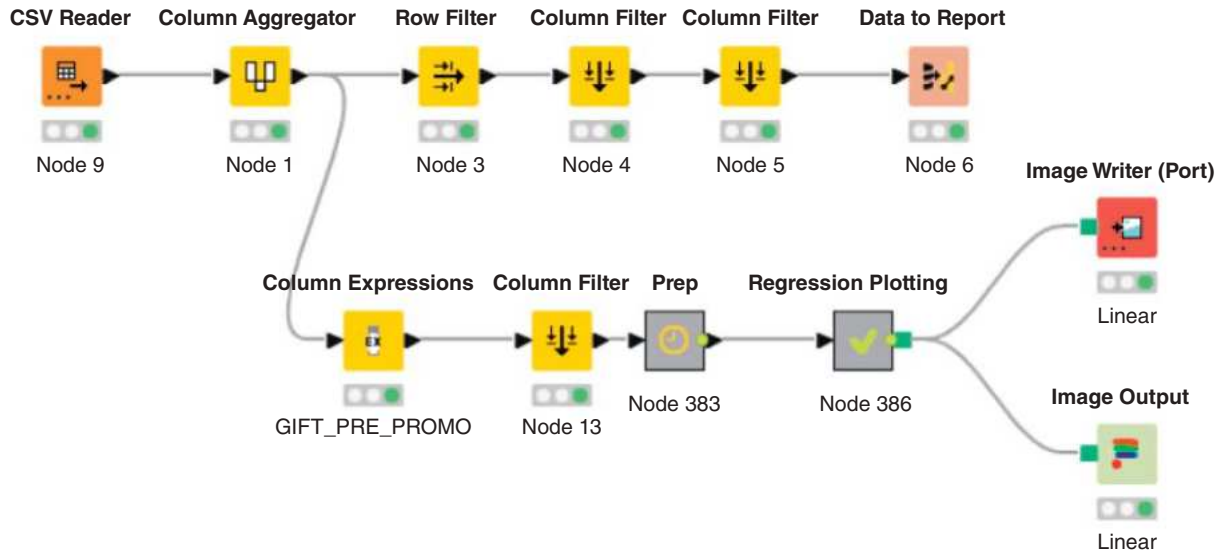


FIG. 10.11 Workflow to perform summing of the RAMNT variables and to calculate the ratio GIFT_PER_PROMO for linear regression plotting.

TABLE 10.4 Comparison of RAMNTALL (sum of donations for all time) and SumRAMNT (sum of RAMNT_3 through RAMNT_24).

Row ID	D RAMNTALL	D Sum
Row0	240	73
Row1	109	65
Row2	107	23
Row3	20	20
Row4	245	70
Row5	37	27
Row6	61	23
Row7	108	66
Row8	50	50
Row9	68	29

Table 10.4 shows the RAMNTALL variable and the sum of RAMNT_3 through RAMNT_24 as Sum for 10 rows processed with the top data stream shown in Fig. 10.11. The GIFT_PER_PROMO variable derivation and regression plotting will be discussed below in the Ratios section.

Notice that only two of the values for RAMNTALL and Sum are the same. Most of the values for RAMNTALL are greater than those of Sum. This result means that some of the values used to compose RAMNTALL were donated in Campaign_2 or prior to

campaign_24. The large disparity between most of the numbers in the two fields suggests that there were many campaigns previous to campaign_24 to which specific donors contributed. Therefore the Sum variable may be more predictive of the probability of donating in the next campaign than the RAMNTALL variable. The Sum variable was not included in the data provided for subsequent exercises; it was an afterthought. The problem with doing this now on the data file in its present state is that we filled a lot of missing values in these the RAMNT variables with the mean value for demonstration. We should have filled these variables with 0, if anything, before calculating the sums. This is a good example of the need to observe practical. Later in the process, you may discover later in the process some operations that you should have done earlier in the process. In this case, we have two choices for proceeding: (1) go back and use the Missing Values node to fill double values (D) with 0 instead of the mean and then regenerate all of the data files in all subsequent exercises, or: (2) forget it and go on. The latter course was followed in this book. Such practical considerations of cost and scheduling may prevent you also from choosing option #1 in your company.

Ratios

Calculation of ratios is one of the most common operations among new variable derivation activities. Values of RAMNTALL may not be as predictive of future donations as the Sum derived variable might be; however, it can be used in ratios with other variables covering the

donor lifetime response. The bottom data processing stream in Fig. 10.11 contains a KNIME Column Expressions node, which calculates values for `RAMNTALL/NUMPROM` and derives a new variable `GIFT_PER_PROMO`. This ratio expresses the frequency of donations over the donor lifetime until 1998. This donation frequency measure can be regressed with the average amount of gifts (`AVEGIFT`) to generate the linear regression plot shown in Fig. 10.12.

Updating variables

The `PVA_Data_All.csv` file used in the previous section contains the `NUMCHLD` variable, preceded by 5 `CHILDnn` variables. If you analyze them, you will see that the numbers are not consistent. Evidently, the `NUMCHLD` variable was obtained from a different source than the `CHILDnn` variables. We could use the `CHILDnn` variables to update the number in the `NUMCHLD` variable

Data abstractions

It may be useful to map (relate) a given data value to some other context which is more general or more specific than that of the raw data element. A data abstraction is a technique used to relate these raw data elements to a specific context environment. Data abstractions can be classified into four groups (Lavrac et al., 2000):

- *Qualitative abstraction* – a numeric expression is mapped to a qualitative expression. For example, in an analysis of teenage customer demand compared with that of others, customers between the ages of 13 and 19 could be abstracted as a value of 1 to a variable “teenager,” while others are abstracted to a value of 0.
- *Generalization abstraction* – an instance of an occurrence is mapped to its class. For example, in an analysis of Asian preferences compared with non-Asian, listings of “Chinese,” “Japanese,” and “Korean” in the `Race` variable could be abstracted to 1 in the `Asian` variable, while others are abstracted to a value of 0.

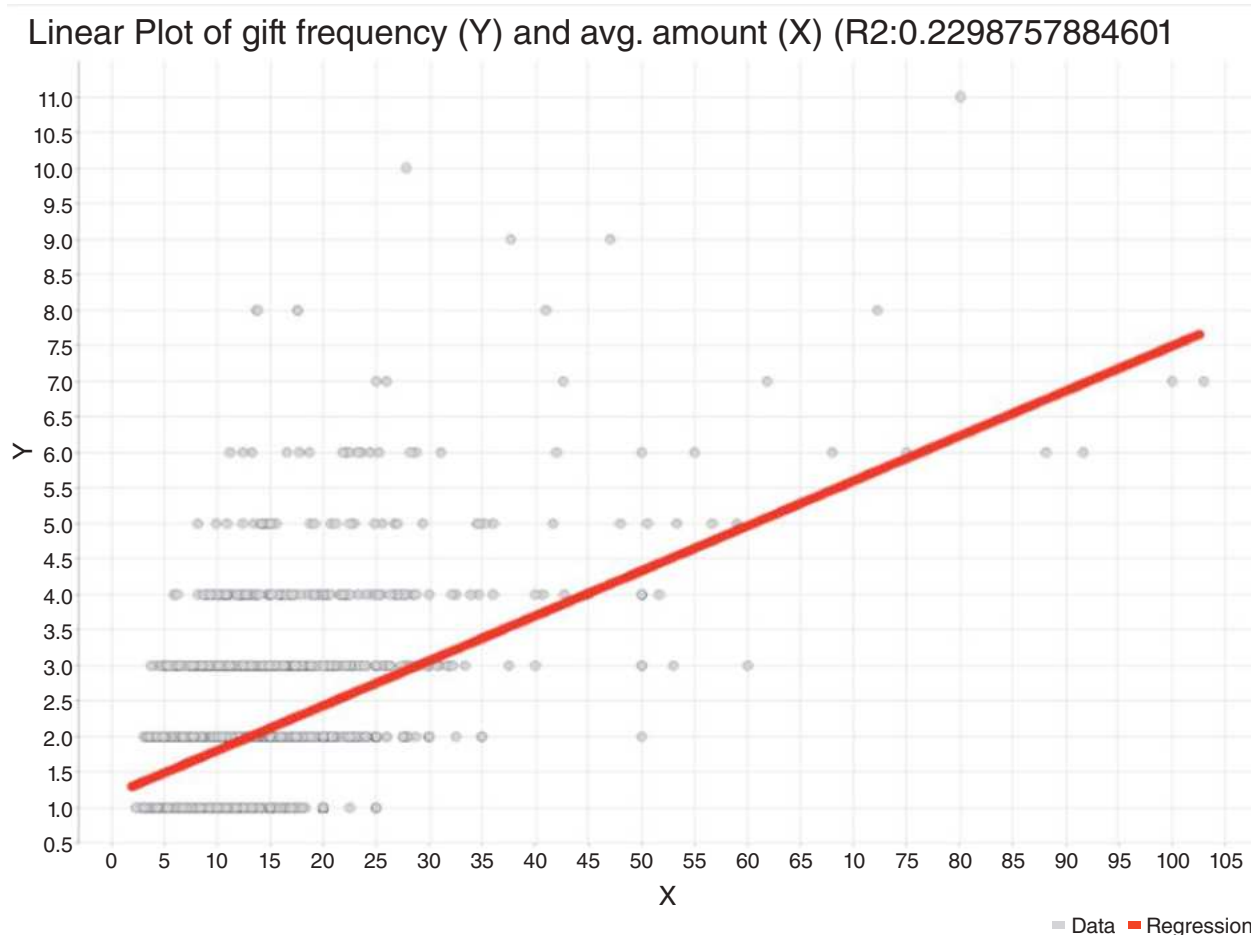


FIG. 10.12 Linear regression plot of gift frequency (`GIFT_PER_PROMO`) and the average gift (`AVEGIFT`).

- *Definitional abstraction* – in which one data element from one conceptual category is mapped to its counterpart in another conceptual category. For example, when combining data sets from different sources for an analysis of customer demand among African-Americans, you might want to map “Caucasian” in a demographic data set and “White Anglo-Saxon Protestant” in a sociological data set to a separate variable of “Non-Black.”
- *Temporal abstraction* – a variable in a time domain in which one reference is mapped to a time domain with another reference. The most common temporal abstraction is the *lag variable*, in which a data record at time t_0 includes values from a given field at time $t_{-1}, t_{-2}, \dots, t_{-n}$. These lag variables can be very predictive in CRM applications.

Lag variables

Temporal abstractions are referred to commonly as “lag variables” because the effect is related to the target variable “lags” by a specified time period. Lag variables can be among the most powerful predictors in a customer relationship management (CRM) model. Chapter 19 focuses exclusively on the use of lag variables in CRM modeling. A lag variable is a value representing the state of a variable at some time in the past. The theory which lies behind the use of lag variables is that the best indication of what customers will do in the future is what they did in the past. Sometimes, a given customer response may lag behind a cause by some period of time. For example, the response of some banking customers to leave a bank (close the last checking account) may not occur until 2 to 3 weeks after the intent to do so because of work schedules and other delays in the life of the customer. Deposits and withdrawals may still occur during this period of time, which may follow a pattern related to the decision of leave the bank (churn). In this case, the modeler must copy the number and amounts of deposits and withdrawals made by the customer 2 to 3 weeks ago and install them in the same record (row in the data set) along with other nontemporal variables (e.g., Age and Gender). Many predictive analytics tool packages provide the capability to generate these lag variables.

An early example of the use of lag variables was to support churn analysis in a phone company. Consider the telephone usage data in Table 10.5. These records are similar to those that could be extracted from a telephone company billing system. These records constitute a time series of up to 12 monthly billings for each customer. You will notice the first customer was active for the entire 12 months (hence 12 records). But the second customer has only nine records, because this person left the company. In many industries (including telecommunications), this loss of business is called attrition or “Churn.”

TABLE 10.5 Telephone company billing system data extract. (Note: the “churn month” for customer #2 is 9, or September.)

Cust_ID	Month	MOU	Due	\$Paid	\$Balance
1	1	26	\$21.19	\$21.19	0
1	2	91	\$74.17	\$37.08	\$37.08
1	3	43	\$35.05	\$17.52	\$17.52
1	4	74	\$60.31	\$60.31	\$0.00
1	5	87	\$70.91	\$35.45	\$35.45
1	6	99	\$80.69	\$40.34	\$40.34
1	7	60	\$48.90	\$24.45	\$24.45
1	8	99	\$80.69	\$40.34	\$40.34
1	9	68	\$55.42	\$55.42	\$0.00
1	10	50	\$40.75	\$20.38	\$20.38
1	11	38	\$30.97	\$15.49	\$15.49
1	12	92	\$74.98	\$37.49	\$37.49
2	1	20	\$16.30	\$8.15	\$8.15
2	2	26	\$21.19	\$21.19	0
2	3	38	\$30.97	\$15.49	\$15.49
2	4	61	\$49.72	\$24.86	\$24.86
2	5	84	\$68.46	\$68.46	\$0.00
2	6	84	\$68.46	\$34.23	\$34.23
2	7	35	\$28.53	\$14.26	\$14.26
2	8	31	\$25.27	\$12.63	\$12.63
2	9	26	\$21.19	\$10.60	\$10.60

One of the authors of this book led the team that developed one of the first applications of data mining technology to the telecommunications industry, churn modeling, in 1998. This model was based on an Analytical Record created by reverse pivoting the time-series data set extracted from the billing systems of telephone companies.

This data set could be analyzed by standard statistical or machine-learning time-series tools. Alternatively, you can create an Analytical Record from this data set by doing a transpose operation. This operation copies data from rows for a given customer and installs them in columns of the output record. This output operation would generate the record (Table 10.6) for the first 9 months.

This record is suitable for analysis by all statistical and ML tools. Now that we have the time-dimensional data “flattened out” into an Analytical Record, we can re-express the elements of this record in a manner that will show churn patterns in the data. But in its present format, various customers could churn in any month. If we submit the current form of the Analytical Record to the

TABLE 10.6 Minutes of use (MOU) by month number.

CUST_ID	MOU-1	MOU-1	MOU-3	MOU-4	MOU-5	MOU-6	MOU-7	MOU-8	MOU- 9
1	26	91	48	74	87	99	60	99	68

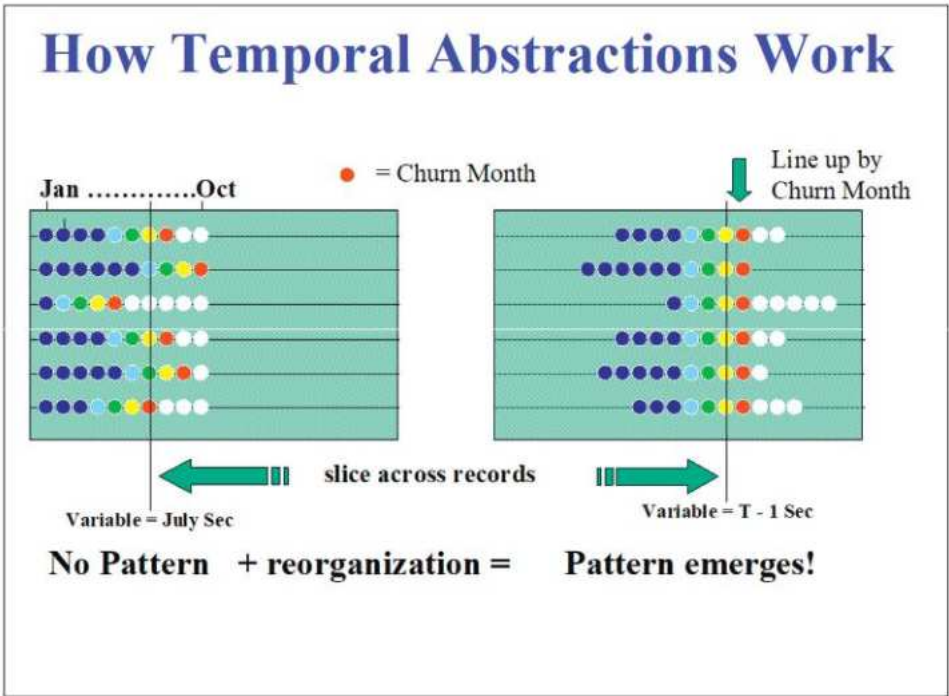


FIG. 10.13 Demonstration of the power of temporal abstractions (lag variables) for “amplifying” the churn signal to the analysis system (our eyes and brains).

modeling algorithm, there may not be enough “signal” to relate churn to minutes of use (MOU) and other variables in a specific month. How can we rearrange our data to intensify the signal of the churn patterns? We can take a page out of the playbook for analyzing radio signals by performing an operation analogous to signal amplification. We do this with our data set by deriving a set of lag variables in which the values in each variable are related to the churn month. Instead of analyzing the relationship of churn (in whatever month it occurred) to specific monthly values (e.g., January MOU), we relate the churn month to the MOU value for the previous month and the month before the previous month, and so forth. Fig. 10.13 illustrates the power of lag variables for clarifying the churn signal to our visual senses, and it will appear more clearly to the data mining algorithm also.

The red dots in Fig. 10.13 represent the month (month t) that each of the six customers churned. The yellow dots are the MOU values for the month previous to the churn action (month $t-1$). Aligning the yellow $t-1$ dots for each customer along the same line displays the pattern among the other dots (orange for $t-2$ and light blue for $t-3$). Training the model on lag variables (rather than the raw variables) permits the algorithm to recognize the pattern in the dots prior to churn in a way similar in

effect to the way reorganization of data in the left chart to form the right chart in Fig. 10.13 permits our eyes to see it.

In KNIME, the Lag Column node performs this task with a historical data set for a given customer. An example using the lag variable node is presented in Chapter 19 – Customer response modeling. You can practice lag variable derivation with that workflow as described in Chapter 19 and provided for you on the book Web page.

Temporal data mining

Box and Jenkins (1970) pioneered traditional time-series analysis, which introduced, the Auto-regressive moving-average (ARMA) algorithm. This algorithm analyzed changes in the moving average of values in a time series. The algorithm was extended by adding a data abstraction of the difference between adjacent values in the time series (the I-term in the ARIMA algorithm). The application of time-series analysis with ARIMA has quite a long history. Applications include weather forecasting, sales forecasting, and financial analysis.

During the last 20 years, other time-related (temporal) data sets have become increasingly popular (e.g.,

containing historical medical records). The nature of these temporal data sets is quite different from those in the past analyzed by ARIMA. According to Laxman and Sastry (2006), one major difference between them is the large size and heterogeneous content of the data. Many of the temporal data variables are numeric and could be analyzed by traditional time-series methods. However, many temporal variables may be nominal or symbolic, which cannot be analyzed by algorithms like ARIMA. Another aspect of the nature of temporal data is that the analyst has no control over data collection to ensure that the statistical assumptions of ARIMA are satisfied (e.g., normality and independency).

The second major difference between traditional time-series analysis and temporal analysis is the analytical goal of the analysis. Instead of searching for a relationship between a given variable (the target) and specific predictor variables, the analyst may not even know which variables are expected to exhibit any relationship to the target variable. A good example that the authors cite is market basket analysis to identify which items tend to be purchased with other specific items. A famous example is that beer occurs often with diapers in the same market basket to serve husbands who agree to tend Junior when Monday night football games are on television while their wives go out with friends. That insight from market basket analysis led supermarket managers to invent end caps, where diapers are located at the end of the beer cooler aisles. A similar observation led to the co-location of chips and salsa.

The challenge with temporal data mining is the extensive data preprocessing necessary to perform it. See Laxman and Sastry (2006) for a description of the data preprocessing operations that must be done to support pattern-matching algorithms and model-based algorithms used commonly in temporal data mining.

Other temporal analysis techniques

KNIME offers several general purpose temporal analysis functions.

- Time-difference between two date-time values
- Time shift by some constant or variable heuristic
- Date-time row selection

Discretization

PS algorithms can't process categorical variables that contain text values (e.g., Gender categories of "Male" and "Female"). One of the data engineering processes required to prepare data for analysis by statistical modeling algorithms (e.g., Linear Regression) is to convert category classes (e.g., "Male" and "Female") into numbers. Even ML algorithms may perform better with these transformations, because they permit the algorithms to

focus on each categorical class separately. This transformation process is called *discretization*.

Dummy variables

The most common way to discretize categorical data variables is to create a new binary variable for each class in a categorical variable. These variables are called "Dummy" variables. A dummy variable row has the number 1 in a given row if the original categorical variable class was present in that row; otherwise, it contains the number 0. If there are many categorical variables in the data set with many classes in many of them (e.g., state names in the State variable), many dummy variables must be derived, and it may take a very long time to do it. Fortunately, many analytics tools contain facilities for generating dummy variables for each class automatically for a list of input variables. The KNIME One to Many node performs this job in seconds, even when hundreds of dummy variables are required.

Postscript

After you have spent a large amount of time building your list of features in feature engineering, you will be faced with the task of whittling it down to only those features that will build the best model. Why do you have to do that and how do you do it? Chapter 11 answers those questions and describes a number of common feature selection methods for creating that almost-magic product, the "short list." This short list of features is most likely to build the optimum model; therefore it might be the most important product you create along the way to building a highly predictive model.

Keywords and phrases

Assumptions, Aggregation, Linear, Abstractions, Normality, Discretization, Transform functions, Dummy variables, Parametric statistical algorithms, Feature derivation, Feature extraction, Feature transformation.

References

- Box, G. & Jenkins, G. (1970). *Time series analysis: forecasting and control* (pp. 553). San Francisco: Holden-Day.
- Feng, C., Wang, H., Lu, N., & Tu, X. M. (2012). Log-transformation: applications and interpretation in biomedical research. *Statistics in Medicine*, 32, 230–239.
- Laxman, S., & Sastry, P. (2006). A survey of temporal data mining. *Sadhana*, Vol. 31(2), 173–198.
- Lavrac, N., Keravnou, E., & Zupan, B. (2000). "Intelligent Data Analysis in Medicine," *White Paper, Faculty of Computer and Information Sciences*. Slovenia: University of Ljubljana.
- McDonald, J. H. (2014). *Handbook of biological statistics* (3rd ed., pp. 140–144). Baltimore, Maryland: Sparky House Publishing.

Feature selection

Preamble

“Every block of stone has a statue inside it and it is the task of the sculptor to discover it.” (Attributed to Michelangelo – [Creighton, 1980](#)).

The problem with this oft-quoted statement by Michelangelo is that he never said it ... at least not directly. There are several references in his works to this thought, however. One reference to this thought occurs in one of his sonnets, “Non ha l’ottimo artista alcun concetto.” One English translation of the first stanza of the Italian in the sonnet is:

The best of artists hath no thought to show
Which the rough stone in its superfluous shell
Doth not include: to break the marble spell
Is all the hand that serves the brain can do. [Vaughan \(2016\)](#)

Another translation of the sonnet is:

Not even the best of artists has any conception
that a single marble block does not contain
within its excess, and that is only attained
by the hand that obeys the intellect.

A rendering of one of the stanzas in the sonnet by [Creighton \(1980\)](#) is actually a transliteration:

“Nothing the best artist can conceive but lies as potential in
the block of stone, superfluous matter round it.”

Another reference to this thought is contained in his Sonnet 151 (circa 1538–1544):

No block of marble in the quarry lies Unknown and unap-
preciated by
The sculptor’s mind, which in his skill applies
The form and figure that within it lie.

Many subsequent historians paraphrased the thought contained in these sonnets as:

“A statue is in every block of marble; it is the sculptor’s job
to remove the superfluous material around it”.

Some Historians have taken the idea expressed in Michelangelo’s sonnets and inferred the conclusion that the

sculptor must remove the superfluous material from around the statue to “discover” it. This idea has been named by art historians as Michelangelo’s Sculptural Theory of Subtraction ([Columbia University, 2023](#)). Even though this inference has been imported into the quotes by modern scholars, we might conclude from the text of these sonnets that Michelangelo believed it, and that is the important thing in reference to the subject of this chapter.

Modeling and sculpting

Analytical modeling functions a lot like Michelangelo’s sculptural theory of subtraction. Like the sculptor, the analytical modeler starts out with a bunch of potential predictor variables, most of which are superfluous “matter” for building the intended model. Like the sculptor, the modeler must remove this “superfluous matter” from the data set, following the dictate of Occam’s Razor (see below). There are a number of feature selection techniques available for him to use, including the very iterative operation of machine learning algorithms (e.g., decision trees) that function very much like the process of sculpting. The sculptor follows a very iterative process for removing a lot of superfluous material around the “statue” he “sees” in the block of marble. He removes some initial material with his chisel and steps back to view it. He steps forward to remove more material and repeats the process until all superfluous “matter” is removed. The analytical modeler follows an analogous iterative process to remove superfluous features to optimize the predictive power of the trained analytical model. This chapter will present a few feature selection methodologies to permit the modeler to find the optimal group of features that can be used to build the best model.

Some introductory concepts

Features versus variables

A candidate predictor value in its raw state is referred to as a “variable.” Some predictive analytics algorithms

(e.g., Support Vector Machines – SVMs) recode (“maps”) the raw variable values using some kernel function to generate a “feature.” The multidimensional feature space generated by this mapping operation permits a complex relationship to be expressed much easier than possible in raw variable space. In general usage, however, it is often the case that the term feature is used to refer to any modified or transformed variable. In practice, features are variables in the data set after it is passed through a number of data preparation operations, whether or not all variables are modified. We will follow that distinction in this book.

Why reduce the number of features submitted to a predictive analytics algorithm?

The nature of complex systems follows an ancient maxim in philosophy popularized by a 14th-century clergyman and philosopher, William of Occam. William wrote:

Latin: “Non sunt multiplicanda entia sine necessitate”

English literal: Entities (or solutions) should not be multiplied (or elaborated) beyond necessity.

In the modern vernacular: “Keep it simple, stupid (*KISS*)”.

William of Occam did not invent this thought, he just popularized it. The principle behind it, known as the Law of Parsimony (or the *Lex Parsimonae*), can be traced through a number of philosophers including Thomas Aquinas all the way back to Ptolemy and Aristotle. The mathematical reason for following this principle in analytical modeling is that extraneous features just get in the way of composing the optimal data pattern of the target variable by the modeling algorithm. In respect to William of Occam, this expression of the *Lex Parsimonae* has become known as Occam’s Razor.

A generalized predicted relationship can be expressed in regression format by Eq. (11.1).

$$Y = f_1 X_1 + f_2 X_2 + f_3 X_3 + \dots + f_n X_n \quad 11.1$$

Each candidate predictor variable in Eq. (11.1) represents a dimension of an n-dimensional decision hyperspace of error in which any predictive algorithm must “hunt” for the best solution, defined usually as the solution with the minimum prediction error. Ideally, this minimum prediction error is located in the hyperspace at the lowest point over the entire hyperspace (the global minimum error), and not at just some local minimum error location. Judicious setting of appropriate hyperparameters like the learning rate and momentum in a neural network algorithm can optimize the likelihood of finding the location in the hyperspace with the global minimum prediction error (see the further discussion of these parameters below). The Y value in Eq. (11.1) is the

outcome of the regression equation, the X values represent each predictor variable, and the f value associated with each predictor variable represents a coefficient that expresses the combined effect of variable scale and the relative strength of each variable in the outcome. Each unimportant variable contributes more “noise” than “signal,” and just gets in the way of the search for the optimum solution.

Consistent with Occam’s Razor, we often find in predictive analytics that modeling algorithms perform better and better as the number of predictor variables increases ... up to a limit. Beyond that limit (or beyond Occam’s condition of “necessity”), algorithms perform poorer and poorer as the number of variables increases. What causes this effect?

The modeling algorithm must find the combination of variables whose values permit the algorithm to generate the least total error in prediction. It does that by conducting a directed “walk” over the decision “surface.” Sometimes, the walk is driven by a random number (called a “random walk”) and sometimes, it is driven by some heuristic. The modeling algorithm can do a much better job if the noise is reduced. Eliminating the number of candidate predictor variables that are unimportant predictors (those that are not “necessary,” according to Occam’s Razor) does this job. Sometimes, the walk over the decision surface becomes “trapped” in a local low point of total error, and it “decides” that this is the absolute lowest error point in the entire decision space (global minimum error). We can see this situation expressed graphically in Fig. 11.1, which shows a two-dimensional decision surface (two variables).

Optimizing the search for the minimum error location on the decision surface

The blue line in Fig. 11.1 shows the search path on a decision surface by an algorithm following a steepest downhill path of prediction error, beginning at a starting point. The path of the blue line causes the solution to become “trapped” in a point on the decision surface associated with only a local minimum total prediction error. The challenge for algorithm designers is to provide some mechanism to broaden the search to “look” beyond the nearby “hump” (local maximum error point) in its search for an error lower than the minimum error point evaluated previously. Imagine that the decision point can be represented by a ball bearing rolling over a two-dimensional decision surface. The tendency of the ball to keep rolling over the uneven decision surface can be characterized by two factors (or hyperparameters): *velocity* of movement of the ball and its *momentum*.

The momentum of a physically moving ball is proportional to its size. In the algorithm configuration, this relationship is provided by the hyperparameter,

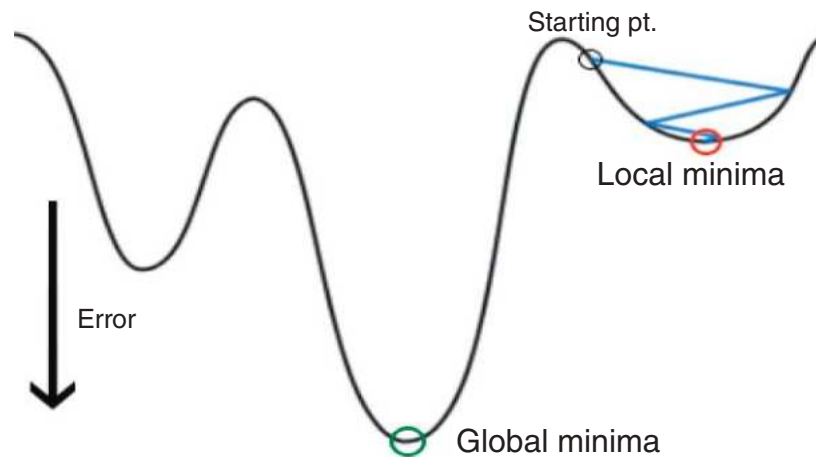


FIG. 11.1 Relationship between local minimum error and global minimum error.

also called the momentum. The other hyperparameter that characterizes the motion of the rolling ball is related to its speed of movement, represented by a term called the “learning rate.” The combination of the momentum and learning rate will control the movement of the search path by the algorithm over the decision surface. If a routine of the algorithm can be programmed to function as learning rate and momentum of the search process on a decision surface, a momentum of a certain size moving at a certain velocity (learning rate) will be large enough to “roll” over the hump of the local maximum error point on the decision surface, and “hunt” for a local minimum error level lower than the previous one. Such an algorithm routine could carry on the error minimization search to find the global minimum point shown in Fig. 11.1. A neural network algorithm actually supplies both of these parameters (called *hyperparameters*). It is easy to visualize this process in the two-dimensional space of Fig. 11.1, and it is relatively “easy” for the algorithm to find a local minimum error point. These parameters can be tuned in most machine learning neural network algorithms to maximize the likelihood of finding the global minimum error position on the decision surface. This process of model refinement is called hyperparameter optimization.

The Curse of Dimensionality

As the number of dimensions increases, it becomes exponentially more difficult for the modeling algorithm to find the global minimum error point on the decision error surface, because it reduces the probability of finding it with the chosen values for the learning rate and momentum hyperparameters. That effect is expressed by the concept of the Curse of Dimensionality.

The “short-list”

The “cost” of increasing the number of variables in a candidate solution must be “paid” until the total error is minimized. The goal in the definition of the number of variables to submit to the modeling algorithm is to limit the number to only those that decrease the total solution error significantly. Beyond that point, the total error is likely to increase, following the Curse of Dimensionality. The final list of candidate variables submitted to the modeling algorithm is called the “short-list.” This short-list is very valuable. Except for the predictions provided by the optimized model, the short-list is the most valuable product of a predictive analytics project. Several court cases have judged that this short-list can be classified as a trade secret (see *DTM Research, LLC v. AT&T Corp.*, 245 F.3d 327 (4th Cir. 2001), and *Advanced Fluid Systems, Inc. v. Huber*, No. 1:13-CV-3087 (M.D. Pa. June 18, 2014). The financial disposition of the DTM versus AT&T case awarded DTM Research \$17 million, based on the value of the short-list, which the primary author of this book defined to the jury.

Feature selection versus dimensionality reduction

All feature selection operations reduce the dimensionality of the decision surface, but not all dimensionality reduction algorithms reduce the number of features.

Classification of dimensionality reduction methods

Development of the short-list by some feature selection method prior to building predictive models has

been studied intensively during the last 30 years. A classification of methods presented by Kohavi and John (1997) includes three classes of methods. This scheme has become a common way to classify variable selection methods.

- Filter methods.
- This approach uses various heuristics in the design of filters to select or eliminate variables.
- Wrapper methods.
- In the context of predictive analytics, a “wrapper” is a program that translates input data to fit a subroutine of the wrapper program to do a specific data manipulation operation. Such a wrapper program might recode input categorical variables into a series of “dummy” variables, one for each categorical class containing a 1 or 0 for the presence or absence of the class in the categorical variable for a given record. A second wrapper subroutine might normalize all integer and real (decimal) numbers to range between 0 and 1. A third subroutine might run a regression algorithm to calculate the regression coefficients of each input predictor variable. A final subroutine might output the list of variables with coefficients with values above a specified threshold.
- Embedded methods.
- A good example of an embedded method is provided by a stepwise multiple regression, which includes an additional routine to pass only those variables as predictors with f values greater than a specified threshold. In this case, the variable selection operation is embedded in the architecture of the algorithm. Forward and backward variations of a stepwise regression are available, which either start with no variables and include the variable with the highest f value in a given iteration, or start with all of the variables and exclude the variable with the lowest f value. Following either approach, the operation stops when the model built in a given iteration includes only those variables with f values equal to or greater than the threshold. For other algorithms, other values are used to terminate processing. For example, the termination value might be set to be the maximum number of processing iterations or the maximum processing time. These algorithm configuration options are called “stopping” functions.” For business purposes, the best choice is a stepwise Logistic Regression (LR) algorithm configured with a maximum number of processing iterations. Training of the LR algorithm will progress until either the solution converges on the minimum error, or it reaches the maximum number of iterations. If the algorithm does not converge on a solution, examination of the

list of predictor variables could reveal a problem with a specific variable (e.g., too small a variance), and that variable can be removed from the specification of the short-list. The LR algorithm uses a natural log function in the definition of the Y variable to account for much of the nonlinear response between the Y variable and the combined effect of the predictor (X) variables common in many business responses (e.g., propensity to buy).

There are also some nonparametric approaches to feature selection. Multifactorial Dimensionality Reduction (MDR) was developed by The Computational Genetics Laboratory at Dartmouth University to analyze single nucleotide polymorphisms (SNPs) in DNA sequence data to determine their relationship to candidate predictor variables. MDR employs a form of a nonparametric entropy method that evaluates information gain (IG) to identify the short-list of “interesting” SNPs (cf. the discussion of Information gain below). Subsequently, predictive interactions between the selected SNPs can be generated (Moore et al., 2006). MDR can be used to analyze almost any type of input file that includes a list of candidate predictor variables, followed by a class variable (the target) at the end of each row saved in tab-delimited format. MDR is basically a categorical analysis method, and is limited to categorical variables with less than 200 classes (or numerical variables with less than 200 unique values). MDR does something that very few other analytical programs can do: it defines interaction variables automatically at any specified level (two-variable to n -variable interactions). MDR can be downloaded at: <https://sourceforge.net/directory/os:windows/?q=multifactor+dimensionality+reduction>

Decision tree processing elements useful as feature reduction filters

Information gain and Gini coefficient are two methods incorporated into the processing of decision trees to decide where to split the range of a given feature to build a branch of the tree. These methods can also be used as feature reduction filters.

Information gain

In addition to its use in the processing of the LR algorithm, IG is used by many decision tree algorithms to decide where to split the range of a given feature at one level in the decision tree. If the input data are normalized to a range from 0 to 1, the entropy of a given candidate split in the range of a feature is expressed as Eq. (11.1).

$$H(Y) = -p \log_2 p - (1-p) \log_2 (1-p); \quad 11.1$$

where, p is the prior probability that a given variable Y is 1, and $1-p$ is the probability that $Y = 0$. The prior probability of $Y = 1$ is the relative proportion of values in the subrange on one side of a candidate split where $Y = 1$. The expression $1-p$ in Eq. (11.1) is the prior probability of $Y = 0$. Note that the log to the base 2 is used in Eq. (11.1).

The IG is expressed as the difference between the entropy of all Y values on one side of a candidate split (before) and the entropy on the other side of the split (after), expressed by Eq. (11.2).

$$IG = H(Y)_{\text{before}} - H(Y)_{\text{after}} \quad 11.2$$

This principle can be used to calculate the IG of the optimal split along the range of each factor. This is the approach followed by the MDR method (described below) to generate a short-list of factors to be presented to a predictive algorithm.

Calculation of information gain

The direct way is to calculate the IG for each combination of candidate input features, but that operation can be very time-consuming. An easier way to do it is to use a decision tree algorithm to do it for you, and then continue with further data preparation operations prior to building a model with the list of features selected by the decision tree to train the final model with the selected algorithm.

Gini coefficient (or Gini impurity)

Another method used by some decision trees (e.g., CART) to choose where to split the range for a given branch is the Gini coefficient. The Gini coefficient is not to be confused with the Gini Index used to compare incomes of countries in eco-geographical studies. The Gini coefficient can be calculated by dividing the normalized scales of the features into a number of subranges and summing the subrange relative frequencies for each feature according to Eq. (11.3).

$$\text{Gini coefficient} = 1 - \sum_{k=0}^n \left(\frac{\text{Subrange}[j][k]}{\text{Total frequency}} \right) \quad 11.3$$

where j is the number of features, and k is the number of subranges in the range of a feature. In effect, the Gini coefficient is a measure of the nonhomogeneity in feature frequency of values throughout the range of a feature. The theory is that features with very feature frequencies of values along their ranges are better predictors than those with relatively even feature frequencies (high-frequency homogeneity) along their ranges.

Information gain or Gini coefficient?

The Gini coefficient is simpler to calculate and does not require the $\log_2$ to be calculated. The Gini coefficient

has a closed form solution, while IG does not. For those reasons, the Gini coefficient might be the better choice. Both IG and the Gini coefficient judge the strength of a candidate feature solely by its internal characteristics, not by its effect in the solution.

Caution in using the Gini coefficient for feature selection

A Python script was used to calculate the Gini coefficients to reduce the dimensionality of a modified input data set from the KDD Cup 1998 Challenge. One decision rule that can be applied to the output data is to delete all features with Gini coefficients ≤ 0.5 . When this decision rule was applied to the list of features, the feature RAMNTALL (total amount given) was deleted, but this feature proved to be a relatively important predictor feature, despite its relatively high homogeneity of relative frequency throughout its range (giving it a relatively low Gini coefficient of 0.22). This result highlights the caveat that the final decision of which features should be selected should be guided by multiple feature selection methods.

KNIME examples of feature selection methods

The KNIME predictive analytics tool provides examples of all three types of feature selection methods listed by Kohavi and John (1997).

- Correlation Filter node (a filter method).
- Low Variation Filter node (a filter method).
- Probabilistic Regression node (part of a wrapper method), used in conjunction with a Normalizer Node and a Column Filter node to filter out features with regression coefficients below a threshold value.
- Decision tree attribute statistics report, showing the number of times a feature is used in each of the first three splits to build the decision tree (available in the Random Forest and Tree Ensemble Learner nodes).
- Feature Elimination Node (an embedded method), using a processing loop to iterative delete features with a decision parameter (e.g., Chi-squared value).

Each of the feature selection methods provided by KNIME will be discussed briefly below, followed by several examples of KNIME workflows that can be used to perform feature selection operations.

Low variation filtering (a filter method)

The theory behind low variance filtering is that features with little variance make poor predictors, because there is little basis in feature's range to use to distinguish between target states of values. The Low Variance Filter node is not as "smart" as the Linear Correlation node. It will accept

both Doubles and Integers, but the integer data types are not changed to Doubles before the analysis begins. Therefore, we must filter out the integers before the operation. The reason for this pre-filtering operation is that almost all of the integer features are binary, containing only 1 or 0. In the KDD Cup 1998 data set, the positive TARGET_B binary response occurs only in 5% of the records. A relatively few number of TARGET_B = 1 values may be evaluated as having very low variance, but the feature may be a very important predictor of the response.

Constant value filtering

Sometimes, one or more features may have only one value in them. Some modeling algorithms screen out features with constant values, others don't. You may have to do it manually. KNIME provides a Constant Value Column Filter node used for this purpose.

Correlation filtering (a filter method)

Features that are highly correlated increase the probability of an ill-posed solution. For parametric statistical modeling algorithms (like Regression), an ill-posed problem prevents the matrix inversion operation necessary to solve the regression problem mathematically. For machine learning algorithms like decision trees and neural networks, highly correlated features increase the likelihood of the solution being trapped in a local, rather than a global minimum error condition. For those reasons, it is prudent to delete one member of a highly correlated feature pair. The relevant threshold is set often at 70% correlation (sometimes even at 80%). Deletion of these features also serves as a feature reduction strategy. It is probably safe to perform this feature selection method first, and then use the subsequent KNIME features selection methods on the output. Fig. 11.2 shows a KNIME workflow that employs both constant value filtering and correlation filtering operations performed prior to modeling.

The Column Filter node shown in Fig. 11.2 eliminates a list of features specified by the user, while the Constant Value Column Filter and Correlation Filter node eliminate features according to decision rules. The Constant Value Column Filter eliminated no columns, but the Correlation Filter node eliminated 108 of the input 387 features (columns) with >0.8 correlation coefficients with other variables.

Feature selection with a model (an embedded method)

Some parametric statistical algorithms select the important predictive features in the parameter estimation process (e.g., Logistic Regression). Other algorithms have feature selection methods built into their operations (e.g., Stepwise Regression). KNIME doesn't provide any stepwise regression algorithms, but it does provide a looping structure in which the operation can be performed. His looping capability is one of the most powerful data manipulation features available in KNIME.

Backward Feature Elimination node (an embedded method)

KNIME includes a very powerful stepwise modeling capability that can be used with any of the KNIME modeling algorithms. The feature selection loop shown in Fig. 11.3 can be configured to perform a backward operation, which eliminates iteratively features below a threshold of significance. Or, the node can be configured to perform a forward feature selection process, retaining iteratively those features above a level of significance. The backward configuration of the node is analogous to the operation of the backward stepwise regression algorithm, in that features are eliminated one at a time following a decision rule, and only the features important to the solution are left. The workflow shown in Fig. 11.3 can be collapsed into a *metanode* for reuse in other

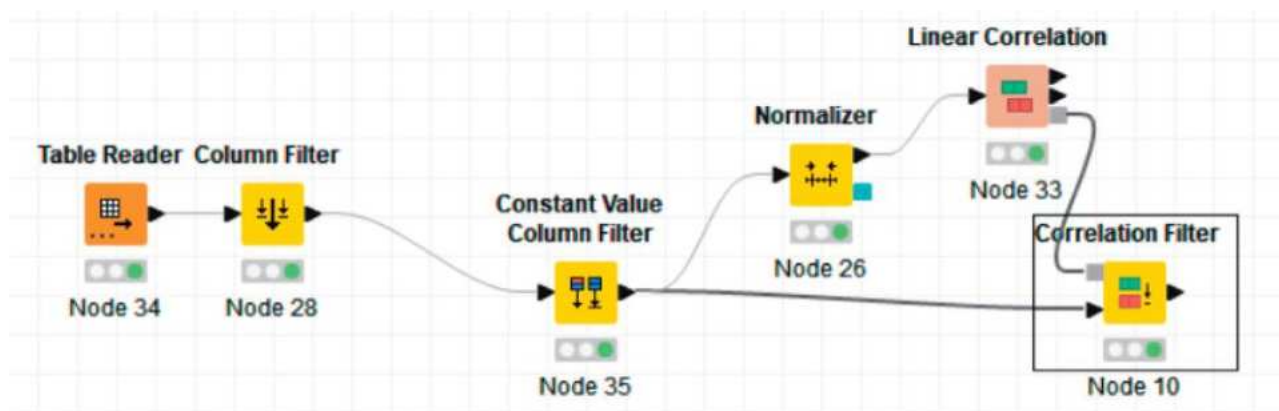


FIG. 11.2 Operation of column filtering, constant column filtering, and correlation filtering operations in KNIME.

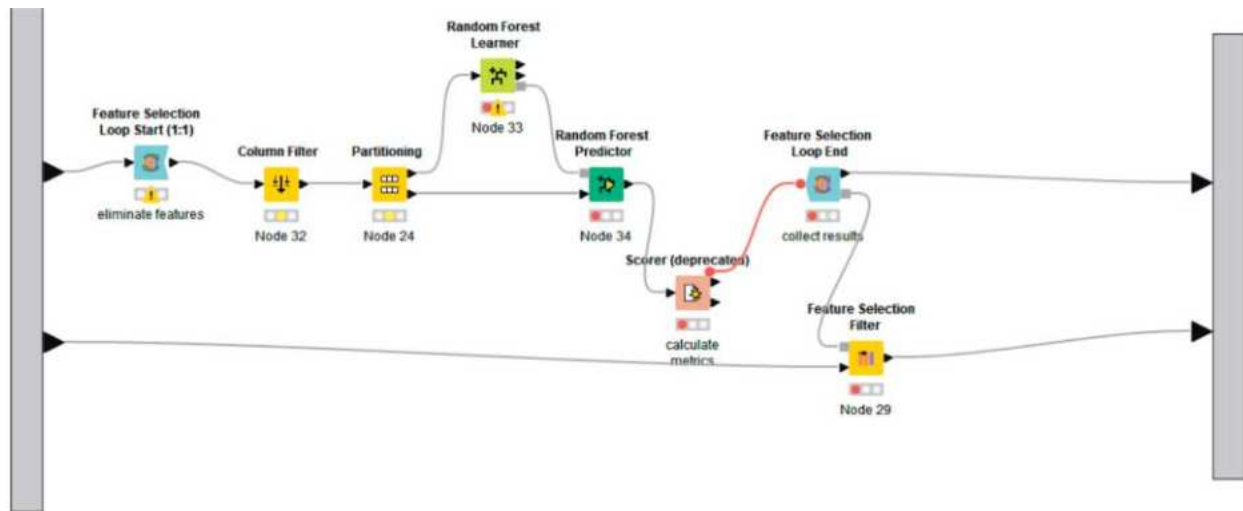


FIG. 11.3 A KNIME feature selection loop configured for backward elimination.

workflows, where it functions like a subroutine in a computer programming language like C++.

Postscript

This chapter may be the most important chapter in the book, because feature selection can control the predictive power of the final model to a large extent. The short-list of features is an extremely valuable output of the data preparation phase of the data analytics process. This short-list can be input to the modeling algorithm to build the final model. We will do this in [Chapter 14](#) and train several models to show the effectiveness of several data preparation operations.

Keywords and phrases

Filtering, Correlation, Dimensionality, Short-list, Hyperparameters, Optimization, Occam's Razor, Feature elimination, Decision surface, Information Gain, Global minimum error.

References

- Columbia University. (2023). Michaelangelo: Selected Poems. http://efaidnbmnnnibpajpcglclefindmkaj/https://arthum.college.columbia.edu/sites/default/files/PDFs/arthum_michel_reader.pdfarthum_michel_reader.pdf. Accessed 6.17.2023.
- Creighton, R. (1980). In R. Linscott (Ed.), *Complete poems and selected letters of Michelangelo*. Princeton University Press. https://arthum.college.columbia.edu/sites/default/files/PDFs/arthum_michel_reader.pdf.
- Kohavi, R., & John, G. H. (1997). Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97(1–2), 273–324.
- Moore, J., Gilbert, J., Tsai, C., Chiang, F., Holden, T., Barney, N., & White, B. (2006). A flexible computational framework for detecting, characterizing, and interpreting statistical patterns of epistasis in genetic studies of human disease susceptibility. *Journal of Theoretical Biology*, 241(2006), 252–261.
- Vaughan, C. (2016). *Michelangelo's notebooks: the poetry, letters, and art of the great master* (pp. 384). New York, NY: Black Dog & Leventhal Publ.

Additional bibliography for feature selection

- Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., & Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences of the USA*, 96(12), 6745–6750.
- Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees* (pp. 9). New York: Chapman & Hall.
- Doak, J. (1992). *An evaluation of feature selection methods and their application to computer security*, Tech. Rept. CSE-92-1 8. CA: University of California at Davis.
- Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96, 1348–1359.
- Fan, J., & Lv, J. (2008). Sure independence screening for ultra-high dimensional feature space. *Journal of the Royal Statistical Society. Series B*, 70(5), 849–911.
- Groemping, U. (2006). Relative Importance for Linear Regression in R: The Package relaimpo. *Journal of Statistical Software*, 17, 1. Issue.
- Grömping, U. (2007). Estimators of Relative Importance in Linear Regression Based on Variance Decomposition. *The American Statistician*, 61, 139–147.
- Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. N. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1–3), 389–422.
- Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 1157–1182.
- Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning*. New York: Springer, Berlin; Heidelberg.
- Kim, K., & Rendell, L. A. (1992). A practical approach to feature selection: *Proceedings 9th International Conference on Machine Learning, Aberdeen, Scotland*. Los Altos, CA: Morgan Kaufmann.
- Rakotomamonjy, A. (2003). Variable selection using SVM-based criteria. *Journal of Machine Learning Research*, 3, 1357–1370.
- Vafai, H., & De Jong, K. (1993). Robust feature selection algorithms: *Proceedings 5th International Conference on Tools with Artificial Intelligence* (pp. 356–363). Rockville, MD: IEEE Computer Society Press.

A data preparation cookbook

Preamble

In previous chapters, we focused on the performance of various major aspects of predictive analytical modeling. Novice modelers might consider taking shortcuts in performing the tasks listed in this chapter, but that is likely to lead to the development of an inadequate model or to no good model at all. It is a commonly held maxim in predictive analytics that 60%–90% of the project time will be consumed by data preparation tasks. In this chapter, we will collect the “pieces” of data preparation into one document for ease of use. This document will present a list of tasks that are performed in the first four phases of cross-industry standard process for data mining (CRISP-DM), which may be the most critical part of the project. You may wonder why this chapter on data preparation includes tasks in the CRISP-DM process phases prior to and following the Data Preparation phase. The reason for including these tasks is that they are the initial preparation operations for the data preparation and the basis for loop-back to data preparation from the Modeling phase. If the tasks preliminary to the final modeling task are not performed properly, the data preparation operations may be largely a waste of time.

Introduction

The question arises often with novice analysts about the proper process to follow in a predictive analytics project. The big-picture answer is expressed by the CRISP-DM process model, shown in Fig. 12.1. This diagram was introduced in Chapter 3, and it will be explained further in this chapter.

The primary phases involved with data preparation are business understanding, data understanding, data preparation, and modeling. The modeling and evaluation phases are included in a discussion of data preparation because preliminary modeling is a *part* of the normal sequence of events in successful data preparation. This might be considered as a circular and iterative process designed to increase the merits of the model until an

“optimal model” is achieved. Thus, the CRISP-DM process model appears as a sequence of processes organized in an iterative processing format, but under the “covers” it has many feedback loops in the sequence of phases in the project (in addition to those shown in Fig. 12.1). This chapter will present a common sequence of operations in building a customer relationship management (CRM) model for customer response, such as is discussed in greater detail in Chapter 21. You can adapt this approach to fit any analytical project application. At the end of the chapter, we will present some of the common errors committed in the data preparation process to provide further guidance to help you perform successful analytics projects. You might review the discussion of the CRISP-DM process model described in Chapter 3 before proceeding to implement the tasks listed in this cookbook. Also, it is recommended very highly that you compose the tasks in the cookbook for data preparation to form a project plan in some planning tool like Microsoft Project. *The goal is to “plan the work,” then “work the plan.”*

1. CRISP-DM phase I: business understanding

Before you do any data or analytical task, you must understand clearly and define the business goal(s) of the project. It is a *big* mistake to proceed ahead without these goals set clearly in mind. Otherwise, you will be guilty of the target shooter’s common mistake of following a sequence of Ready – Fire – Aim. Before we “fire” off the data and analytical processes, we have to know clearly what goals the projects aim to achieve. This is the most important piece of initial “data” to prepare for the successful completion of the project. Tasks are listed in the approximate order that they might be implemented most efficiently. The order of these tasks, however, might change for other projects.

Major tasks in this phase include:

- **Task 1a: Define the business goals of the project.** What are the questions you want to answer and do the answers correspond with the business needs that are driving the project? Proceeding ahead

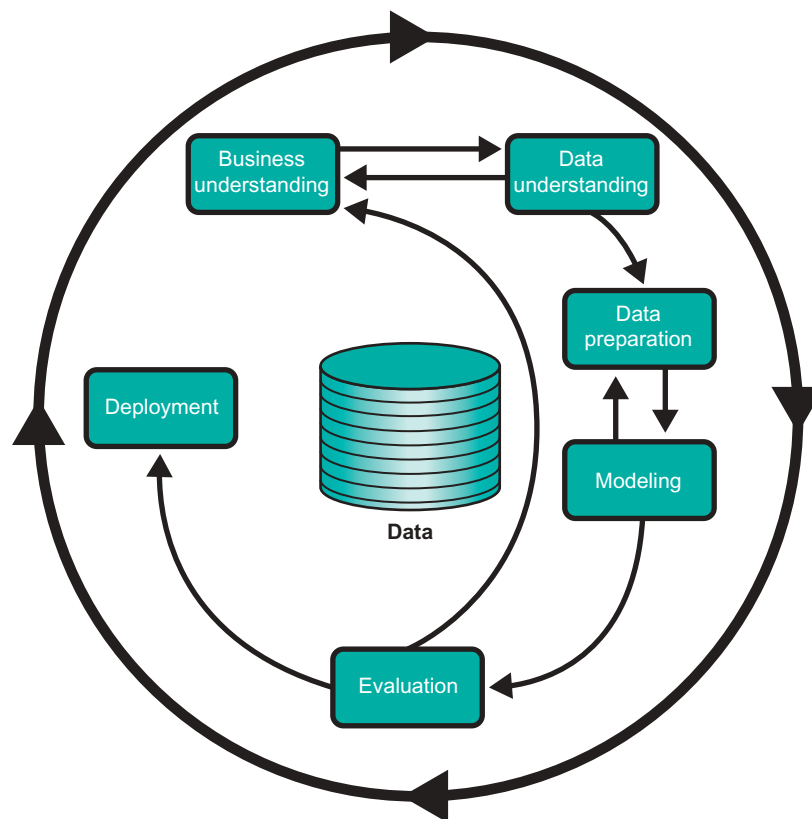


FIG. 12.1 The cross-industry standard process for data mining (CRISP-DM).

without the answer to this question will almost certainly lead to the failure of the project.

- **Task 1b: Identify the business stakeholders.** If you try to produce a product from the analytical project that will affect the operations of various staff functions in the company, you had better include their representatives in the project discussions from the very beginning. Failure to do that may “tick-off” the staff members in these functionalities (e.g., marketing) and make it very awkward and time-consuming to implement.
- **Task 1c: Define working relationships.** In a CRM analytics project, the two most important functionalities in the company are marketing and information technology (IT). Representatives of IT are the local “owners” of the data you will want to access, and they will have a strong proprietary interest in it. IT staff are necessary for any project to explain what data fields exist, where they are, what restrictions are associated with their use, and how to extract them. The analytic modeler must develop good relations with IT staff, otherwise, an adversarial relationship between the modeler and IT staff is likely to develop. They are your “food sources,” and you don’t want to close down the distribution channel.
- **Task 1d Define the analytical goals of the project.**
- The analytical goals of the project must be formed initially in discussions with all stakeholders, and defined clearly in their minds and the mind of the modeler. Otherwise, you are likely to discover too late that “you can’t get there from here” following the project plan.
- **Task 1e: Define acceptance criteria.** At the beginning of the project, you must define how you can “know that we are good,” both from an analytical standpoint and in the view of the stakeholder also. Failure to do this may delay the project deliverables significantly, or the project will fail to meet the needs of the stakeholders.
- **Task 1f: Define the service level agreement (SLA) of the project.** This requirement is very easy to overlook. The marketers and IT staff may be very confident that the model to be developed will be very useful in the company. The implementers of the model, however, might be unable to use it in the form provided. For example, a customer churn project for a large Midwestern bank chain required that the bank relationship managers have two weeks to contact high-probability churners to convince them to stay in the bank after it switched from free checking to fee-based checking. That requirement

prevented the analyst from using any temporal data as predictors in the model that were less than 2 weeks old (e.g., number of past weekly deposits and withdrawals), lest the customers churned before the managers could contact them. The model delivered required a 2-week forecast horizon to satisfy this SLA, or it would be unusable by the bank managers. We must make sure that we are not guilty of defining a “square” solution to solve a “round” problem.

- **Task 1g: Define the target variable of the analysis.** The target variable may appear to the analysts and the marketing/IT staff to be appropriate, but it may not satisfy the requirements of the upper management. For example, the target variable for a customer churn model for a South American wireless phone company had to be defined as Target = 1 if there was a greater than 70% decline in minutes of use (MOU) over the previous two billing periods, else as Target = 0. The upper management wanted a model that would be more diagnostic of the development of the churn “signal” in their records rather than just focusing on the fact of churn among past customers. We must make sure that the product of the project will hit the “target” as required, not some other target.
- **Task 1h: Create the project plan with milestones and an appropriate timeline.** Project planning may be attempted by those not formally trained in the art, but this is almost always a mistake. In addition to a data analyst on the modeling team, there should be a project manager, who is adept in project planning, preferably with a Professional Project Manager (PMP) certificate from the Project Management Institute (PMI). PMI professionals are trained to include all necessary elements in a project plan, and to avoid process mistakes that can prove very costly and time-consuming.

2. CRISP-DM phase II: data understanding

Being familiar with a data domain does not equate to understanding data in it in the context of the project. After the business aspects of the analytical project are worked out and documented clearly, attention can be turned to identifying, accessing, integrating, and understanding the available data. The major tasks in this phase of the modeling project include the following tasks.

- **Task 2a: Access your data.** This task is composed of a number of subtasks that must be orchestrated to produce the customer analytic record (CAR). Not all of these subtasks will be appropriate for all modeling projects, but the basic elements of data access are shown in Fig. 12.2.

Sub-Data Access Analysis

- Identify data sources (database, spreadsheet, application storage, etc.)
- Resolve ownership issues (gain access permission)
- Resolve data security issues (e.g., HIPAA, personal identity)

Sub-Data Extract Design

- Determine practical problems with data extraction
- Evaluate ODBC versus query-based data extraction options
- Write the data extract scripts to assemble the data from data sources

Sub-Assemble Data into the CAR

- Syncrize data types and formats (e.g., numbers and text, delimited files and spreadsheets)
- Define specific data source documents/tables from which data will be extracted
- **Task 2b: Go-vs-NoGo decision.** You must decide whether the nature of the data available is suitable for analysis of the intended outcome and issue a Go_No-go decision.

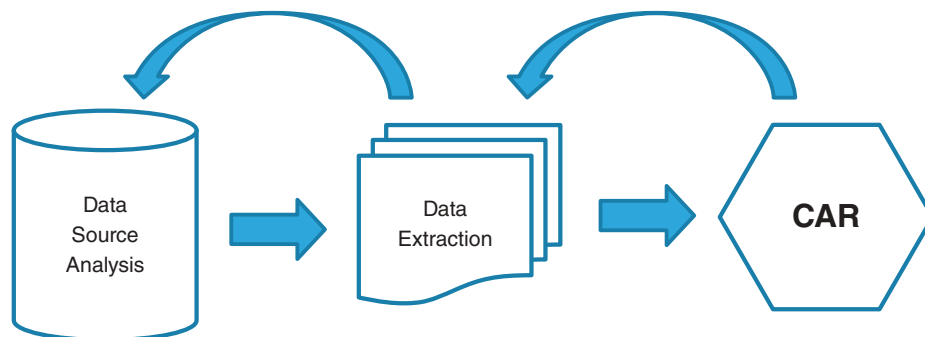


FIG. 12.2 Basic subtasks of data access.

- **Task 2c: Enhancing and enriching the data.** The data fields provided by company data sources represent only a few of the dimensions of customer information that could be used to define the pattern of the target variable (e.g., customer attrition or churn). If funds are available to purchase them, there are abundant sources of external information about customers that could prove valuable in defining the target pattern more fully. This value is absolutely necessary for building customer acquisition models, where you have no internal data to use. When internal data is available, incorporating this external data into the CAR is an example of a feedback loop between subtasks in this phase of analysis. These external data sources include a rich variety of information for defining customer locations, attitudes, performance in credit operations, and general public information including the following list.
 - Demographic data
 - Firmographic data
 - Psychographic data
 - Credit data
 - Census data
 - Other public domain data
- **Task 2d: Characterize and describe your data.** Before you begin to use the data elements for any operation, you should describe them, and perform several subtask operations on them to diagnose any potential problems associated with their use (e.g., multicollinearity caused by using highly correlated variables in the ensuing analysis). Common subtasks include:
 - Calculate descriptive statistics for all variables. Information about means, medians, standard deviations, and other metrics can be useful in deciding which variables to use for predictors in the training of the model. For example, the ratio of the mean/standard deviation (coefficient of variation) is a measure of the useful variation in the variable, which might be used in the feature selection process to filter out variables with little variation. For example, KNIME provides the Low Variance Filter node to perform this operation. These low variation variables are very unlikely to be good predictors of the target variable.
 - Categorize input variables into continuous (numbers) and discrete (categories). This subtask must be performed to understand what data preparation operations must be performed on each variable. For parametric statistical analysis (e.g., Linear Regression, which uses only numbers), categorical variables must be converted to “dummy” variables (containing either a 1 or a 0) indicating the presence of a value of a specific category code in a given row of data in the CAR. This operation will be performed in the data preparation phase.
 - Identify/define the target variable(s) in terms of available data elements. This subtask may appear to be trivial, but management requirements might define this definition in a rather complicated way.
 - Look for outliers and decide how to handle them in the data preparation phase. For some projects, outliers could be recoded to the highest value or the median value, in order to prevent their undue influence on the model solution. In other projects, large outlier values might be removed from the data set. In some projects (e.g., fraud analysis), outliers are the target of interest and should be retained.
 - For parametric statistical modeling algorithms (e.g., Logistic Regression)
 - Graph data distributions.
 - Determine specific transformation to use for each predictor variable to convert the data distribution to a form as close to the Normal curve as possible.
 - Derive any obvious interaction variables.
 - These operations are planned in this phase and performed in the data preparation phase.
 - Calculate correlation coefficients. This correlation coefficients generated by this subtask can be used to determine which variables are correlated to each other. Only one variable of a group of highly correlated variables should be used; the other one should be deleted.
 - Determine which variables have missing data, and design a strategy for missing data handling for each variable with missing values. Missing data imputation will be performed in the data preparation phase. This subtask is particularly important, because many machine learning algorithms used for modeling can delete the entire row of data that contains a missing value in even just one of the variables within that row (case).
 - Analysis for data bias

Types of bias:

 - Sample bias – different samples of a larger data population might yield very different analysis results. Make sure that your sample is representative of the population, as far as you can determine.
 - Experimental bias – A study of performance of high school students in a certain high school might be biased against dropouts.
 - Measurement bias – If survey interviewers

included deaths that occurred before the time period of the study, it would cause an overestimate of the mortality rate among the study population.

- Intentional bias – The problem is more pernicious, because it may be unexpected, and it is very hard to assess. An experimenter in the mortality study above might limit the population interviewed to only those with healthy diets, because of the desire to prove a “point.”
- Determine if data set samples (subsamples of the entire dataset) should be performed prior to analysis Reasons:
 - To reduce data volume.
 - To incorporate into any planned resampling operations (e.g., bagging or cross-validation).
 - Develop plans for any under-sampling or over-sampling strategy to be employed to correct for effects related to rare target values during the data preparation operations.
- If you plan to include resampling operations in the modeling phase, add them to the project plan.

3. CRISP-DM phase III: data preparation

This phase of the process model is often the most crucial one to do properly. Because of the diversity of tasks in this phase and the complexity of some of the operations, it may consume up to 90% of the project time.

- **Task 3a: Perform any planned sampling regime** Types of sampling regimes
 - Random sampling
Performed to draw a representative group of data rows from a larger data population.
 - Stratified sampling
Performed when a population is composed of two more groups that can be grouped together according to the same criteria (e.g., geographical location).
 - Over-sampling and under-sampling
Performed to produce the same number of rows for each target class. See [Chapter 4](#) for more information on this topic.
 - Assign case weights or prior probabilities to specific target classes, instead of balancing data sets with a rare target class. Some analytical tools can balance data sets with rare target classes by using differential weights or using probability of occurrence of the target classes (prior probabilities) to govern the effect of the classification operation. See [Chapter 4](#) for more information.
 - Recognize that sampling operations in the Modeling phase (e.g., data partitioning) may generate a training data subset that is not a good

representative of the entire data set. This problem can be remedied by constructing a bagging processing architecture in the training of the model, or a cross-validation processing architecture in model evaluation. See [Chapter 14](#) for more details on how to construct these modeling architectures.

- **Task 3b: Data cleaning** Types of cleaning operations:
 - Deletion of “garbage” values
 - Data tables from which you extract data might have inappropriate records in them, which cannot be used for modeling (e.g., training records).
 - Deletion of sensitive information
 - Data sets intended for analysis might contain certain data elements that are confidential or protected by law (e.g., Social Security Number). These data elements cannot be used as predictor variables in a model.
- **Task 3c: Data reduction** Types of data reduction operations:
 - Reduction of dimensionality (or feature selection). This is one of the most important operations in data preparation, therefore, an entire chapter is devoted to feature selection ([Chapter 11](#)). Most analytical packages provide one or more feature reduction tools. Some of these tools were described in greater detail in [Chapter 11](#); review it if necessary. For the purposes of this cookbook document, however, the primary reasons to reduce the number of features in an analysis can be summarized as: (1) reduce the mathematical complexity of the feature space, enabling the modeling algorithm to work more efficiently, and (2) to reduce the “noise” of the target signal.
 - Reduction of numerosity. This operation refers to the reduction of the number of distinct numeric values. A modeling algorithm is trained to recognize a pattern in the numbers or categories of the data set. The process of defining the pattern to the “senses” if the algorithm can be increased significantly in efficiency and speed of operation by reducing the number of distinct numeric values it must process. Neural network algorithms accept categorical inputs, but they must transform the categories into numbers before they can be processed. Decision trees can work directly with category names, but their efficiency in building and trimming branches of the tree can be enhanced by reducing the numerosity also. Common approaches:
 - Aggregating. Data from similar variables can be aggregated to generate a generalization of the group of inputs.

- Clustering.
Groups of records can be grouped to reduce the data volume.
- Sampling.
A 10% to 20% sample of the data population may provide sufficient rows to build the pattern for the algorithm to recognize.
- Discretization. The narrow definition of this term covers only the operation of binning of numeric values to move from discrete values along a range of a variable to groups of values in specific subranges of the variable (e.g., 0–9, 10–19, 20–29 ...). This term is sometimes applied to the process of derivation of “dummy” variables from categorical variables.
- Development of concept hierarchy generation. For example, colleges, high schools, middle schools, and grammar schools compose a concept hierarchy for schools. The specific concept in this case might relate to the sophistication of curriculum materials. The fact that a given variable refers to data from high school students might not matter nearly as much as the information about where in the hierarchy of curriculum complexity it stands.

Task 3d: Standardization. Standardization or normalization of data is necessary for parametric statistical algorithms, but not for machine learning algorithms. Normalization is the process of transforming data values to span a specific range (e.g., 0 to 1). Standardization is the process of dividing each value by the standard deviation. The most common form of standardization is to calculate the Z-score for each numeric value. This transformation converts the variable to a range between $-\infty$ and $+\infty$, in which 99.5% of the values (in a normal data distribution) range between -3 and $+3$ values for the Z-score (this range also corresponds to a departure of values within three standard deviations from the mean). The Z-score standardization method is compatible with many parametric statistical analysis methods based on inferences from the 95% confidence level (corresponding to a range between -2 and $+2$ Z-score units), or the 99% confidence level (corresponding to a range between -3 and $+3$ Z-score units).

Task 3e: Recoding. Some variables may include codes that refer to two different coding systems. The modeler must recode code values on the old coding schema to the new schema. Another example of the need for recoding is to replace “garbage” data (e.g., data entry errors) with blank filler codes.

Task 3f: Filtering. Use of some rows in a data set might not be legally permissible to use in predictive models. For example, the Gramm-Leach-Bliley Act of 1999 constrains banks from using customer-specific data in public reports, but data summaries across customers are permitted.

Task 3g: Missing value imputation. This task is described in detail in [Chapter 8](#) on data cleaning, therefore, only short summaries of common techniques used are presented here. The safest approach for handling missing values is to delete variables with many missing values or rows with missing values in many variables. If you decide to impute missing values, there are three types of operations to impute missing values in variables:

- With constants. This operation fills all missing values for a list of variables in the data stream with a specified constant, either a number or a text code.
- With formulas. The mathematical evaluation of a formula can be used to fill in missing values, if relationships between the missing value and the elements of the formula are known.
- With models. A simple model can be used to impute missing values, using other variables as predictor. For example, Age, home ownership, and Zip Code might be useful predictors of Income.

Task 3h: Derived variables

- Summarization. In very detailed data domains (e.g., retail transactions and call detail phone records), the analysis grain of detail is coarser than the raw data grain. Call detail phone records at the telecommunications switch consist of link paths, start and end times of the phone call converted to MOU. If the analysis time-grain is specified at the weekly level, the MOUs for all calls in a week must be aggregated for each customer. Another example of data summarization in a telecommunications data domain is to calculate the means and standard deviations for certain variables for each week.
- Complex calculations. In a banking data domain, it may be necessary to calculate the distance from prospective customers and the nearest bank branch, using spherical geometry.
- Dummy variables. Dummy variables are derived for each class of a categorical variable, installing a 1 for a given row if the class is present, else 0. Dummy variables are discussed more fully in [Chapter 4](#).
- Distribution transformations. Parametric statistical predictive algorithms (e.g., Linear Regression) assume that the distribution of each

variable is normal. The degree to which the data distribution of a variable departs from normality is proportional to the amount of error that may be added to the solution. Various transformation functions (e.g., square, square root, natural log, etc.) can be applied to the data distributions until the transformed data distribution approaches normality.

- Deriving lag variables with temporal data. Temporal data is one of the most challenging problems encountered by analysts. If the goal is to forecast sales, for example, traditional time-series analysis algorithms can be used effectively (e.g., ARIMA). If the goal is to relate a customer response in the future to a series of customer events in the past, these events must be copied to the row in the CAR for a given customer, and renamed for the time periods in the past to which they pertain. Such a variable is a temporal abstraction, and is commonly called a *lag variable*. Lag variables can be very predictive in many customer data domains, such as telecommunications (where they were used effectively beginning in the late 1990s). They can also be effect predictors in Insurance, Credit, and Banking data domains. Lag variable can express the customer's intent to respond several weeks ahead of time. For example, a banking model built with a number of 2-week lag variables could be used to identify those customers who intend to leave the bank ("churn") soon, providing their relationship manager sufficient time to contact them and convince them to remain as customers of the bank. This approach was followed in developing a model for a large Texas bank to predict which customers were likely to churn when the bank moved from free checking to fee-based checking operations. Fig. 12.3 displays an acrostic formed by orienting customer loyalty issues to spell the acrostic *CUSTOMER*.

This organization of words is reflected in an analogous organization of temporal records shown in Fig. 12.4.

Fig. 12.4 displays fields in six customer records from a telecommunications company lined up like beads on an abacus. The data on the left abacus represent information stored for monthly call duration extracted from multiple records in the database. This arrangement is similar to the format of the data extracted from databases into flat files to be submitted to the modeling tool for analysis. In the default configuration (left abacus), the yellow beads (a given state in the time-series) are scattered all over the abacus. The diagram on the right of Fig. 12.4 shows the rearranged data. Now, the yellow beads are lined up. The pattern emerges to the physical senses of our eyes and likewise to the mathematical senses of the modeling tool.

Service
Quality
Satisfaction
Loyalty
Support
Communication
Feedback
Trust

FIG. 12.3 Focus on the customer is the basis for CRM modeling programs.

Task 3i: Handling of outliers. This task requires the initial decision of whether or not to leave outliers in the data set. In many cases, it is preferable to recode outliers to some value (e.g., the largest) to focus the modeling algorithm on the normal range of the variable in its training operation.

4. CRISP-DM phase IV: modeling

Task 4a: Preliminary modeling operations to test the effectiveness of certain data preparation operations.

This task is actually a group of tasks. They don't appear at first consideration to be a part of data preparation. They are, however, among the most important type of operations in the data preparation process. There are many data preparation operations on variables (manipulations) and entire data sets (conditionings) which work better than others for a given data set. The only way to find the right operation for a given data set is to try various operations and test their effectiveness with a preliminary model. If a given operation is not effective in a model, it can be dropped from the processing sequence. Preliminary modeling operations do not include model enhancements which might be performed to build the optimum model for a given project.

Examples of data preparation operations that might be tested include:

- Standardization.
- Undersampling operation to compensate for a rare target class.

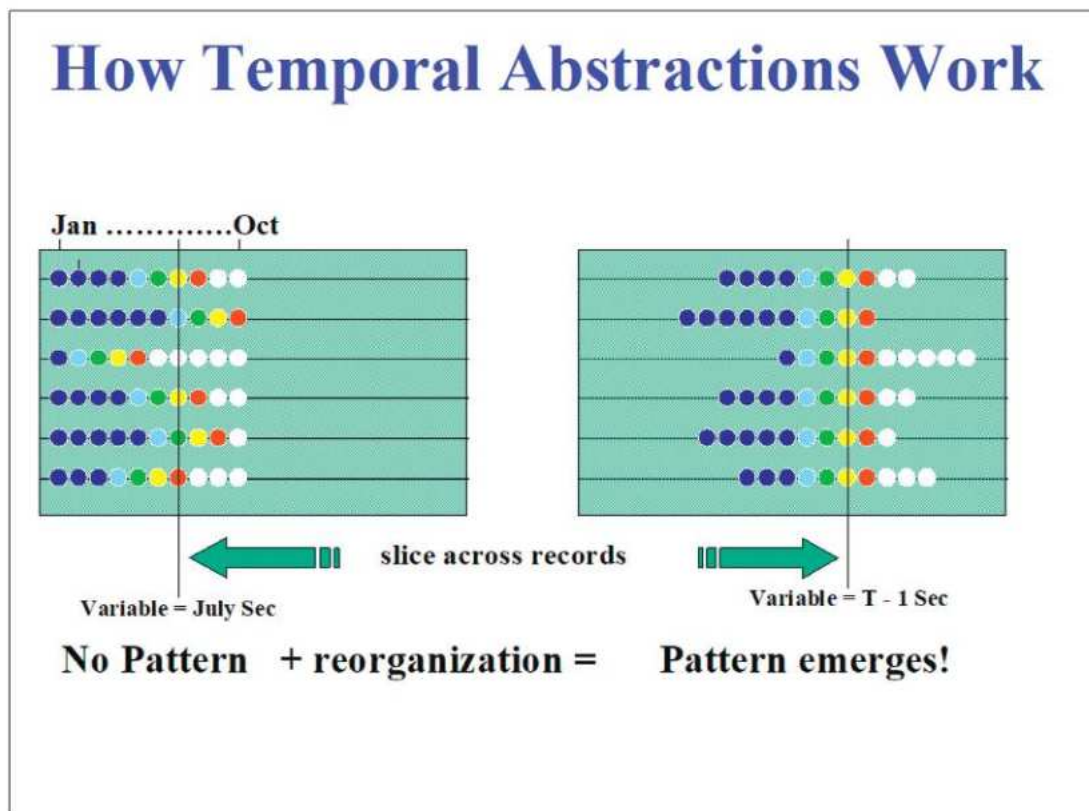


FIG. 12.4 Reorganization of response date records to show the pattern of yellow dots representing the $t-1$ record, where $t = red$.

- Oversampling operation to compensate for a rare target class.
- Data segmentation operations – It was discovered in a wireless phone company that urban customers had a very different calling behavior pattern than rural callers. Separate churn models were built for urban and rural customers. Otherwise, the very different calling behavior signals would confuse the pattern of the target variable in the modeling process.

The cookbook for data preparation presented above may be followed with an approach that is too closely associated with the details of performing the relevant tasks in an analytical project. It is helpful, however, to review some mistakes that are made commonly in analytics projects, with a view in mind to avoid them like the plague while performing the project tasks.

18 common mistakes in data preparation in predictive analytics projects

The following common mistakes have occurred in one project or another in the past of the authors' experience. The first 10 are ordered roughly in terms of their estimated importance or impact on the fidelity of the

results. The last eight are judged to be of lesser importance, yet still they occur often enough to list them here.

1. Failure to fill all missing values.

This can be a very pernicious error when modeling with machine learning tools, because the algorithm you are using may delete the entire row if there is a missing value in any column. Failure to fill all missing values before modeling might reduce the data in the structure submitted to the algorithm to a level that will not serve to train a good model.

2. Modeling past response with future data.

The most common form of this error includes a predictor variable (the "past") that is part of the definition of the target (the "future" response). For example, you can't use an Exit_Reason variable as a predictor if the code pertains to the reason the customer left the company ("churned"). All churn customers will have values in the Exit_Reason variable, and those who are still in the company will not. The goal in a churn model is to predict a future response with data variables from the past, but the Exit_Reason variable describes the future response, and cannot be used to predict its occurrence. Whenever a perfect model is built, it is most likely that some information related to the "future" response has been used to predict that response.

3. Failure to evaluate variables sufficiently.

Examples of this type of error include: (1) using the wrong method of variable selection; (2) using only one variable selection method; and (3) failure to include the right variables in the short-list of variables submitted to the modeling algorithm.

The final list of variables submitted to the modeling algorithm (the short-list) is a very important part of the modeling project. This list is very valuable to the project and in the marketplace at large. For example, a small company worked for a year to generate this list from thousands of candidate variables in a large client company. The client refused to pay for it; the small company sued, and won a judgment of \$17 million for the short-list.

4. Failure to partition data sets into three subsets.

Some analytical tools provide techniques for dividing (partitioning) an input data set into a number of subsets prior to analysis. Some tools generate only two data sets: the Training data set and the Testing data set. These two partitions are used in the training and model testing processes of many machine learning algorithms. The performance of the model is usually defined according to how well the model predicts the Testing data set. This is not the best way to do it. Alternately, a third data set should be partitioned and used to model evaluation, instead of the Testing data set. The reason is that model evaluation with the Testing data set is *tautologous*. A tautology is a definition of something in terms of itself. The model building process uses the Testing data set to choose the best model, and it should *not* be used to evaluate its performance.

5. Using the record number or an ID number as a predictor.

This is an easy error to commit. Make sure that all identifying columns of the data set are excluded from the list of predictor variables submitted to the modeling algorithm, including ID, Name, and street address. For some applications, however, city name and Zip Code can be valuable predictor variables (they are surrogates for geographical location).

6. Failure to consider interactions and deriving new variables to express them.

The assumption in parametric statistical analysis is that the effects of all predictors are independent. This assumption is violated significantly in many data sets. These interactions might be important predictors. Consider a very simple example of length, width, and height. Independently, they might have very little relationship to a model, but the product of all three (Volume) might be very predictive.

7. Failure to devote enough effort to data transformations and deriving new variables.

Even though certain variables may be used currently and serve as predictor variables in their

own right, both logical and mathematical combinations of them can be even more predictive. Consider variables NUMPROM (number of marketing promotions) and NGIFT (number of donations) in the 1998 KDD-Cup Competition data set (available in the UC-Irvine Machine Learning Archive). Alone, these variables are valid but are not important predictors. The ratio of them (NGIFT/NUMPROM) represents the frequency of donation (PROMO_FREQ), and it is very predictive of the binary target for donation.

8. Forgetting about outliers.

In most applications, values beyond three standard deviations from the mean happen only about 0.5% of the time (in a normal distribution). If a value is far beyond three standard deviations, it is most likely an anomaly or an error. We should not model on errors or anomalies in the data. We can clarify the vast bulk of the target “signal” by removing outliers (by recoding them). Yes, it deletes data, but the benefit of removing the outliers will in most cases far exceed any effects of this data loss. In some cases, outliers are the target of interest (e.g., in fraud, intrusions, etc.) and should be retained.

9. Failure to normalize or standardize numerical inputs for parametric statistical algorithms.

Parametric statistical algorithms require that a number of assumptions be satisfied (e.g., Normality, Independency, Linearity). Another assumption is that the scales of all predictor variables are relatively similar. If one variable has a scale that is significantly greater than the other variables, the estimation of parameters will be biased toward the parameter with the greater scale. The solution to this problem is to convert the data to a common scale. This can be performed with a function. The most common function is to use the Z-transform function, which will convert scales to a range of $-\infty$ to $+\infty$, but 99.5% of the values will lie between -3 and $+3$. Another approach is to “normalize” the data ranges to 0 to 1.0.

Machine learning algorithms (e.g., neural nets, decision trees, SVMs, and Bayesian classifiers) don’t require standardization, but most models will be improved by using standardized data.

10. Using biased samples.

A sample of public high school students for the use of drugs can be very biased because private schoolers, home-schoolers, and dropouts are not included in the sample. The resulting sample is biased toward those students that are successful in a public school, and may not be representative of students in the population at large. Policies formed on the basis of the analysis of this biased sample may lead to invalid conclusions in many cases. Another form of sampling bias is when data are collected for only positive

results. You might receive 100 applications for a service, and reject 50 of them initially. But, what if the initial screening is faulty? You might expand your analysis to include the 50 who were rejected initially to see if any of them are potentially good customers otherwise. In the Credit industry, this operation is called “Reject inference.” Some of the potential customers rejected initially can be good customers after all.

11. Ignoring temporal data, and failing to derive temporal abstractions.

It is difficult to relate temporal variables directly to a response variable (the target). Traditional time-series analyses model only the target “signal,” and do not use predictor variables. Temporal information can be analyzed only after successive rows of temporal data (in a time-series) are abstracted and related to the target variable. This is performed by deriving “lag” variables, in which the effect of change in the target lags behind the influence of the variable by some time period. See [Chapter 14](#) for more information on this operation.

12. Using historical data that is not accurate.

For example, customers may have moved from a city location to a suburban location part way through the historical sequence of data for the customer. In this case, the geographic location of all records should be standardized to City or Suburban, depending on the goal of the model.

13. Ignoring historical changes in codes.

This error may appear to be similar to #12 above, but the context is very different. Sometimes, a coding scheme for a given variable may change during the historical course of data collection. This mixture of codes for the same thing must be reconciled before analysis can proceed.

14. Including variables which have changed in the past, in format or codes, or which are likely to change in the future.

Error #6 above refers to previous coding changes. These changes may happen in the future also. For example, you might be tempted to include in the model a predictor whose data format or code is expected to change in the future. One example might be a predictor in studies of patients in a medical office that expects to change to electronic medical records (EMRs) in the near future.

15. Failure to remove duplicate records.

The initial error is failing to scan for duplicate records. Some analytical tools provide capabilities for scanning for duplicate records. Another strategy for finding duplicate records is to do some multi-level sorts of your data sets to see if multiple records sort out together, and appear to be duplicates.

16. Including retired fields as predictors.

Occasionally, certain fields in a database might not be used in business anymore. You must be careful

to qualify each data field to be included in the list of candidate predictors to make sure that each one is used currently.

17. Confusing correlation with causation.

A principle that is drummed into students in their statistics classes is that correlation does not imply causality. A good example of this is seen in a study in *Nature* magazine in 1999 which showed that children who sleep with a light on are more likely to develop myopia in later life ([Quinn et al., 1999](#)). One conclusion that might be made from this data is that sleeping with a light on causes myopia. However, a more recent study showed that children of myopic parents had a high likelihood of developing myopia ([Ohio State University, 2000](#)). It is possible that myopic parents left lights on in children’s rooms to be able to see them while attending them at night. Analysis of business data correlations might follow a similar pattern, and we might be tempted to draw erroneous conclusions about causality.

18. Thinking that failure is not an option.

Trial and error is one of the most tried and true scientific methods used. One industry management expert promote the concept of “Fast Failures” ([Peters, 1988](#)).

Postscript

The steps in data preparation presented above are listed in the general order in which they should be performed. In practice, however, there is much feedback of information to previous steps in the sequence. Errors also can be made but not noticed until you perform some later step; you have to go back and fix them. Just like in the CRISP-DM overall process model presented in [Chapter 3](#), data preparation follows an iterative process with feedback.

Keywords and phrases

Modeling, Task, Phase, Information, Predictions, Algorithms, Data preparation, Missing values, Parametric statistical, Dummy variables, Temporal abstractions, Standard deviation.

References

- Ohio State University Research News (March 9, 2000). Night lights don’t lead to nearsightedness. <https://news.osu.edu/night-lights-dont-lead-to-nearsightedness-study-suggests/> Accessed 1.15.2024.
- Peters, T. (1988). *Thriving on chaos – handbook for a management revolution*. Harper Perennial. 736 p.
- Quinn, Graham E., Shin, Chai H., Maguire, Maureen G., & Stone, Richard A. (May 1999). “Myopia and ambient lighting at night.” *Nature*, 399(6732), 113–114. <http://doi.org/10.1038/20094>. PMID 10335839.

Modeling

The general focus of this part of the book is on modeling and includes discussions of many different aspects of building, evaluating, and enhancing an analytical model.

Building analytical models

[Chapter 13](#) and [14](#) discuss the characteristics of analytical algorithms and how to use them to build analytical models. These two aspects of modeling must work hand in hand. [Chapter 13](#) focuses on the three most common types of predictive analytical models: (1) classification, (2) numerical prediction with regression-type algorithms, and (3) clustering. The emphasis of this chapter, however, is on classification-type and regression-type models because they constitute the vast majority of analytical models used in customer relationship management (CRM).

[Chapter 14](#) focuses on the operations of building an analytical model. This is where the “rubber” of data preparation meets the “road” of model building. The point is made that this process appears to be linear; it is not. The logical flow is iterative, following several feedback loops with various data preparation operations. The reason for these feedback loops is that preliminary models often reveal a need to do more data preparation before continuing. For example, analysis of model output might reveal that there are probably outlier values in one or more variables that must be handled (recoded or eliminated). These outlier handling operations are necessary to build a model that reflects the “normal” case in the data. Misprediction of outlier cases can be viewed as the “cost” of achieving optimum prediction accuracy on the normal cases.

Evaluating analytic models

[Chapter 15](#) covers both of the subjects of model evaluation (how “good” the model is) and model enhancement (how you can improve the model), because these activities are often linked together. One of the linkages is composed by the function of model evaluation that can point to the need to change algorithm hyperparameter values to increase prediction accuracy values. Another linkage appears when preliminary modeling results reveal the need to derive additional interaction variables for the modeling algorithm to use.

Enhancing analytical models

One of the most common techniques for model enhancement described in the chapter is the cross-validation technique that functions to remove effects of data sampling in the production of model training, testing, and validation data sets. Two important other model enhancing techniques, boosting and bagging, are described. The bagging technique can be combined with an ensemble modeling architecture to yield a very powerful modeling system.

A taxonomy of algorithms

Preamble

After the data preparation phase of modeling is completed, the first task in the modeling phase is algorithm selection. The study of machine learning presents an almost overwhelming variety of algorithms. It is tempting to think that either there is a small set of “best,” up-to-date algorithms or there is an optimal algorithm for each use-case. Neither is entirely true. It is true that algorithms have to be selected for their suitability for a class of problems. This fact does allow for the description of a taxonomy of algorithms, the focus of this chapter.

Introduction

Finding the best modeling algorithm for any use case always requires experimentation. This would seem to make achieving mastery daunting. While there is some truth to that, and it can be a career-long endeavor to learn new algorithms, we believe that it is possible to assemble a manageable list of algorithms that everyone should learn first. Such a list would have to have good coverage of the taxonomy in terms of class of problem, would identify algorithms that are fundamental to studying machine learning as a whole, and would be useful in a variety of situations.

Any list prioritizing algorithms will have to address the criterion of covering the most important use cases. The most common classes of problems in machine learning are:

- Predicting a category with supervised learning (classification)
- Predicting a value with supervised learning (numeric prediction)
- Unsupervised segmentation (clustering)
- Time Series Analysis (forecasting)
- Data Reduction (Factor Analysis/PCA)
- Association

The top five modeling algorithms

Weighing those considerations listed above, Table 13.1 shows our recommended list of the top five algorithms to learn first.

Classification and Regression Trees (CART) – actually the combination of two algorithms
Multiple Linear Regression (MLR)

Logistic Regression

k-Means Clustering

Why are these algorithms selected?

These algorithms cover the three most important types of data analytics models:

- Regression
- Classification
- Clustering

The small amount of redundancy in this regard is justifiable. Both Regression Trees and MLR perform numeric prediction, but one comes from the machine learning traditions and one is parametric statistical in nature. Classification Trees and Logistic Regression form a similar pair that both perform classification, the first being a machine learning algorithm and the second

TABLE 13.1 Five algorithms to learn how to use first.

Algorithm	Function	Tool name
Classification Trees	Classifying categories	CART – menu options for choice
Regression Trees	Predicting numbers	
Multiple Linear Regression	Predicting numbers	MLR
Logistic Regression	Classifying categories	LR
k-Means Clustering	Classifying categories	k-Means

Note that Classification and Regression Tree algorithms are combined into one tool (CART) in most implementations.

being parametric statistical in nature. The understanding of these five algorithms will be very helpful in understanding more complex algorithms. CART was developed by Breiman et al. (1984). Several of the most popular algorithms in use today, Random Forest and XGBoost, are also a direct result or influenced by Breiman's work.

Once you have learned these algorithms, in order to get greater coverage of classes of problems, we encourage you to start tackling the following:

- ARIMA (time series)
- A priori (association rules)
- Factor Analysis/PCA (data reduction)
- Random Forest
- XGBoost
- Support Vector Machines (both classification and regression forms)
- "Shallow" Artificial Neural Networks (both classification and numeric prediction)

How do these expand your ability to tackle machine-learning projects? ARIMA, A priori, and Factor Analysis/PCA represent three additional classes of problems. While the other three are classification techniques, they are diverse in nature, and very common in contemporary machine learning. Also, Random Forest and XGBoost are ensemble techniques, which is an important category. They are also noteworthy for frequently winning modeling competitions like those on Kaggle.com. Support Vector Machines are noteworthy for both their popularity as well as their ability to handle "short, wide" datasets combining a relatively small number of instances and a very large number of input variables.

While "shallow" neural networks, with just one or two hidden layers, might be considered a bit out of fashion, they are sufficiently complex to tackle a number of classification and numeric prediction problems. The small number of hidden layers contrasts them with deep learning. More about this difference is discussed in Chapter 17.

More advanced algorithms

After you have a substantial comfort level in these dozen algorithms, additional algorithms can be explored on an ad hoc basis. Most modelers will leverage some kind of automation during the modeling phase. It has become very easy to explore a "tournament" of algorithms by looping through various algorithms and algorithmic settings. When you do this, you may find that an algorithm that you are less familiar with is performing well on your data. That can be an invitation to learn more algorithms. This scenario will be most common with

classification problems as there are dozens of popular classification algorithms in common use. Why so many?

Tom Khabaza provides some insight to the answer to this question in his online article, "Nine Laws of Data Mining" (http://khabaza.codimension.net/index_files/9laws.htm). Of particular interest is his fourth law, "There is No Free Lunch for the Data Miner," which provides some insight to the answer:

Kabaza lists five factors which contribute to the necessity for experimentation in finding data mining solutions:

1. *If the problem space were well-understood, the data mining process would not be needed – data mining is the process of searching for as yet unknown connections.*
2. *For a given application, there is not only one problem space; different models may be used to solve different parts of the problem, and the way in which the problem is decomposed is itself often the result of data mining and not known before the process begins.*
3. *The data miner manipulates, or "shapes," the problem space by data preparation, so that the grounds for evaluating a model are constantly shifting.*
4. *There is no technical measure of value for a predictive model (see eighth law).*
5. *The business objective itself undergoes revision and development during the data mining process, so that the appropriate data mining goals may change completely.*

These forces have been at work since the birth of data mining. Research during the past 40 years has generated many kinds and variants of data mining algorithms that are suited to particular areas in the solution landscape.

Fig. 13.1 illustrates where specific data mining algorithms fit into the solution landscape of various business analytical problem areas: operations research (OR), forecasting, data mining, statistics, and business intelligence (BI).

Fig. 13.1 came from studies by Dustin Hux and John Elder (both of Elder Research, Inc.) on algorithms used in journal articles in different domains. From Fig. 13.1, you can see which field uses what technique and also what techniques are suited to overlaps between areas. For example, visualization and cross-tabulations are used in business intelligence, data mining, and statistics.

Data miners use many analysis techniques from statistics but often ignore some techniques like Factor Analysis (not always wisely). In addition, data mining includes a lot of techniques that are not considered typical in the world of statistics (such as radial basis function networks and genetic algorithms). The discipline of OR not only uses clustering, graph theory, neural networks, and time series but also depends very heavily on simulation and optimization. Forecasting with time-series data sets overlaps data mining, statistics, and OR, and adds a few algorithms like Fourier transforms and wavelets.

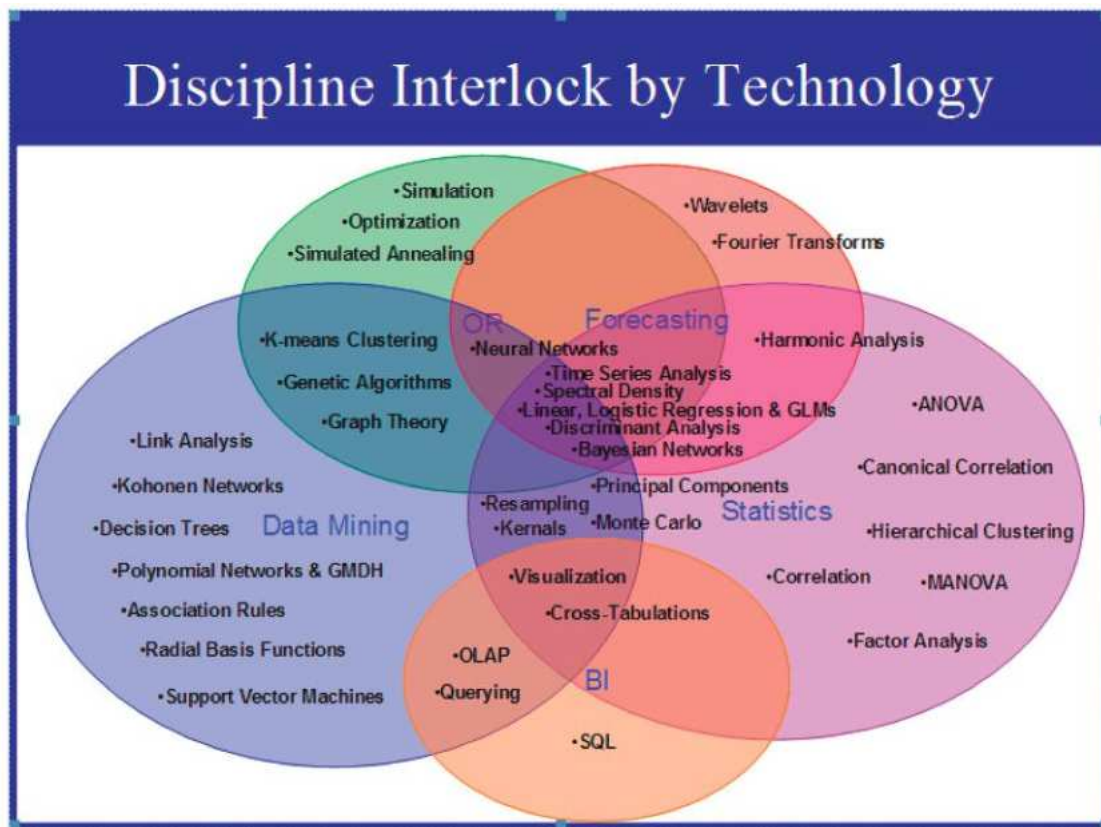


FIG. 13.1 The relationship among specific algorithms and business analytical problem areas (and the overlap among them).

In addition to the overlap of algorithms in different areas, some of them are known by different names. For example, Principal Components Analysis (PCA) is known in electrical engineering as the Karhounen-Loeve transform and is known in statistics also as the eigenvalue-eigenvector decomposition.

In the early college years, students took courses in many different disciplines, and it appears that techniques were developed independently. One of the important byproducts of higher education (especially graduate school) is that we begin to see the interconnections between these ideas in different disciplines. The Ph.D. degree is short for Doctor of Philosophy. Doctoral degrees are handed out in many very technical disciplines, and it might seem strange that “philosophy” is still in the name. What does philosophy have to do with Recombinant DNA Genetics, for example? The answer is – *everything*. One of the jokes often heard in graduate schools is “You learn more and more about less and less until you know everything about nothing.” Granted, a very highly constrained subject matter discipline is the end point (not quite “nothing” even in scope), and through the process of getting there, you can see the connections with a great many other disciplines. And this connected view of a constrained subject area (e.g., genetics) provides the

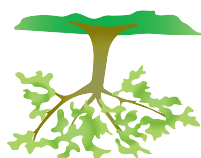
necessary philosophical framework for the study of your specific area. You are not educated properly in a discipline until you can view it in the context of its relationship with many other disciplines. Johansson (2004) maintains that a major source of innovation comes from the union of previously isolated disciplines. It is so with the study of analytical algorithms.

This book will take you far along that path (books like the one by Hastie et al. (2017) go deeper and broader), but this introduction will provide enough background to help you navigate through the plethora of data mining and statistical analysis algorithms available in most data mining tool packages and select an appropriate algorithm for your modeling purposes.

Listed here are some of the more popular algorithms to consider for more advanced study which provide helpful supplements to the top 12 that we’ve already listed. These are both commonplace in software implementations and would also be recommended for eventual professional development for the data scientists.

Alternatives for classification:

- C4.5 & C5
- CHAID
- Discriminant analysis



Decision Trees

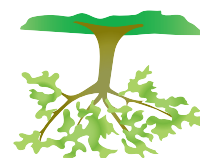


FIG. 13.2 The branching pattern of decision trees is more like the rooting pattern of a forest tree underground than the branching pattern above ground.

- Multinomial Logistic Regression
- k-Nearest Neighbors
- Gradient Boosted Trees
- Naive Bayes

Alternatives for numeric prediction:

- Ridge regression
- LASSO
- Elastic Net
- GAMs

Alternatives for clustering:

- DBSCAN
- BIRCH
- Self-Organizing Maps
- EM Clustering

A brief introduction to the top five algorithms

Although experts could reasonably disagree on which algorithms should be in the “top five,” the ones that we’ve proposed will provide a good grounding in the fundamentals and cover the most common types of problems. We will now review them briefly.

Classification and Regression Trees (CART)

Notice in [Table 13.1](#) that the CART tool is actually a combination of algorithms for performing classification modeling tasks and regression modeling tasks. A strong case can be made for learning the CART first, because it produces single decision trees from which the outcome (predicted value) is obvious. In other words, CART is very “transparent.” This feature is important during preliminary modeling, even when a more complex model is chosen for deployment, because the “short list” of predictor variable is easier to determine with simple decision trees. Finally, CART is the foundation of the development of the Random Forest and XGBoost algorithms. We can begin the review of CART with a discussion of simple decision trees.

The trees in [Fig. 13.2](#) might appear to you as rather odd. You might wonder if they are oriented mistakenly upside-down; this is not a mistake – that is the orientation used to display successive branches in most

decision trees. This orientation is more analogous to the rooting pattern of trees below ground rather than the branching pattern above ground.

What is a decision tree?

CART (or C&RT) methodology was introduced in 1984 by UC Berkley and Stanford researchers, Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone ([Brieman et al. 1984](#)). CART processing is structured as a sequence of simple questions. The answers to these questions determine what next question, if any, is posed. The result is a network of questions that forms a treelike structure. The “ends” of the tree are terminal “leaf” nodes, beyond which there are no more questions.

Key elements defining a decision tree algorithm are:

- Rules at a “node” for splitting the data according to its value on one variable
 - These splits are made at a “cut-point,” determined by heuristics
- A “stopping” rule for deciding when a subtree is complete
- Assigning each terminal “leaf” node to a class outcome (prediction)

Trees recursively partition the data, creating more homogenous groups at each step. The resulting “rules” are the paths it takes to get from the “root” node to each “leaf” node.

Consider the tree shown in [Fig. 13.3](#), created to classify the Iris data set (a standard data set used in data mining discussions and benchmarks). Only two variables are needed to classify iris type: petal length and petal width.

When you consider two of the predictor variables, petal length and width, you will see how the CART algorithm processes the first question at node 1. [Fig. 13.4](#) shows a categorized scatterplot of the results of this simple decision tree model.

You can see that the algorithm found a cut-point that perfectly distinguished the species *Setosa* from the other two species (*Versicolor* and *Virginica*). The rule is that when petal length is less than 2.45 mm, then *Setosa* is characterized. Subsequent questions will distinguish *Versicolor* from *Virginica*.

The second split is on petal width. [Fig. 13.5](#) shows the tree with the decision rules for each node.

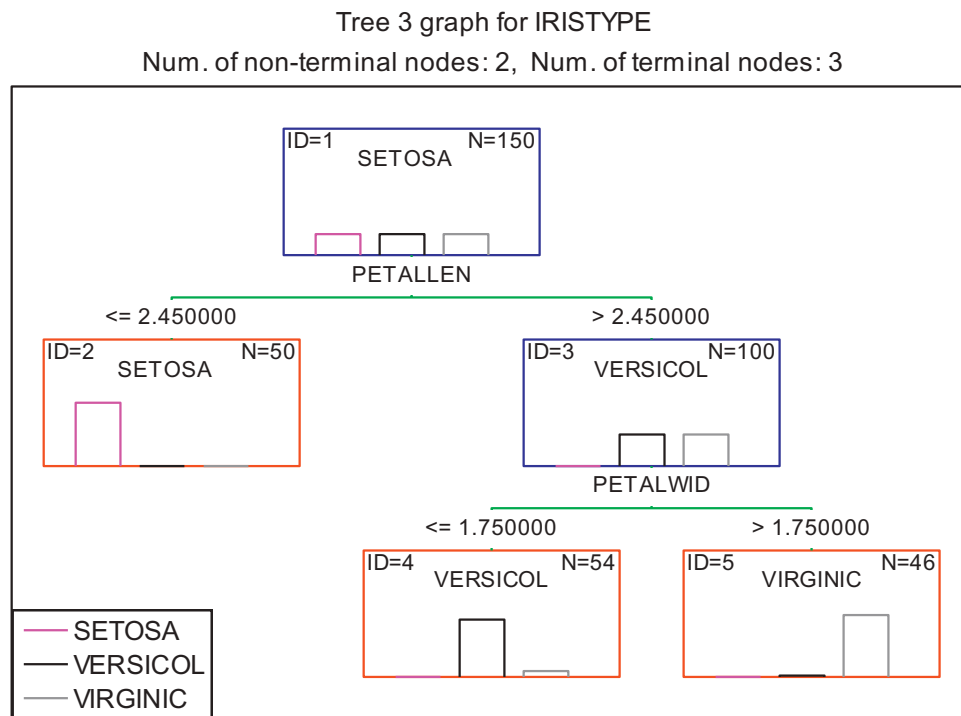


FIG. 13.3 A simple decision tree created with the Iris data set. (Available from the UC-Irvine Machine Learning Repository – <http://archive.ics.uci.edu/ml/>.)

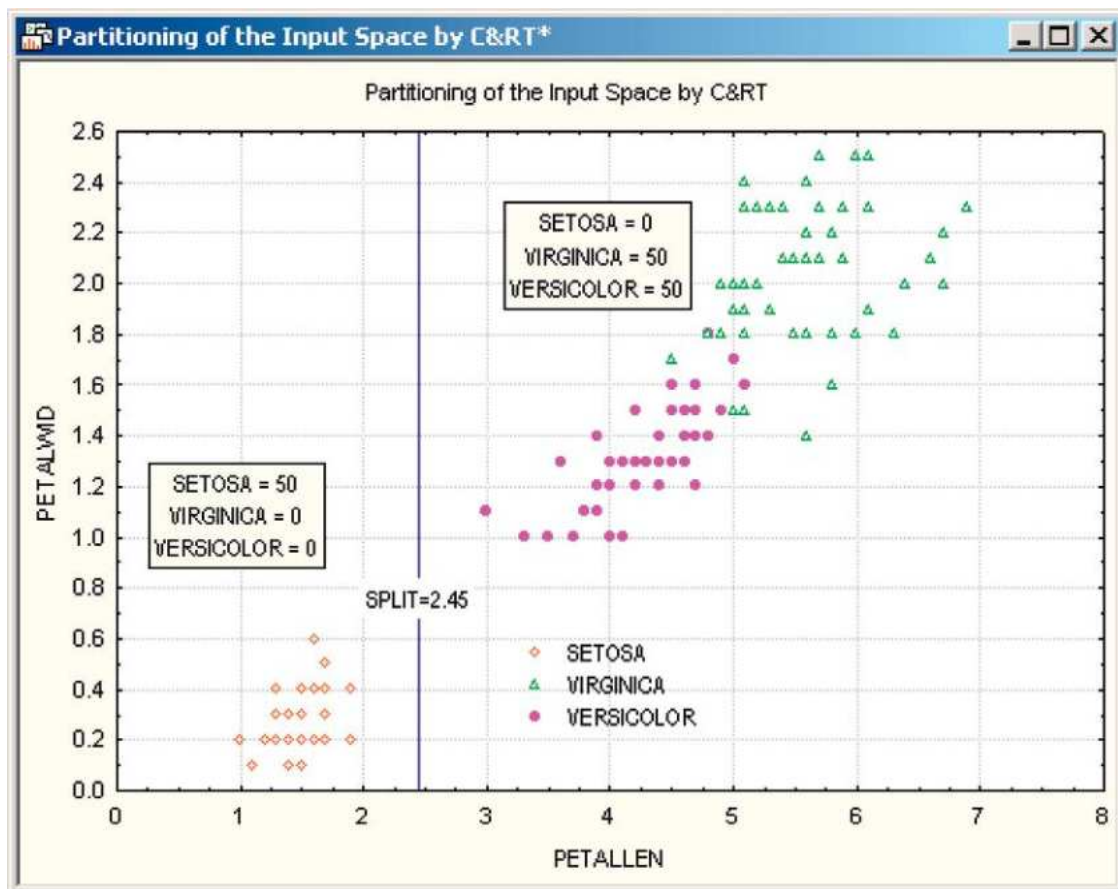


FIG. 13.4 Scatterplot of results of the simple decision tree for the Iris data set.

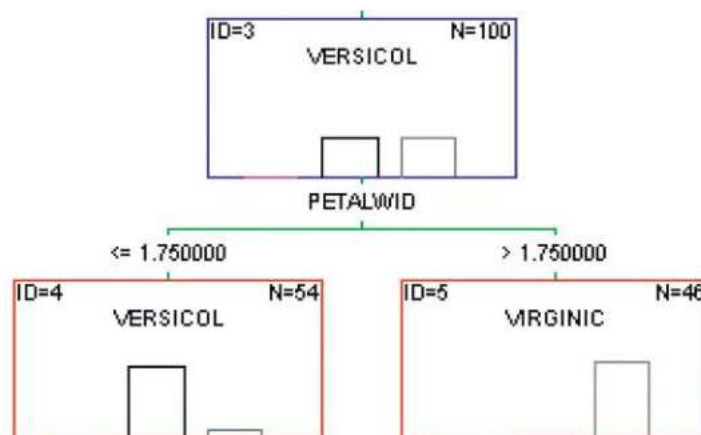


FIG. 13.5 The second split in building the decision tree for the Iris data set.

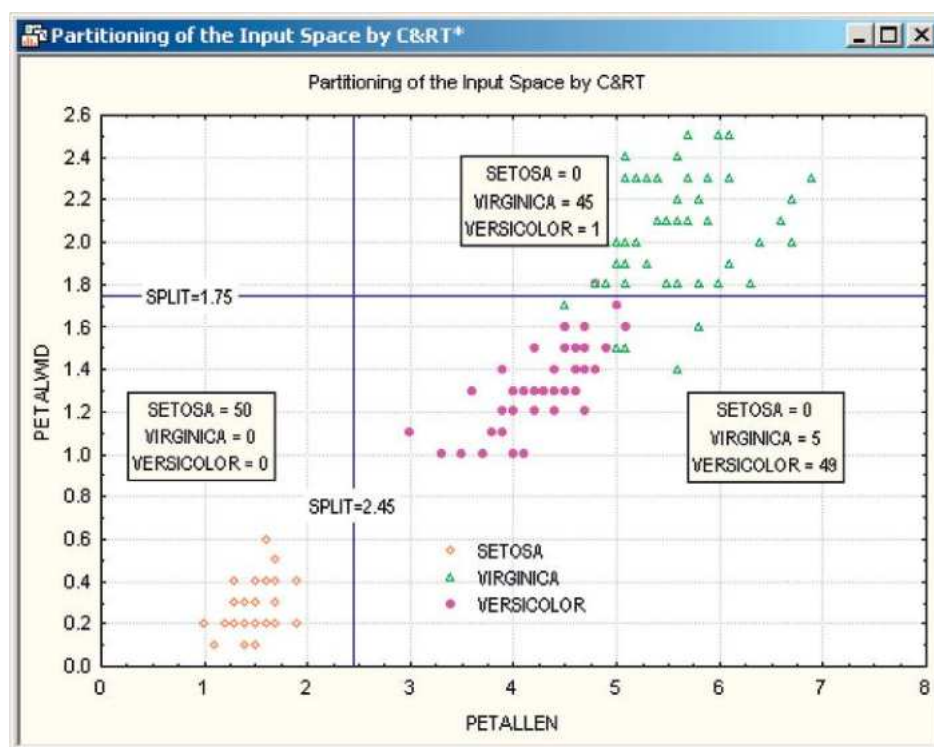


FIG. 13.6 The categorized scatterplot after two splits of the Iris data set.

In Fig. 13.5, we can see that *Versicolor* and *Virginica* can be distinguished adequately (but not perfectly) by asking the second question: “Is the petal width greater than or equal to or less than 1.75 mm?”

The final categorized scatterplot is shown in Fig. 13.6.

The final tree is shown in Fig. 13.7.

Recursive partitioning

Once a best split is found, CART repeats the search process for each node below (“child” nodes), until

further splitting is either stopped by a criterion or splitting is impossible.

Common stopping conditions include:

- Minimum number of cases.
- A certain fraction of the total number of cases is in the node.
- A maximum number of levels of splitting has been achieved.
- The maximum number of nodes has been reached.

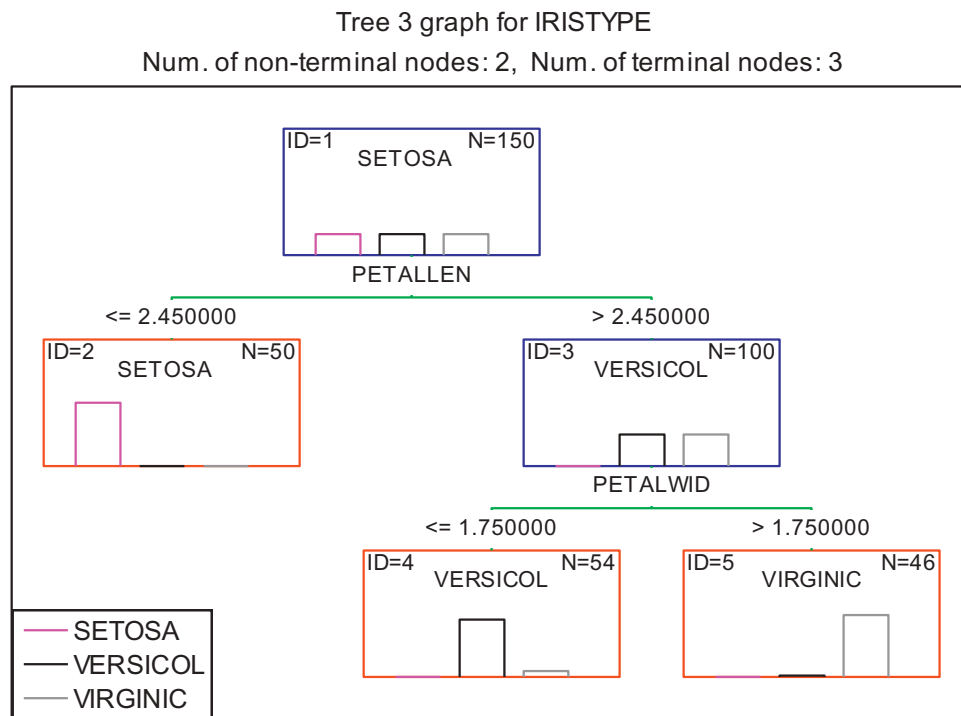


FIG. 13.7 The final tree to classify three Iris species.

Conditions under which further splitting is impossible include when:

- Only one case is left in a node.
- All cases are duplicates of each other.
- The node is pure (all target values agree).

Pruning trees



Rather than focusing on when to stop pruning, CART trees are grown larger than they need to be and then pruned back to find the best tree. CART determines the best tree by using the testing data set or by the process of V -fold cross-validation. The testing validation is performed by scoring the tree with the data set not used for training the model. Cross-validation is a form of

resampling, which draws a number of samples from the entire distribution and trains models on all samples. The V -fold cross-validation is performed by:

1. Partitioning the entire data set into a number (V) of parts (folds).
2. Training V models on different combinations of $V-1$ folds, with the error estimated each time using the V th fold.
3. Using the mean (and sigma) of the V error measurements to estimate tree accuracy on new data.
4. Choosing the design parameters (e.g., complexity penalty) that minimize the error in step 3.
5. Refitting the tree, using all the data and using the parameters of step 4.

Fig. 13.8 shows a three-fold cross-validation operation.

The cross-validation process provides a number of independent estimates of the error associated with the algorithm itself rather than due to the randomness in the data. A model created with a CART algorithm (or any other algorithm, for that matter) should not be accepted until the prediction error is partitioned in this manner.

General comments about CART for statisticians

1. CART is nonparametric and does not assume that input data follow a specific distribution.
2. The final modeling variables are not selected beforehand but are chosen automatically by the algorithm.

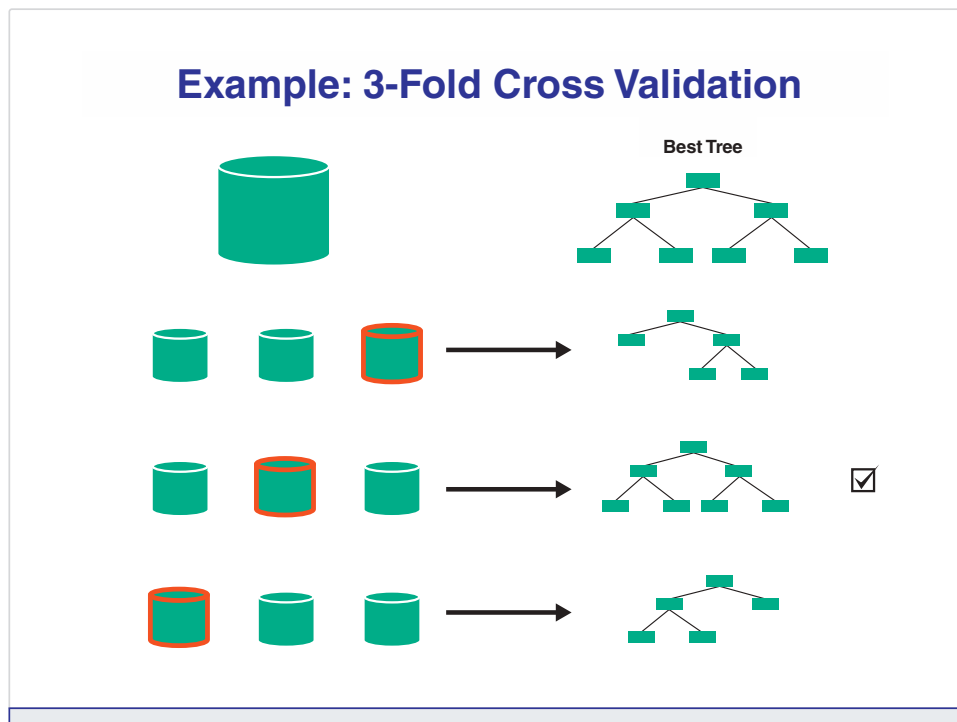


FIG. 13.8 An overview of how a three-fold cross-validation design works.

3. There is no need to transform data to be consistent with a given mathematical function. Monotonic transformations will have no effect.
4. Very complex interaction patterns can be analyzed.
5. CART is not significantly affected by outliers in the input space.
6. CART is affected, but only locally, by outliers in the output variable.
7. CART can accept any combination of categorical and continuous variables.
8. CART can adjust for samples stratified on a categorical dependent variable.
9. CART can process cases with missing values; the cases are not deleted.

Advantages of CART over other decision tree algorithms

1. You can relax the stopping rules to “overgrow” decision trees, and then prune back the tree to the optimal size. This approach minimizes the probability that important structure in the data set will be overlooked by stopping too soon.
2. CART incorporates both testing with a test data set and cross-validation to assess the goodness of fit more accurately.
3. CART can use the same variables more than once in different parts of the tree. This capability can uncover complex interdependencies between sets of variables.

4. CART can be used in conjunction with other prediction methods to select the input set of variables to be submitted to the final modeling algorithm. For example, CART can be incorporated into hybrid models in which CART feeds inputs to a neural network model (which itself cannot select variables).

Uses of CART

1. *Data preparation.* Classical statistical models require that the analyst has a clear understanding of the nature of the function inherent in the data to be modeled. CART requires very little inputs for the beginning data miner.
2. *Variable selection.* Cart can be used to create the short list of predictor variables to submit to the final modeling algorithm. There is no guarantee that the variables most useful for a tree will also prove most useful for a neural network or other function, but in practice, this is a useful technique.
3. The use of predictors multiple times in the tree helps to detect complex interactions in the data.
4. CART can handle missing values by identifying surrogate (alternate) splitting rules. During training, after the best split is found for a node, new splits using *other* variables are scored according to their similarity in distributing the data to the left and right child nodes. The best five or so are then stored as backup or surrogate questions to ask should the main variable not be available.

Multiple Linear Regression

The next algorithm among the top five is MLR. Whenever surveys of data scientists are conducted, regression is always ranked near the top for algorithms used on a regular basis on real-world projects. This finding often generates a combination of surprise and a round of jesting that a 19th-century algorithm is “still used” or even favored over newer techniques. This surprise comes from a misunderstanding of the modeling process (which will be discussed in [Chapter 14](#)) and also from a misunderstanding of how regression is actually used. Linear Regression is used as a component of the Regression Tree processing in CART. Also, there is frequent confusion about whether MLR can handle nonlinear relationships. As we will see, it can be used to explore and model curvilinearity if polynomial regression is understood. And count data can be analyzed successfully with the Poisson form of regression. Therefore even more recently developed algorithms (like Random Forests) cannot be understood properly until Linear Regression is understood thoroughly.

MLR was first proposed by Sir Francis Galton (1822–1911). Galton coined the term *regression* to describe the observation that the majority of very tall fathers had sons who were shorter, and most very short fathers had sons taller than they. The trend of this progression in height was toward the average (or mean) height. This phenomenon was termed *regression to the mean*. His analysis of this effect became known simply as regression.

The major objectives of Linear Regression are to:

- Determine if a relationship exists between one variable and another (or a set of others)
- Describe the nature of this relationship, if it exists
- Quantify the accuracy of this relationship
- Evaluate the relative contributions of each variable, if multiple variables are used

MLR makes all three assumptions described above.

Some practitioners appeal to the Central Limit Theorem (CLT) to correct for nonnormality by taking multiple samples and working on the means rather than the original data. The approach is to generate a new data set by selecting n -samples and to find the mean values for each variable in all n -samples. The CLT states that the distribution of this data distribution approaches normality as the number (n) of samples approaches infinity. For a multivariate data set, you must perform this sampling operation on all variables you want to use in the analysis. The general rule of thumb posits that at least 100 samples ($n = 100$) are necessary to ensure that the distribution will be sufficiently normal for use in predictive analyses. Engineering control charts rely on that property by making many samples of a process flow and creating this group of means. The control charts are

based on the group of means, not the underlying data from which they were calculated.

The basic process of MLR of a data set is to estimate values (named *coefficients*) for each candidate predictor variable (X) to represent the effect that variable has on the response variable (Y). These effects are assumed to be linear and additive (that is, you add them up across all variables to get the total effect). The general form of the regression equation generated by the analysis is:

$$Y = a + b_1X_1 + b_2X_2 + b_3X_3 + \dots + b_nX_n \quad (13.1)$$

In [Eq. 13.1](#), the variable a is the Y value where $X = 0$ (the *intercept*), b_1 is the coefficient for X factor #1 (X_1), b_2 is the coefficient for X factor #2 (X_2), and so forth, through the last X variable (X_n). The coefficients reflect the effects of two sources of relationship of the X variable to the dependent variable (Y), the relative effect of X on Y , and the effect of differences in scale between X and Y . If variable Y and all of the X variables have the same scale, then the coefficients reflect the relative predictability of each X variable. But if the scales are different, the coefficients may not reflect much of the relative predictabilities. Usually, the scales of variables are quite different. One technique for overcoming the effects of scale is *standardization*. This is a process of transforming all variables to a common scale. One common standardization method is to subtract the mean from a value and divide it by the standard deviation (named the *z-score*), which creates a scale in terms of standard deviation units. Another method used to create a common scale from 0 to 1.0 is to normalize the variable over its range as $\text{new\_value} = (\text{old\_value} - \text{minimum\_value}) / (\text{maximum\_value} - \text{minimum\_value})$.

Logistic Regression

This form of a regression algorithm will be discussed in [Chapter 14](#) – Modeling.

Clustering algorithms

The purpose of clustering techniques is to detect similar subgroups among a large collection of cases and to assign those observations to the clusters as illustrated in [Fig. 13.9](#). The clusters are assigned a sequential number to identify them in results reports. A good clustering algorithm will find the number of clusters as well as the members of each. Cases within a group should be much more similar to each other than to cases in other clusters.

A typical example application of cluster analysis is a marketing research study, where a number of variables related to consumer behavior are measured for a large sample of respondents. The purpose of the study

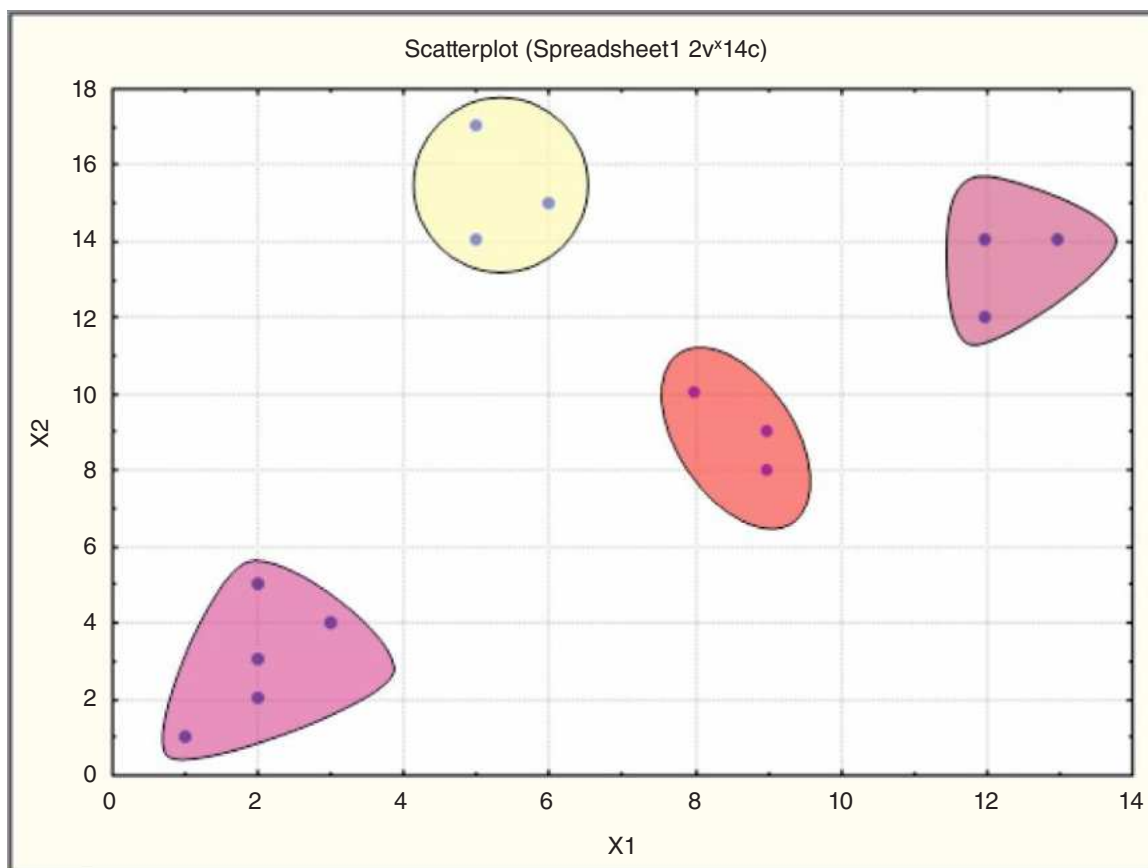


FIG. 13.9 Data clusters in a clustering problem

is to detect “market segments,” i.e., groups of respondents that are somehow more similar to each other than they are to respondents in other groups (clusters). Just as important as identifying such clusters is the need to determine how those clusters are different. The most popular clustering algorithm is k-Means Clustering.

k-Means Clustering

The classic k-Means algorithm was introduced by Hartigan (1975; see also Hartigan and Wong, 1978). Its basic operation is simple: Given a fixed number (k) of clusters, assign observations to those clusters so that the means across clusters (for all variables) are as different from each other as possible. The difference between observations is measured in terms of one of several distance measures, which commonly include Euclidean, Squared Euclidean, City-Block, and Chebychev.

For categorical variables, all distances are binary (0 or 1); it is assigned as 0 when the category of an observation is the same as the one with the highest frequency in a cluster; otherwise it is assigned a value of 1. So for categorical variables (with the exception of the Chebychev distance), the different distance measures will yield identical results.

V-fold cross-validation as applied to clustering

This concept was introduced in the discussion above on CART. As mentioned earlier, cluster analysis is an unsupervised learning technique, and we cannot observe the (real) number of clusters in the data. However, it is reasonable to replace the usual notion of “accuracy” with that of “distance” (applicable to supervised learning). In general, we can apply the V-fold cross-validation method to a range of numbers of clusters and observe the resulting average distance of the observations (in the cross-validation or testing samples) from their cluster centers (for k-Means Clustering).

(Note: The above discussion on k-Means is based on Witten and Frank, 2005).

Postscript

Armed with this overview of prediction algorithms, we can move to a discussion about how to use them in Chapter 14 in the modeling process. The primary job of the analytical modeler is the same as envisioned by Michaelangelo for sculpture: to uncover the “solution” from within a background of superfluous “material.” This notion was unpacked in Chapter 11. We will build upon this notion in Chapter 14 on Modeling.

Keywords and phrases

Algorithms, CART, MLR, Classification, Regression, Clustering, Cross-validation, Samples, Decision trees, Classification trees, Logistic Regression, Random Forest, k-means Clustering.

References

- Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Monterey, CA: Cole Publishing. 358p.
- Hastie, T., Tibshirani, R., & Friedman, J. (2017). *The Elements of Statistical Learning: Data*.
- Hartigan, J. A. (1975). *Clustering algorithms*. New York: Wiley.
- Hartigan, J. A., & Wong, M. A. (1978). Algorithm 136. A *k*-means clustering algorithm. *Applied Statistics*, 28, 100.
- Johansson, F. (2004). *The medici effect: Breakthrough insights at the intersection of ideas, concepts, and cultures*. Cambridge, MA: Harvard Business Review Press.
- Witten, I. H., & Frank, E. (2005). *Data mining: practical, machine learning tools and techniques*. New York: Morgan Kaufmann.

Modeling methods

Prelude

Now that your data sets are properly prepared for modeling, we must dive into the deep subject of different types of models and different approaches in building them. This chapter exposes the fact that it matters how you *think* about your data and the models you want to build with them. Different people naturally think in different ways. The challenge before the data analytics modeler is to harvest the best aspects of multiple thinking approaches to modeling to produce models that reflect all important aspects of the reality represented by data in a specific data domain. The character of Joe Friday in the old *Dragnet* TV series was fond of saying, “Just give me the facts, ma’am, just the facts.” But often, those facts regard only the details of who, what, when, and where, but not the “why” it happened. A forest may decline in volume of wood in the trees not because the trees stopped growing (controlled by details of chemistry of the soil and water availability), but because a massive wildfire (a big-picture event) burned up many of them. The analytics modeler must capture both detailed information data elements (like phone call durations extracted from detailed switch data sources) and big-picture information data elements (like a given customer is a women-owned business, gathered from demographic data sources) and install them into predictor variables to submit to the modeling algorithms. Analytical modeling is not an exact science; it is an estimation process that produces solutions that can be deemed as good enough to satisfy their intended purposes.

Scope of modeling operations

Analytical modeling operations depend on the completion of most of the data preparation operations. Preliminary modeling, however, is a part of data preparation activities, as discussed in [Chapter 3](#). Processes of preliminary modeling may suggest additional data preparation operations, such as derivation of additional

variables, may be indicated by the analysis of preliminary model results. After final data preparation operations are completed, formal modeling operations can proceed. Before describing basic modeling operations, a review of a number of modeling issues is necessary.

General modeling issues

- a. *We tend to think linearly in a nonlinear world.* We tend to think that cause–effect relationships follow a straight line.
We think in terms of trends of change as straight lines inclined up (increasing trend) or down (decreasing trend). The problem is that many real-world trends are not linear; rather, they are highly nonlinear.
- b. *We tend to think that factors of an effect act independently to produce an outcome.* We understand that outcomes (effects) are caused by factors, and we think of those effects operating separately apart from their joint action on the outcome. Real-world effects, however, are caused almost always by the joint action of a number of factors acting together. We must account for the effects of these variable interactions in the structure of our models. Statistical analysis algorithms don’t do this well (if at all). Machine learning algorithms do this operation very effectively.
- c. *We tend to ignore the operation of emergent effects.*

Emergent effects (or properties) manifest themselves only when a complex system is complete, with its components working together, not individually. These effects compose a “whole” that is greater than the sum of its parts (*à la* Plato). A good example is an ant colony. Individual ants seem to move around with little purpose and accomplish very little. When many ants function together, however, they can accomplish much (e.g., build mounds and carry large amounts of food into the mounds). These results of cooperative action emerge only when the ant colony is complete and

functioning as a whole. Many organizations subject to analytical modeling operations have emergent properties that function in ways analogous to an ant colony. We must keep these emergent properties in mind when designing analytical modeling solutions.

- d. We tend to think either from the bottom up (detail-oriented people) or from the top down (big-picture people).

Chapter 1 introduced the most well-known examples of this bifurcation in our thinking patterns, Plato and Aristotle of ancient Greece. Plato's philosophy was based on his belief that a perfect world exists beyond what we can perceive with our senses (Fig. 14.1). This concept is known as the Theory of Forms, which states that things we can sense are only imperfect copies of the things that exist in the perfect world.

Aristotle's philosophy was fixed on those things we can sense in the world around us – what the eye could see and the hand could touch. He believed that true knowledge comes from our senses, rather than from abstract reasoning.

Poets like Homer can portray the world in a one-dimensional manner of their prose. Painters like Rembrandt (who painted Fig. 14.2) can show the world in an expanded format. In selecting Aristotle as the subject, Rembrandt may be presenting the role of vision (of Aristotle) and his "chain" of evidence (wrapped around his middle) as superior

to the blind perception of Homer, the poet. We might look at the progression of the four kinds of analytical models shown in Fig. 14.3 in the same way.

- e. We tend to start with models of the framework of reality, and "hang" our process models on them.

Early analytical modelers in science assumed that Sir Ronald Fisher was right to insist on using the Parametric Model as a framework of reality, and build models of reality within it. The problem is that most real-world problems violate its major assumptions big-time. Alternatively, machine learning modelers start with the data and let the structure of the model be built by the algorithm, using the predictor variables as input. We will explore that difference at the end of this chapter.

Types of analytical models

Many data scientists recognize four basic kinds of analytical models, as shown in Fig. 14.3.

Fig. 14.3 shows overall features of the four kinds of analytics models. The sequence of descriptive – diagnostic – predictive – prescriptive models is associated with increases in practical value (the Y axis), and information gain, complexity, and difficulty on the X axis. The order of models along the diagonal line of Fig. 14.3 implies the order in which each model is built. Descriptive models usually are built first, and prescriptive models are usually built last.



FIG. 14.1 Plato pointing up to signify the importance of ideas.



FIG. 14.2 Rembrandt's Aristotle before the bust of Homer.

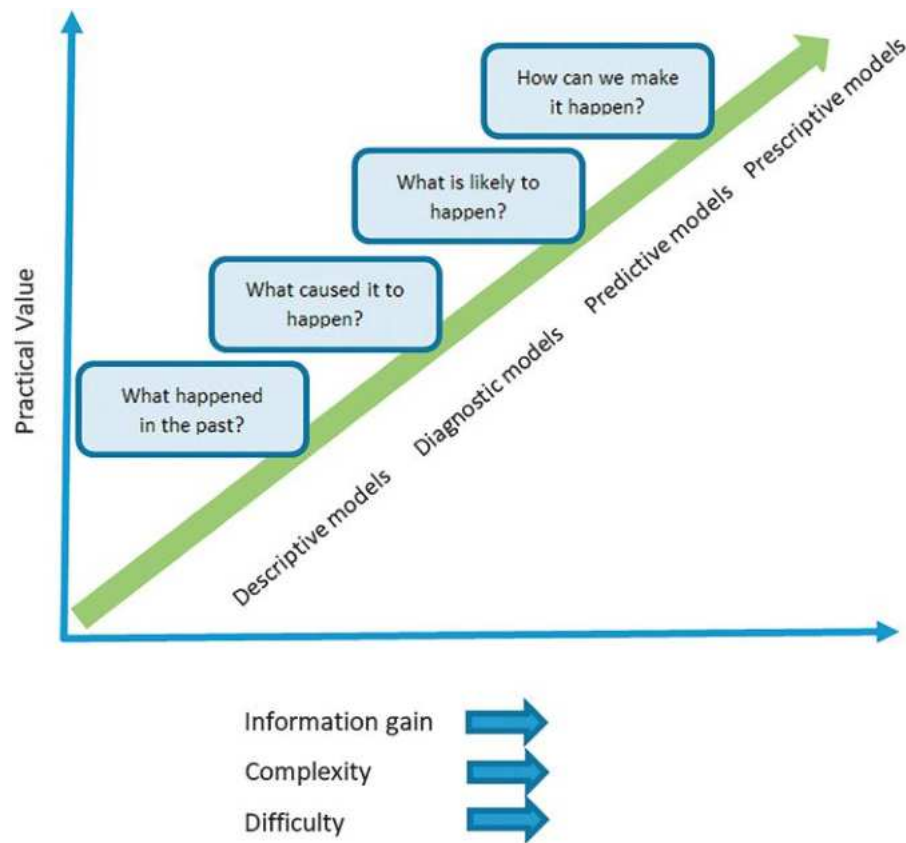


FIG. 14.3 Four kinds of analytical models (after Gartner, 2012).

Descriptive models

Descriptive analytics specifies “what happened in the past.” It focuses on answering questions like:

- How many specific items did we sell last month?
 - What was our net profit last month?
 - How many people left the company last month (those who “churned”)?
- (Note: The term “churn” was coined in the early 1990s to describe the movement of customers from one of the “Baby Bell” phone companies like BellSouth to Bell Atlantic, following the best calling feature promotion offered by Bell Atlantic for the month, such as Personal Secretary. Several months later, the same customers might return to BellSouth in response to a new promotion. This cyclical customer movement reminded the telecommunications analysts of flakes of butter moving around in a butter churn, and they called them “churn.” The term stuck, and subsequently, it was cross applied to many other customer-facing industries).
- Most data integration and reporting tools (e.g., Informatica, Business Objects, and Salesforce.com) have a number of descriptive analytics tools to help integrate and report aspects of the final data structures.

Diagnostic models

Diagnostic analytics attempt to determine why an event happened that was described (possibly) by a descriptive model. Many data science practitioners are eager to move directly from descriptive modeling output (e.g., from sales reporting systems) to predictive models before determining why the described event happened. For example, a system like Salesforce.com might output a descriptive model showing a significant decline in sales by some agents in August without determining why it happened. The reason might be that August is a favorite month for vacations just prior to the beginning of school for the kids in the fall.

Diagnostic models can answer questions such as:

- Why did overall sales decline so precipitously in August?
- Why is the rate of customer churn increasing?
- Why do sales of Beer and Diapers product categories spike together on Mondays?

(Note: This is a classic example of how results from an analytical model (a market-basket model) can help supermarket shelf managers increase their sales. It was noticed by one manager that the results of a market-basket analysis showed that market baskets contained a lot of beer

and diapers on Monday afternoons. After further investigation, it was learned that the customers of those market baskets were almost all young fathers. Dads “tanked up” on beer for them and diapers for the baby they agreed to watch during Monday Night Football games, while the wives stepped out to meet with their friends. This insight led to the design of end-caps on aisles to provide diapers nearby the beer coolers, and to provide salsa nearby the chips, and many other item combinations. Sales of these item combinations soared.)

Predictive models

Predictive models determine what is likely to happen in the future based on what happened in the past. The prediction of future events based on past events assumes that the probability of what happened in the past (number of times an event occurred/total number of competing events). This assumption holds true in many cases, but sometimes it fails miserably. This assumption is violated when new influences arise that cause future events to happen (like a stock market crash). The effect of significant new influences may swamp out old influences, and predictions based on them may fail big-time. In cases where this assumption is true, many predictive models are based on historical data that provide a time course of events like sales or telephone usage. Common applications include:

- Predicting industrial equipment failures
- Calculating credit risk
- Identifying fraud
- Predicting customer behavior such as churn
- Predicting student performance

Prescriptive models

Prescriptive models provide guidance to make a desired future course of action happen. These models are often based on information gained during descriptive modeling and predictive modeling. For example, a descriptive model might show that there is a 2% per year increase in the rate of colon cancer incidence in people under age 50, and a predictive analytics model might indicate that a given patient has a high risk for having precancerous colon polyps (see [Chapter 23](#) for an example of this application). A prescriptive model could provide the physician with information on which treatment plan for colon cancer is most effective.

From this example, you can see that prescriptive analytics requires strong competencies in descriptive, diagnostic, and predictive analytics. Clinical healthcare industries are highly specialized and use-cases are usually well defined. Prescriptive analytics help to address use-cases such as:

- Retail price elasticity models that guide product pricing based on anticipated customer demand and external factors (like the weather)

- Selecting the most effective treatment of a medical condition
- Determining which salespeople need further training in customer relationship management based on analyses of customer satisfaction surveys

Two approaches to building analytical models

Over the history of the development of analytical models to predict future outcomes, two approaches have been followed: statistical modeling and machine learning modeling.

Statistical modeling

Statistical modeling makes predictions based on the mathematical properties of specific data distributions within the boundaries of a number of explicit assumptions. A short history of modern statistical analysis and the primary assumptions that are made in this approach are discussed in [Chapter 1](#). This approach starts with a model of probability (the Normal or “Gaussian” distribution that generates a frequency graph that is bell shaped). This approach reduces all input data to two metrics for each predictor variable, the *mean* (average), a measure of central tendency of the data, and a *standard deviation*, a measure of the distribution of the data set around the mean value. All other metrics are calculated from these two values. One of those metrics is the linear correlation coefficient (r), which is a measure of the degree to which a single predictor variable relates to a target variable to be predicted. The correlation coefficient of the combination of all predictor variables (R) is the multiple correlation coefficient. Statisticians have agreed that the relative amount of causality we can associate with the multiple correlation coefficient R can be expressed by R^2 , called the coefficient of determination. The multiple correlation coefficient R is normalized to range between 0.0 and 1.0. The square of this number declines the numerical value. This decline is assumed to represent the random component of the relationship, and the squared number is assumed to represent the proportion of the relationship that represents the causality of the target variable by the combined effects of the predictor variables. This is another assumption, and it may be the most pernicious.

This assumption can be traced to the suggestion of one man, Sewall Wright, who proposed it in his article published in the *Journal of Agricultural Research* ([Wright, 1921](#)). Actually, this assumption builds upon the theory platform provided by the combined effects of the other assumptions made by this approach (Normality, Linearity, Independency, and Additivity). The R^2 value was quickly adopted by statisticians, and it has become the hallmark of causality in analytics ever since.

The problem is, however, that there are many data domains where it doesn't work very well (e.g., business and patient-centric medicine). It became apparent to practitioners in these data domains (and others) that many data distributions were not Normal, nor were the effects of individual predictor variable independent of each other (many variable interactions were discovered). Also, the combined relationship of all predictor variables to target variables (the outcome to be predicted) was almost never linear. In fact, many relationships were highly nonlinear. This problem was accounted for (at least to some degree) by the invention of statistical modeling algorithms based on nonlinear functions like the logistic function in Logistic Regression. Despite these advances in statistical modeling, analysts recognized that some other method of modeling real-world phenomena was needed, which led to the development of machine learning methods that could account for nonlinear relationships and variable interactions.

Machine learning modeling

Footo (2021) described machine learning as a discipline of artificial intelligence (AI) research. In the 1980s, it branched off AI and began to develop on its own. The first machine learning algorithm developed was the artificial neural network (ANN). It was born in an attempt to simulate the processes of the human brain. The brain uses a network of nerves composed of nerve cells and connections (axons) to other nerve cells. The nerve cells appear to collect electrical signals passed to them until a signal threshold is reached, and then they pass ("fire") the signal to the next cell in the network. Scientists tried to simulate this collect-and-fire approach in the processing of numbers in a computer program (an "algorithm") to predict an end result from the composition of input signals (variables). The result was a relatively poor predictor (Rosenblatt, 1958).

During the 1980s, scientists decided there must be internal processing in neurons, which is not expressed in the initial ANN architecture. They decided to add an internal layer of inputs (a "hidden" layer) between the input layer (the set of predictor variable values) and the output layer (the prediction) and optimized the weights assigned to the signals flowing between them (using a process called backpropagation). Much to their delight, they found that the algorithm (named the Multilayer Perceptron – MLP) worked quite well as a predictor of the target variable. The multiple variable nonlinear processing performed in the processing of the hidden layer signals accounted for a large part of the nonlinearity in the input data. The problem for business applications is that the MLP algorithm did not provide any information about why a given prediction was made – it was a "black box." See Chapter 13 for more details on the structure and function of ANNs.

Decision tree algorithms were developed to address the black-box problem with ANNs and to express highly nonlinear relationships and variable interactions. A decision tree expresses a decision-making process (like prediction) as a series of embedded if-then-else statements. Fig. 14.4 shows a very conceptual diagram of an if-then-else logical construct.

The logic expressed by Fig. 14.4 is: if x is greater than 0, then x is a positive number, else if x is greater than 0.1, x is a small negative number, else x is a large negative number. This logic can be used to induce a rule set:

- a. If x is greater than 0, then x is a positive number.
- b. If x is not greater than 0 and x is greater than -0.1, then x is a small negative number.
- c. If x is not greater than 0 and x is less than -0.1, then x is a large negative number.

This ability to generate a logical rule set from the results of a decision tree is crucial in data domains where the important factors of the decision must be known (e.g., business and medical diagnosis). Consequently, various forms of decision tree algorithms have become very popular in these data domains, particularly the use of decision tree ensembles like Random Forest. The need for explainable algorithms led to the passing of the General Data Protection Regulation (GDPR) by the European Union (EU), which went into effect on May 25, 2018 and mandates transparency between cause and effect in analytical processes in the EU.

Two basic types of predictive models

Among the four types of analytical models, this book focuses primarily on two types of predictive analytics problems: (1) Classification-type models and (2) Regression-type models. Moreover, this book is limited to discussions of analytical methods of tabular data, rather than images, time-series data, voice data, text documents, or any other array-based data sets. Also excluded from this book are analyses of fraud and intrusion data.

Tabular data is organized into rows (records) and columns (variables). Values in each of these columns can be composed of numbers or categories. For example, you can describe human age in terms of number of years since birth or in terms of age classes (Baby, Toddler, Child, Preteen, Teenager, Adult). A categorical variable (e.g., Religion, Age, Gender, etc.) contains one or more types of contents called *classes*. A class refers to specific kinds of a given categorical variable (e.g., age classes listed above). Sometimes, classes are called categories, but that equivalence should be avoided. Analysis of categorical data cannot be performed in the same way as numerical data are analyzed. For example, for five people aged 14, 16, 18, 20, and 22, the average

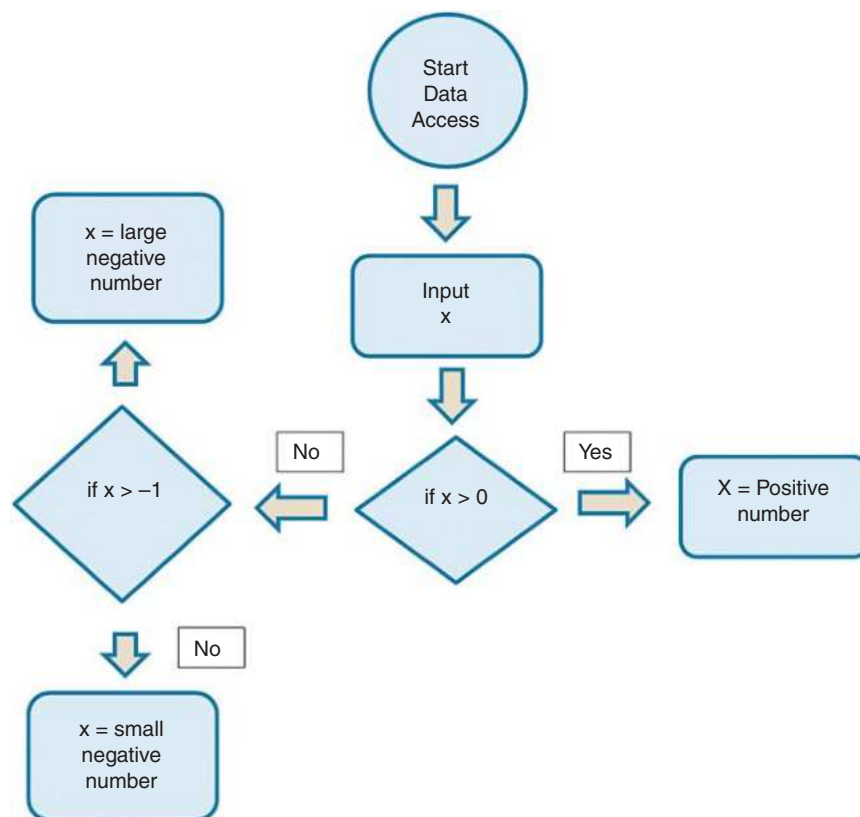


FIG. 14.4 Diagram of a simple if-then-else logical construct.

age is the sum of these ages divided by 5 = 18. If those same five people are in five Profession classes of Insurance agent, Teacher, Doctor, Lawyer, and Dentist, the average of these five classes has no meaning. Therefore different computational methods must be employed to analyze these two problem types. That requirement is the reason for the development of two types of many machine learning modeling algorithms: one type for analyzing Classification problems and another type for analyzing numerical problems. Often, numerical-type problems are called “Regression” type problems, because linear regression was the first algorithm used to analyze them and because linear regressions are components of their processing.

Classification analyzes categorical data (where the class is to be predicted) and Regression analyzes numbers (where a number is to be predicted). Most machine learning algorithms are available in two versions, one for analyzing categorical data and one for analyzing numerical data:

- Classification: to predict a categorical variable
- Regression: to predict a numeric variable

Classification-type models

Classification is the operation of separating various entities into several classes. These classes can be defined by business rules, class boundaries, or some mathematical function. The classification operation may be based on a relationship between a known class assignment and characteristics of the entity to be classified. This type of classification is called *supervised*. If no known examples of a class are available, the classification is *unsupervised*. The most common unsupervised classification approach is *clustering*. The most common applications of clustering technology are in retail product affinity analysis (including market-basket analysis) and fraud detection. In this chapter, we will confine the discussion to supervised classification methods. Unsupervised classification methods will be discussed in [Chapter 17](#) in relation to the detection and modeling of fraud.

There are two general kinds of supervised classification problems in data mining: (1) binary classification – only one target variable, and (2) multiple classifications – more than one target variable. An example of analyses with only one target variable is a

model to identify high-probability responders to direct mail campaigns. An example of analyses with multiple target variables is a diagnostic model that may have several possible outcomes (influenza, strep throat, etc.).

Initial operations in classification models

Before classification can begin, there are some initial tasks you must perform:

1. Determine what kind of classification problem you have. This means that you have to determine how many target classes you have and define them, at least in general terms.
2. Define the boundaries of each class in terms of the input variables.
3. Construct a set of decision rules from class boundaries to define each class.
4. Determine the *prior probability* of each class, based on the frequency of occurrence of a class in the entire data set.
5. If appropriate, you should determine the cost of making the wrong choice in assigning cases to a given class. This task is extremely important in some classification situations. For example, in applications of medical diagnosis, it is far more “costly” to classify a patient as cancer-free when there actually is cancer present than to classify a patient as having cancer when there is none. This information will help you evaluate the classification models. Some algorithms permit you to input misclassification costs, which are used to focus the error minimization operation of the algorithm on the misclassification with the highest cost. For example, SAS Enterprise Miner permits you to define a cost matrix in several of its algorithms.

Major issues of classification

There are a number of issues that you must face before proceeding with the classification project. It is important to consider each of these issues and either resolve the issues before modeling or set some expectations surrounding them.

What is the nature of the data set to be classified?

The purpose of the classification should be specified, and it should be related to the expected interpretation of the results. For example, consider the problem of cancer misclassification discussed above. You probably would not worry much about the false-positive classification rate in a mailing campaign, but failing to diagnose a breast tumor could be fatal.

How accurate does the classification have to be?

If you are in a time crunch, a model that produces 80% prediction accuracy built in 2 days may be good enough to serve the model’s purpose. In business, time is money!

How understandable does the solution have to be?

One of the strengths of a decision tree model is that it produces results that are easy to understand in terms of the predictor variables and target variables. An induced rule set might be even better because it expresses the decision tree splits in terms of if-then-else rules, which are easy for managers to understand. On the other hand, results from a neural net model may be more predictive, but the understandability of the results in terms of the predictor variables is less clear. Often, there is a trade-off between accuracy and understandability of the results. This trade-off may be related to the choice of modeling algorithm. Some algorithms do better for some data sets than others.

What is the relative importance of model accuracy versus generality?

Often, modelers are faced with a trade-off between how accurate the model is and how well it performs on new data (its *generality*). This new data might be additional data from the same sources, but it includes several patterns not represented in the training data sets. Several resampling methods are available to help reduce this problem. Some algorithms perform better on such new data than other algorithms (they have a higher generality). Some modeling problems focus on response patterns that may change significantly over time. Good examples of these dynamic patterns are bank deposit and withdrawal patterns, which may change significantly with the state of the economy. Algorithms chosen in banking applications should have a higher generality (also called *robustness*) than models in many other applications.

Assumptions of classification procedures

Classification in general requires that you accept a number of assumptions. The fidelity of your classes and their predictive ability will depend on how closely your data set fits these assumptions. In [Chapter 4](#), we stressed the importance of describing your data set in terms of the nature of its variables, their possible interactions with the target variable and with each other, and their underlying distributional pattern. In classification, you should try to satisfy these assumptions as much as possible.

Numerical variables operate best

Categorical variables can be used directly in nonparametric classification algorithms, but they should be decomposed into dummy variables if possible (see [Chapter 4](#) for an introduction to dummy variables). Some classification algorithms require that all data are numbers (e.g., Logistic Regression).

No missing values

By default, most data mining algorithms (including those for classification) will eliminate cases with one missing value in a predictor variable. Imputation of missing values is one way to fix this problem (see [Chapter 4](#)). Another way that some classification algorithms use (e.g., C&RT) is to use surrogate variables. A surrogate variable has a similar splitting behavior to the variable with the missing value, and its value in this case can be used to replace a predictor variable with missing values.

Target variable classes are mutually exclusive

Classification targets selected to define categories must be mutually exclusive and categorically exhaustive (MECE). The term “categorically exhaustive” means that the outcome is at least one category class. For example, a data set used to classify shades of red balls cannot contain any blue balls. “Mutually exclusive” means one and only one target can be assigned to each case. If one candidate target variable is “Residential dwelling” another target variable cannot be “Single-family dwelling,” because single-family dwelling is also residential. Both categories may have an equal probability in the classification operation. If MECE is not satisfied, assignment of some cases into categories may be arbitrary (to some extent) and not related exclusively to the predictor variables.

Analyzing imbalanced data sets with machine learning algorithms

Many problems in data mining involve the analysis of rare patterns of occurrence. For example, responses from a sales campaign can be very rare (typically, about 1%). It is relatively easy for a classification algorithm to build a model that is 99% accurate by concentrating only on the common class. There are several ways to balance a data set for use with machine learning algorithms.

1. Oversample the rare class until the number of both classes is the same.
2. Undersample the common class by limiting the number of common class records to equal that of the rare class.

3. Use a weight or prior probability to govern the effects of each class on the classification process. Some algorithms will accept weights or prior probabilities, but most do not.

Models built with many neural net and decision tree algorithms are very sensitive to imbalanced data sets. This imbalance between the rare category (customer response) and the common category (no response) can cause significant bias toward the common category in resulting models.

A neural net learns one case at a time. The error minimization routine (e.g., backpropagation, described in [Chapter 13](#)) adjusts the weights one case at a time. This adjustment process will be dominated by the most frequent class. If the most frequent class is “0” 99% of the time, the learning process will be 99% biased toward recognition of any data pattern as a “0.” Balancing data sets is necessary to balance the bias in the learning process. IBM SPSS Modeler provides a node for balancing rare data sets, and so does Konstanz Information Miner (KNIME). For other tools, you may have to balance a data set manually.

When choosing the balancing method, a caveat is that the under sampling method eliminates some of the common signal patterns. If the data set is not large, it is better to oversample the rare category. That approach retains all the signal patterns of the common category and just duplicates the signal pattern of the rare category. The Synthetic Minority Oversampling Technique (SMOTE) has become very popular for over sampling data sets with rare classes. SMOTE uses a nearest-neighbor method to synthesize new minority class records until the number of records in the minority class equals the majority class. SMOTE can also balance a data set with multiple target classes by synthesizing enough records of each class to equal the number of records in the most common class. Modeling with synthetic data may give some analysts pause, but the successful use of SMOTE for balancing data sets in many different data domains augurs for its acceptance.

Regression-type models

Modern humans have always been fascinated with numbers. The Industrial Revolution was founded on Aristotelian logic, and numerical relationships between movements and actions and the things that cause them. World War II was at the same time the most traumatic conflict in the history of the world and the impetus that drove us into the age of high technology by the need to calculate the trajectories of ordinance. The speed and complexity of these calculations were limited by hand-calculation methods. The single most important influence in modern technology is exponential rate at which we have been able to increase our ability to “crunch” numbers very quickly with computers.

The PC computer is arguably the single most influential technological force in the world today. The ~4GHz PCs of today are almost 1000 times faster than the original 4.77MHz IBM PC in 1981. Central processing units (CPUs) of today can process over 400,000 million instructions per second (MIPS) compared with the 0.3 MIPS of the original PC with an Intel 8080 CPU.

The human brain has about a trillion cells, and each one has many connections with each other. Each one of the brain cells operates with numbers also (in terms of nerve impulse strengths). The “thinking processes” of both humans and computers involve numerical analysis. Thus it is fair to say that numerical analysis is the most basic function in both the carbon-based human world and the silicon-based computer world.

It is not the intention of this book to teach readers how to do numerical analysis or even to understand how modern analytical algorithms formalize it. So far, you have seen very few equations in this book; that motif will be relaxed slightly in this chapter. The equations presented provide convenient display objects to refer to in subsequent discussions. But the presentation of these equations does not constitute a formal definition of the algorithms discussed. Rather than present algorithms in a formal mathematical format, we seek to explain how to *use* these algorithms to build analytical models to solve problems, the most basic of which involve numerical prediction. Even the classification of things is based on numerical operations.

In this section on Regression, we will explore the concepts of *linear* and *nonlinear* relationships that underlie different types of Regression analysis. A linear relationship between a causal variable (or the combined effects of a set of variables) can be plotted on a graph as a straight line, while a nonlinear relationship must be plotted as a curved line. Each of these types of relationships requires different analytical techniques to express them. For linear relationships, we will review the assumptions of the “Parametric Model” of reality (introduced in Chapter 1), and relate them to some classical methods of numerical prediction. We will review the major assumptions of the Parametric Model (introduced in Chapter 1), and relate them to some classical methods of numerical prediction. It is not our purpose to present an exhaustive treatment of numerical prediction algorithms but rather, to discuss common examples found in most data mining tool packages. Several of the algorithms discussed in the Classification section above (Classification&Regression Trees, Random Forests, Boosted Trees, and neural nets) can be configured for use also in numerical prediction.

Linear response analysis in numerical prediction models

The goal in linear analysis is to find a set of predictor variables (X_1 to X_n) in which changes in each predictor variable cause a change in the response variable (Y). This

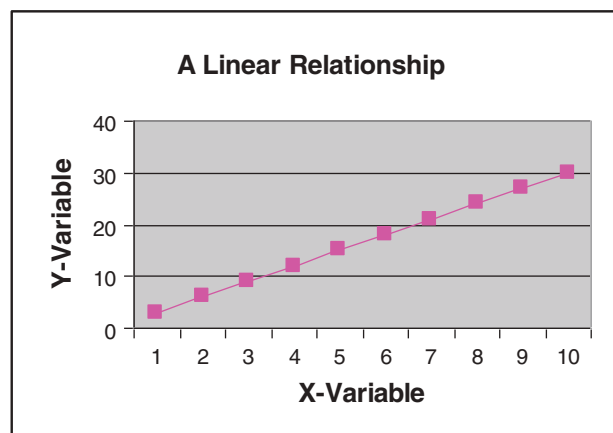


FIG. 14.5 A typical linear relationship: $Y = 3X$.

type of change is called a *geometric* progression. A geometric progression follows a straight line of increase when plotted on a graph (Fig. 14.5).

The relationship shown in Fig. 14.5 follows a simple geometric progression of increase; Y increases three units for each unit of increase in X . The defining elements of this relationship are the *slope* (3.0) and the *intercept* (where the trend line crosses the Y axis; zero in this case). Linear relationships were assumed by Sir R.A. Fisher in his parametric methods of analysis.

Parametric statistical analysis

Parametric statistical analysis was introduced in Chapter 1 in terms of a number of assumptions that underlay it. It was Sir R.A. Fisher’s purpose in proposing his radical methods in 1921 to bring some consistency to the analysis of data in medical studies at the Center for Eugenics at the University College of London (Fisher, 1921). Fisher elaborated his concern in 1922 (Fisher, 1922) by stating his concern that different Bayesian medical researchers could come to different conclusions from the same experimental data because they each brought a different set of past experiences and knowledge to the study (their *subjective priors*). To solve this problem, he decided to restrict his analysis to only those relationships present in the experimental data in a given study. He reasoned that the two most important aspects of a variable in a data set are its central tendency and the distribution of data values around this point. He chose the average (mean) value as the measure of central tendency, and the average difference (deviation) of each data value from the mean for that variable. Of course, he found that even if the data values were all positive, about half of the deviations were positive (for data values greater than the mean) and half of the values were negative (for data values less than the mean). Their arithmetic average was zero! But what he really wanted to express was the deviation itself, not its sign. So he just squared them (to remove

the negative sign), added them up, and divided by the number of values. Fisher recognized that representing the data values by the mean eliminated details in the data, so he decided to follow a convention of subtracting 1 from the number of values for each mean calculated in his statistical methods which ensued. This subtraction increased the average deviation slightly to account for the increased uncertainty of the mean as a measure of the data values in the data set. The result was termed the *variance*. The square root of the variance was calculated, in order to convert the variance value back to the original scale, which in turn yielded the *standard deviation*. The final formula for the standard deviation is,

$$\sqrt{\frac{\sum (X - \bar{X})^2}{N - 1}} \quad (14.1)$$

where, X is a data point, N is the total number of data points, $\bar{X}$ is the mean, and $\sum$ is the symbol for summing all of the squared differences from 1 to N .

The rest of Fisher's statistics (standard error, correlation coefficient, etc.) are just elaborations of these two *parameters* (mean and standard deviation). Hence, we see the origin of the term parametric statistical analysis. The next step was to derive a standard table of probability (the F-distribution) in 1924 (see Fisher, 1928), based on his definition of *likelihood*. The F-value was used to determine the significant difference between the two data sets.

The final step for Fisher was to develop a scheme for analyzing the variance in his experimental data set to determine if there was a significant difference between one medical treatment in his experiment and another treatment (based on his tables of probability). This analysis of variance (called ANOVA) became the basis for his

statistical conclusions about which treatments were *significantly* effective for each of several cases of a medical problem, and which treatments were not. This is the basic pattern of analysis for all of Fisher's parametric statistical procedures. Granted, this is a very simplistic explanation of the elements of Fisher's landmark works, but, it serves to set the stage for evaluating how to apply his methods in other experimental areas.

Assumptions of the Parametric Model

In order to achieve this analytical standardization, Fisher had to make a number of assumptions about the data he used. First, he assumed that the data sets he analyzed were normally distributed, following a bell-shaped curve (the assumption of normality). Second, he assumed that the predicted response followed a linear relationship to the predictor variables (the assumption of linearity). Third, Fisher assumed that the effect of one variable on the predicted value was independent of effects of any other variable in the data set (the assumption of independence).

Normality

An even more basic assumption than that of variable independence is the assumption of *normality*. Fisher's combination of deviations from the right of the mean with those from the left of the mean assumed that both sides of the distribution were similar. His probability tables also assumed such a distribution. The distribution this situation describes is called the *normal distribution*, shown in Fig. 14.6.

The normal curve shown in Fig. 14.6 is displayed with units on the X axis graduated in terms of standard deviation units. The curve represents the frequency of values

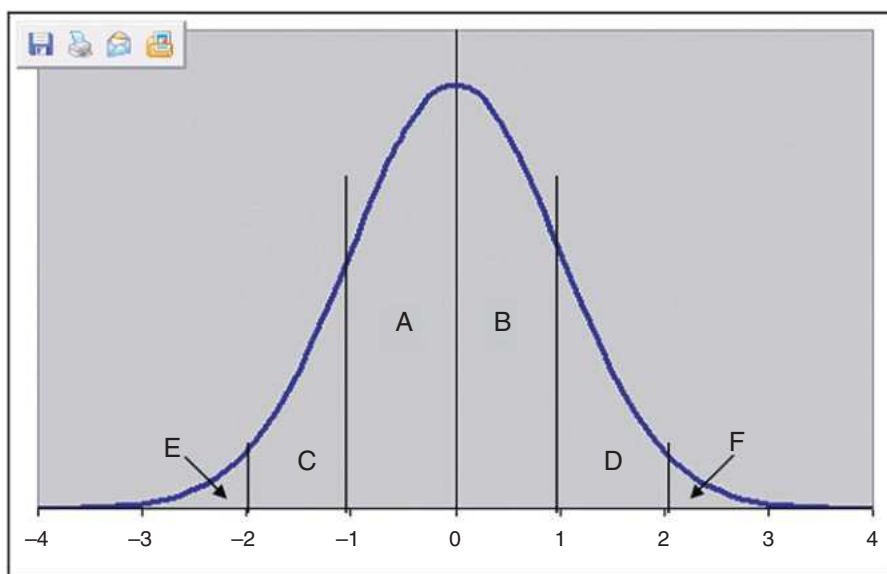


FIG. 14.6 The standard Normal (bell-shaped) curve. (Source: http://www.tushar-mehta.com/excel/charts/normal_distribution/)

for any point along the X axis. All of the area between the curve and the X axis represents 100% of the values in this distribution. The area under the curve between $X = -1$ (one standard deviation unit below the mean), and $X = +1$ (one standard deviation unit above the mean) represents about 68% of the area under the curve (areas A plus B). The area included by ± 2 standard deviations is about 95% (areas A + B + C + D), and the area included by three units is about 99.5% (areas A + B + C + D + E + F). Fisher's mathematics assumes that the distribution of values in each variable of the data set follows a normal distribution around the mean value. If an analyst uses any classical parametric statistical procedure, they are making the assumptions of normality and independency, however unconsciously. Significant departures from a normal distribution can lead to spurious conclusions inferred from the application of normal parametric. Significant departures from independency (effects of some predictor variables are strongly related to each other) are even more problematical.

Linearity

Consider the problem of predicting one variable with two other variables. The plot of predicted points in a linear regression is a straight line (Fig. 14.7).

This straight (diagonal) line is the best the linear algorithm can do to express the variation in the predicted values. Prediction of the data point plotted on the observed axis (Y) as 18.2 would be predicted along the diagonal line as 11, generating a 65% prediction error.

Independency

Fisher was fortunate to have at his disposal a medical laboratory in which he could conduct very *controlled* experiments. It was necessary for all methodological and environmental effects to be held constant, varying only one variable in the actual treatment of the patient. Other studies could study the effects of varying methods or environmental conditions for a given medical treatment. For example, he could study the effects of room temperature on the activity of a drug used in a given treatment. To do this, he had to hold all other variables constant (including the treatment) and vary only the room temperature. Then he could do likewise for humidity and so forth. This experimental approach ensured that the recorded effects of each variable were *independent* of each other.

Fixes for violation of the normality assumption

A common treatment of nonnormal distributions is to transform the data with various utilities in the data mining or statistical tool package. Common transforms include:

- Beta distribution
- Gamma distribution
- Binomial distribution

Normality and the Central Limit Theorem

Some practitioners believe that drawing a random sample from a nonnormal distribution will create a

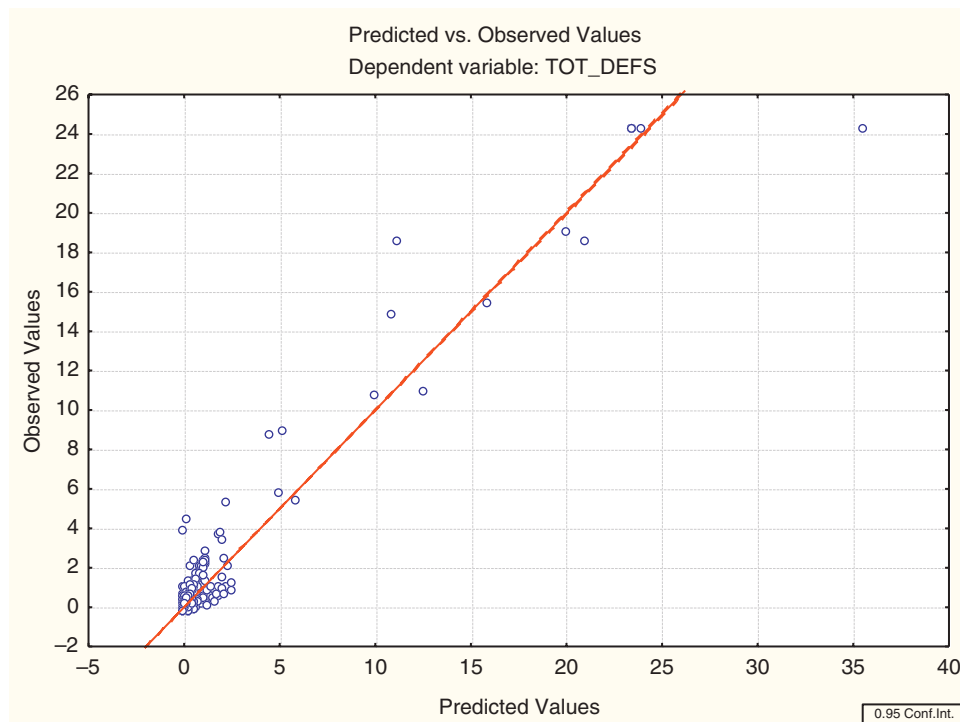


FIG. 14.7 Plot of a three-factor regression.

normal distribution, but this is not true. The Central Limit Theorem states,

A group of means of n samples drawn from a nonrandom distribution approaches normality as n approaches infinity.

This theorem is often misinterpreted by naïve practitioners. Careless reading of the theorem leads some people to believe that all you have to do to get a normally distributed data set is to sample a nonnormal population of data, and magically, the sample is normally distributed. That is not the case! The theorem states that the *group of means* taken from a nonnormal distribution is normally distributed. It is generally accepted that at least 100 random samples are necessary to create a distribution that is very close to normal. That means that if you take 100 samples of a nonnormal distribution and calculate the mean for each of them, the distribution of the 100 means is close to being normally distributed. This attribute of sampling can be applied when you take multiple samples of a population and submit the data to linear regression analysis. Don't look for quick ways out of this assumption. If your data set distribution is significantly different from normal, it can ruin any analysis based on parametric statistical methods. And the most painful part of this problem is that you don't know when you are wrong! All the statistical tests may show a strong relationship in your model, but it may fail miserably when you try to apply it.

Fixes for violation of the assumptions of nonlinearity and independency

The second major assumption inherent in classical parametric procedures is that the variables have a *linear* effect on the response variable (the target). This means that a plot of the relationship of any variable to the response variables is a straight line. Examples of common statistical procedures that make these three assumptions are ANOVA and linear regression. Many fixes exist for violations in the assumptions of linearity and independency in data sets. We will consider them only after a short review of linear regression to provide the proper context for understanding them.

Linear Regression

Linear regression was first proposed by Sir Francis Galton (1822–1911). Galton coined the term *regression* to describe the observation that the majority of very tall fathers had sons who were shorter, and most very short fathers had sons taller than they. The trend of this progression in height was toward the average (or mean) height. This phenomenon was termed *regression to the mean*. His analysis of this effect became known simply as regression.

The major objectives of linear regression are:

- To determine if a relationship exists between one variable and another (or a set of others)
- Describe the nature of this relationship if it exists

- Quantify the accuracy of this relationship
- Evaluate the relative contributions of each variable if multiple variables are used

Linear regression makes all three assumptions described above. But you can appeal to the Central Limit Theorem to correct for nonnormality by taking multiple samples and working on the means, rather than the original data. That is, you can generate a group of data for a given variable by drawing a group of samples, of the population, finding the mean, drawing another group of samples, finding the mean, and so forth until you have a group of means representing that variable. For a multivariate data set, you must perform this sampling operation on all variables you want to use in the analysis. That group of means will be normally distributed. Engineering control charts rely on that property by making many samples of a process flow and creating this group of means. The control charts are based on the group of means, not the underlying data from which they were calculated.

The basic process of linear regression of a data set is to estimate values (named *coefficients*) for each candidate predictor variable (X) to represent the effect that variable has on the response variable (Y). These effects are assumed to be linear and additive (that is, you add them up across all variables to get the total effect). The general form of the regression equation generated by the analysis is:

$$Y = a + b_1X_1 + b_2X_2 + b_3X_3 + \dots + b_nX_n \quad (14.2)$$

In Eq. (14.2), the variable “ a ” is the Y value where $X = 0$ (the *intercept*), b_1 is the coefficient for X factor #1 (X_1), b_2 is the coefficient for X factor #2 (X_2), and so forth through the last X variable (X_n). The coefficients reflect effects of two sources of relationship of the X variable to the dependent variable (Y): the relative effect of X on Y , and the effect of differences in scale between X and Y . If variable Y and all of the X variables have the same scale, then the coefficients reflect the relative predictability of each X variable. But if the scales are different, the coefficients may not reflect much of the relative predictabilities. Usually, the scales of variables are quite different. One technique for overcoming the effects of scale is *standardization*. This is a process of transforming all variables to a common scale. One common standardization method is to subtract the mean from a value and divide by the standard deviation (named the *z-score*), which creates a scale in terms of standard deviation units. Another method used to create a common scale from 0 to 1.0 is to normalize the variable over its range as $\text{new\_value} = (\text{old\_value} - \text{minimum\_value}) / (\text{maximum\_value} - \text{minimum\_value})$.

The concept of the response surface. Another way to look at the relationships among the variables is to look at a response surface in three-dimensional space.

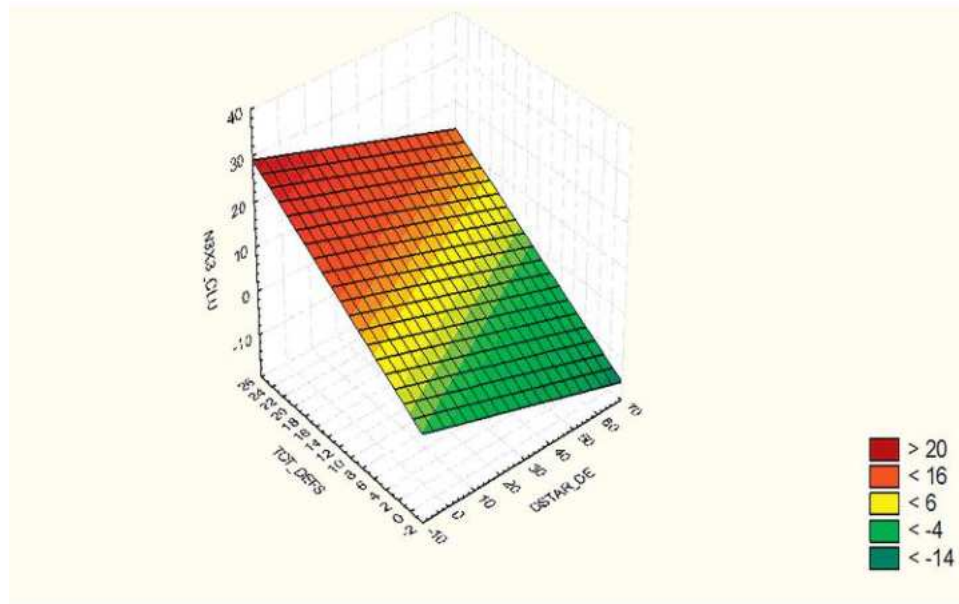


FIG. 14.8. Fit of a linear three-factor response surface.

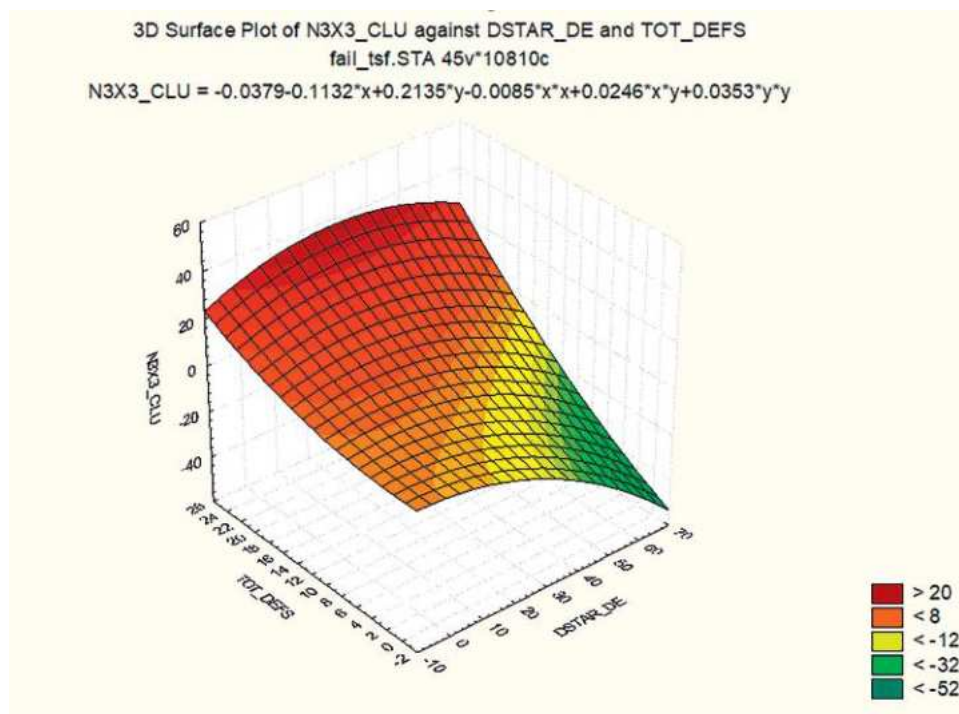


FIG. 14.9 Plot of a quadratic fit of three variables.

Fig. 14.8 shows the relationship as a plane when a linear function is used to represent the data relationships.

Fig. 14.9 shows the fit using a quadratic function, rather than a linear function.

By fitting even more complicated functions to the three data values, you can conform the response surface even more closely to the data. The plot of the three-factor

response surface using a negative exponential (an even more nonlinear function) is shown in Fig. 14.10.

The departure of a data point from its predicted value is called the *residual*. One of the common reports of a multiple regression algorithm is a plot of the residuals versus the predicted values, shown in Fig. 14.11.

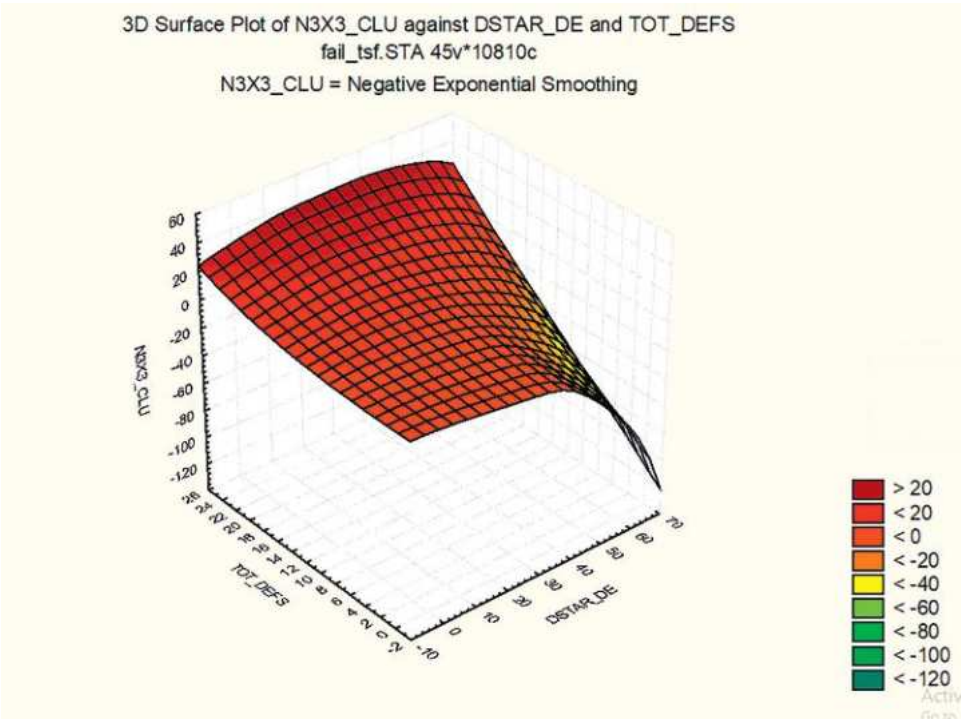


FIG. 14.10 Plot of a negative exponential smoothing function applied to fit three variables.

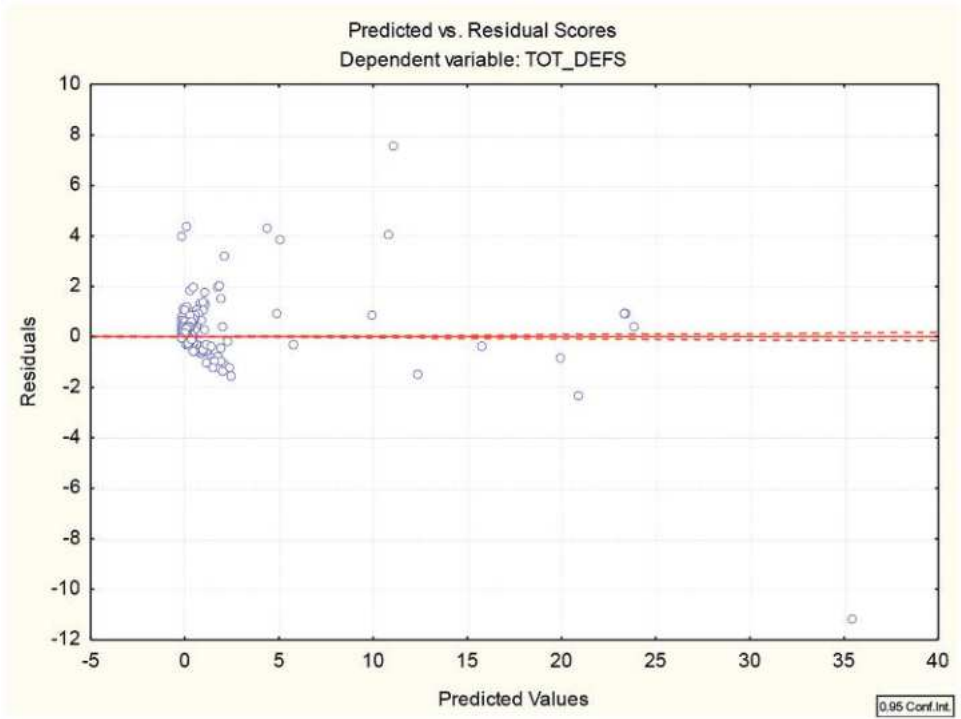


FIG. 14.11 Plot of the residuals versus predicted values.

The ideal plot of residuals versus predicted values would be a long cluster along the X axis at $Y = 0$ (residual = 0). In Fig. 14.11, you can see that most of the values are near the ideal. However, there are several predictions that

differ significantly from the raw data values (have a large residual). This plot is a visual form of model evaluation. Another visual reflection of the strength of the model is the normal probability plot, shown in Fig. 14.12.

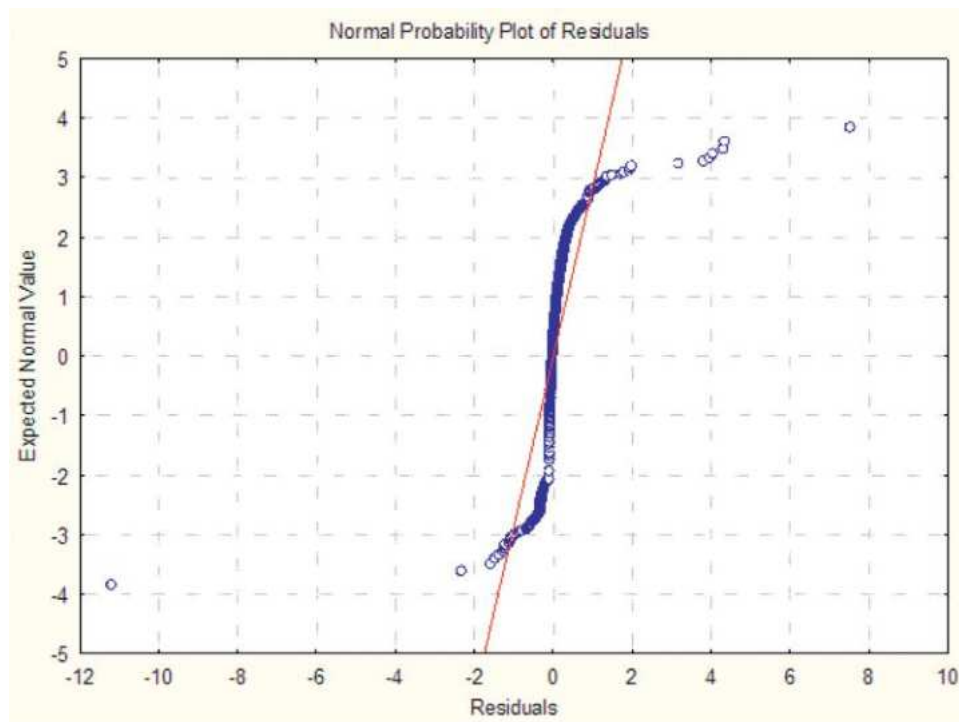


FIG. 14.12 Normal probability plot of the three-factor regression.

The normal probability of a residual is the expected value of the residual based on the normal distribution, calculated as:

$$P_i = i / (N + 1) \quad (14.3)$$

where i is a residual value in a list and N is the number in the list.

The ideal normal probability plot of residuals is a straight line. The red line in Fig. 14.12 is the linear fit through the residual data points. This normal probability plot appears relatively close to normal, except for a few data points at the upper end. This means that the underlying regression which generated the residuals is likely to be valid.

General linear models (GLMs)

The multiple linear regression (MLR) models suffered from a number of constraints and assumptions which could not be varied lest the inferences from the model become invalid. Much work in the mathematical and statistical community over the last 20 years has provided a much more flexible framework for analyzing data in a regression context. This work has successively expanded and generalized the MLR model by:

1. Permitting the Y variable to be replaced by a set of values. Therefore multiple dependent variables could be analyzed with the same solution techniques as single dependent variables.

2. Permitting each X variable to be replaced by a set of values.
3. Providing a method for analyzing the sets of values by matrix algebra rather than standard arithmetic.
4. Providing for linear transforms of the Y matrix and the X matrix to perform high-order Polynomial Regressions.
5. Providing a number of methods for coding categorical predictors to accomplish the same goal as dummy variables without increasing the number of variables.
6. Providing a method for overcoming the linear independency assumption of MLR but allowing a solution of normal equations with a *generalized inverse* operation. A normal matrix inversion is very restrictive (you can do it only one way). But a generalized inverse operation can be done in many ways, leading to many possible solutions.
7. Providing methods for handling redundant predictors.

The combination of these provisions of the GLM approach removes the most significant limitations of MLR. The matrix architecture of the analysis operations permits full-factorial designs ($N \times N$ variable analyses) to incorporate *all interactions* between predictor variables to be evaluated for effects on the dependent variable. These *interaction effects* can be combined with the *main effects* (attributable to the independent effects of the predictors).

A wide variety of experimental designs can be accommodated by GLMs. Examples of these designs include:

- Classical ANOVA designs to assess the $N \times N$ effects of factorial experiments.
- ANCOVA (analysis of covariance) designs with both continuous and categorical predictors can be evaluated. ANCOVA designs permit the evaluation of the *significance* of the interactions in factorial designs.
- Mixed models of ANOVA and ANCOVA can be accommodated.

There are many more experimental designs that can be evaluated by GLM analysis, making it the most flexible parametric procedure to use, with the fewest assumptions. If you have a normally distributed data set, a GLM analysis may be the best way for you to go. Data mining and machine learning methods (see below) may be able to handle more complex response surfaces, but the techniques for evaluating models are far fewer than those available for parametric techniques. See Chapter 15 on Model Evaluation and Enhancement for a discussion of some of these techniques.

Methods for analyzing nonlinear relationships

In a nonlinear relationship, the trend line of Y plotted against an X variable is *not* a straight line but rather, it is a curved line, as shown in Fig. 14.13.

Fig. 14.13 shows the relationship between not a multiple of X (as it was in the geometric progression), but according to the natural logarithm ($\ln$) of X . Notice that the slope of the plotted line is not constant, it can be determined at only a given point on the curved line. Most relationships in nature and in the business world are intrinsically nonlinear, rather than linear in nature.

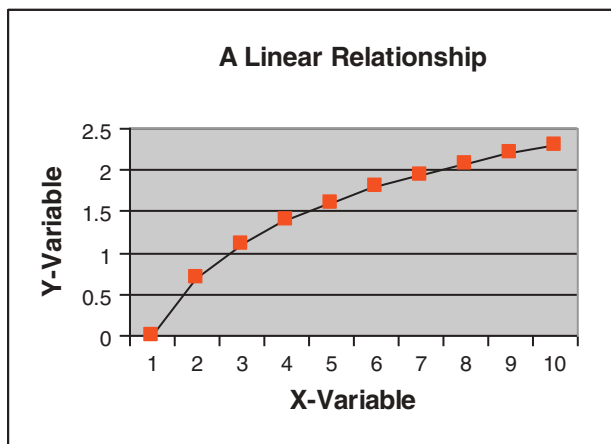


FIG. 14.13 A plot of a nonlinear relationship ($Y = \ln X$).

Nonlinear regression and estimation

It is possible to fit a nonlinear function simply by replacing some of the variables with polynomial terms, which include that variable. This approach is called Polynomial (or Curvilinear) Regression. For example, if we replace predictor variable X with X^2 , or some higher-order polynomial, the regression equation can account for some nonlinear effects in X . This procedure is usually a trial-and-error process, because it is difficult to know ahead of time which polynomial to use. A GLM model can be configured to do this easily. Contrary to what it appears to be, it still uses a linear function to fit the data, and we should not call this nonlinear regression. Techniques like this are often referred to as *intrinsically linear regression models*.

An intrinsically nonlinear regression model uses an arbitrary nonlinear function to replace one or more of the variables. This nonlinear function has no exact solution but rather, the parameters of it must be estimated; hence the better name for it is nonlinear estimation. These estimation procedures make a number of passes through the data set and minimize an error function along the way. Any one of a number of techniques can be used to find the “minimum” error (e.g., Least Squares, Maximum Likelihood, Quasi-Newton method). There are a number of common nonlinear estimation techniques, which can be very useful in data mining applications. These techniques include:

- Logistic (or Logit) Regression
- Probit Regression
- Poisson Regression
- Piecewise Linear Regression

Logit and Probit Regression

Logistic Regression is actually a form of classification in its output, but it uses regression mathematics to get there. Probit Regression is similar to Logit Regression, in that it too has only two possible outcomes, but there is a “fuzziness” associated with these outcomes. Probit data are often integers used in a multipoint scale to measure responses. A five-point scale might be defined as: 5 = strongly agree; 4 = generally agree; 3 = neither agree nor disagree; 2 = generally disagree; and 1 = strongly disagree. Actually, this scale reflects a “feeling” about one of two possible outcomes, agree or disagree. The distribution of these responses can be transformed to reflect the appropriate area under a normal probability curve (assuming a normal distribution, of course), and they can be analyzed using the Probit model in Equation 4.

$$NP(\text{feeling}) = NP(b_0 + b_1 * x_1 + b_2 * x_2 + \dots) \quad (14.4)$$

where NP is the normal probability or space under the normal curve.

Poisson Regression

Poisson Regression uses the Poisson distribution (rather than the normal distribution) to express data relationships. The Poisson distribution fits count data well, such as attendance counts on different days or for different events.

Exponential distributions

The normal and Poisson distributions are types of exponential distributions because they include an exponential factor (a value with an exponent). These distributions can be classified according to two parameters: a *dispersion parameter* and an *index parameter*. For a detailed discussion of these parameters and the distributions they express, see [Jørgensen \(1987\)](#). For our purpose below, we can classify a distribution by its index parameter p :

- $p = 0$...Normal distribution
- $p = 1$...Poisson distribution
- $p = 2$...Gamma distribution
- $p = 3$...Inverse Gaussian distribution

One of the common problems in data mining is to model the occurrence of significant events. The significance of this occurrence is composed of two components: frequency of the event and severity of the event. Statistical analysis of these significant events requires the calculation of probabilities. The calculation of the probabilities for frequency follows the Poisson distribution and that for severity follows a log of the normal distribution. You can

calculate the probabilities easily enough according to the properties of each exponential distribution. The problem arises when you try to combine them. The calculated probabilities are *not* additive! You must find a way to combine inferences from the different probabilities. The most common application where this must be done is in the modeling of insurance credit risk.

There are three ways to model problems like insurance risk:

1. Model frequency and severity separately, using different algorithms, and then report on them separately.
2. Use a distribution, like the Tweedie distribution (see [Jørgensen, 1987](#)), which has a p value of between 0 and 1. This is a compromise between the normal and Poisson distributions.
3. Use Transform Regression, a technique to analyze a probability defined using elements of the mathematical expressions of both the normal and Poisson distribution (see [Pednault, 2001](#)).

Piecewise linear regression

Piecewise linear regression fits a linear regression on a number of portions of a nonlinear response curve. Piecewise linear regression curves up a nonlinear relationship into a number of linear ones. Consider the inverse logistic curve that is typical in many business data sets (e.g., telecommunications minutes of use – MOU) with three linear functions fit to it ([Fig. 14.14](#).)

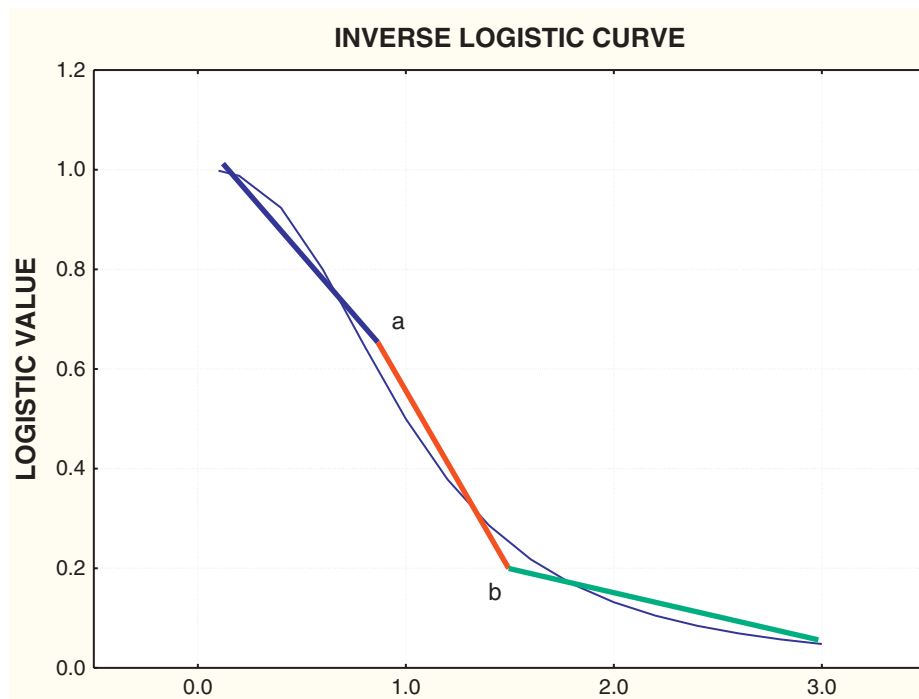


FIG. 14.14 Piecewise linear segments expressing the nonlinear inverse logistic curve.

Conceptually, we can see in Fig. 14.14 how piecewise linear regression preprocesses the data. First, it determines appropriate break points along the line at (a) and (b), and defines the three straight lines shown in the figure. Next, the algorithm fits an ordinary linear regression to each of the three lines and presents the results for the combined relationship. Essentially, this is the approach that C&RT follows to build a regression tree. Random Forest for Regression uses the same approach.

There are many other nonlinear estimation techniques not included in this handbook. For more information on other common techniques, the reader is referred to Denison et al. (2003).

Methods for handling variable interactions in linear regression

The earliest approach to correct for variable interactions was to use factorial designs to separate “main” effects from “interaction effects.” The combined interaction effect, termed as C, was added to the ANOVA analysis equations to “correct” the interactions. Application of this approach to linear regression is not strictly appropriate because the calculation of the variable coefficients includes the effect of interactions with other variables. If an interaction between two variables is obvious, an additional variable can be derived as the product of the interacting variables. When these multiplicative terms are added to the regression equation, it may significantly increase the collinearity of the two variables. The only way to avoid the inclusion of variable interactions is to do what Fisher did in his medical laboratory by recording the response of the Y variable to different values of a single (X) variable while holding all other variables constant.

Collinearity among variables in a linear regression

When two variables are highly correlated to each other, the plots of these variables lay on nearly the same line; this effect is called *collinearity*. The total of all the collinearity between variable pairs is called *multicollinearity*. You can assess this effect by comparing the square of the sum of the Pearson simple correlation coefficients for all variables with the coefficient of determination (R^2). The R^2 -value measures the combined effect of all the variables in explaining the variance in the dependent (response) variable. The Pearson simple correlation coefficients measure the degree of correlation between a single variable and the dependent variables. The degree to which the squared sum of the simple correlation coefficients exceeds the R^2 -value is a measure of the amount of collinearity between the variables. Another metric used to assess the degree of multicollinearity in a relationship is the Variance Inflation Factor (VIF) described by Shrestha (2020) expressed in Eq. 14.5

$$\text{VIF} = 1/(1 - R_i^2) \quad (14.5)$$

Relatively high multicollinearity in a regression analysis (e.g., $\text{VIF} > 5$) will make it difficult, if not impossible, for the algorithm to find a single optimum solution. Because the interacting effects vary with the values in the variables, there may be several “optimum” solutions, depending on the relative frequencies of values among collinear variables. As a result, the covariance matrix cannot be solved by matrix inversion, and the process fails. A good rule of thumb to follow in parametric statistical analysis is to eliminate one member of any pair of variables that is more than 80% correlated with the other. The other suggestion we can make is to limit the number of interaction variables to only those that are obvious.

Machine learning algorithms used in numerical prediction

The most common machine learning algorithms used for predicting continuous response variables are:

- C&RT
- Neural Nets
- Decision Trees and Random Forests
- SVMs and other kernel techniques

These other algorithms were discussed in greater detail in Chapter 13.

Common machine learning modeling preliminary operations

Machine learning algorithms are not affected by violation of the three major assumptions of Fisher’s Parametric Model. There are other considerations, however, that must be addressed with preliminary operations before a model can be trained:

- Input data sets must be partitioned for use with the algorithms.
- Algorithm hyperparameters should be optimized.

Partitioning the input data set

Machine learning algorithms work by training a number of candidate models with a training data set and testing each model with a testing set. It is best to use different data sets partitioned for these operations. These two data sets are created by random sampling of the input data set, often in the proportion of 70:30. The 70% sample is input to the modeling algorithm and is used to train the model. Some predictive analytics tools require that two data sets be partitioned, with an option to partition a third set for validation (e.g., IBM SPSS Modeler, STATISTICA Data Miner). Machine learning algorithms in KNIME require such a partitioning operation for model training, but they do it automatically in the ratio of 50:50. Therefore we have to create only two

data sets in the Partitioning node: (1) a modeling data set (to be split internally by KNIME to form the training set and the testing set), and (2) the validation set for evaluating the performance of the trained model. The best approach in building machine learning models is to use three data sets, one data set to train the model, a second data set to test the accuracy of the model between iterations of the algorithm, and a third data set to evaluate the prediction accuracy of the trained model. Care must be taken not to evaluate a trained model on data used in any way to train or test the model.

Hyperparameter optimization

For example, most decision tree algorithms provide a list of hyperparameters that can be adjusted. Common hyperparameters include:

- Number of estimators: # of trees built per iteration
- Learning rate: the rate of change in internal training parameter, how fast the algorithm
- Gamma: minimum loss required to partition a node further
- Maximum depth of the decision tree
- Minimum number of leaves in a terminal node
- Lambda and alpha: regularization parameters
- Scale_pos_weight: a weight value for a target class = #positives/#negatives

Python code to optimize maximum depth of a decision tree

An example of the Python code to optimize the maximum depth of a Gradient Boosted Tree is listed below.

```
param_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
for max_depth in param_list:
    classifier = GradientBoostingClassifier(
        (n_estimators=100, learning_rate=0.07,
         max_depth=max_depth, random_state=4)
    classifier.fit(X_train, y_train)
    print("max_depth: ", max_depth)
    print("Accuracy score (training): {0:.3f}".
          format(classifier.score(X_train, y_train)))
    print("Accuracy score (testing): {0:.3f}".
          format(classifier.score(X_test, y_test)))
    print("Accuracy score (validation): {0:.3f}".
          format(classifier.score(X_val, y_val)))
```

Table 14.1 shows the accuracy values for each of the 10 values for maximum number of branches of the decision tree (max_depth) for the colon polyp model described in Chapter 23.

Notice in Table 14.1 that the training accuracy increases to 100% in trees with 10 branches, but the validation accuracy doesn't change very much. The high training accuracy values indicate that these models are over trained on the training data. The best model appears

TABLE 14.1 Training, testing, and validation accuracy values with 10 values of max_depth.

max_depth value	% Training accuracy	% Testing Accuracy	% Validation accuracy
1	70.0	67.5	66.0
2	74.8	71.7	67.0
3	78.1	72.6	67.0
4	83.6	72.6	67.6
5	88.6	72.0	67.2
6	94.5	71.0	67.2
7	97.6	71.6	67.0
8	99.1	71.3	66.6
9	99.9	71.3	67.5
10	100	71.6	67.0

to be for max_depth = 4, shown in the shaded row. This model has a relatively low likelihood of being over-trained; therefore it was the model selected. Python also provides a GridSearchCV method in the sklearn.model_selection library for including a series of hyperparameter optimizations in a processing loop.

The parameter optimization loop in KNIME

Another tool that can be used to optimize hyperparameters in a processing loop is the Parameter Optimization loop processing structure in KNIME. Fig. 14.15 shows a KNIME workflow for optimizing three hyperparameters by analyzing all combinations of the three parameters. Data input was provided by the colon polyp model described in Chapter 23.

Fig. 14.15 shows the hyperparameter combinations with the ten highest number of correct predictions (Obj.), showing that the best maxDepth is five branches in a tree using a learning rate (eta) = 0.454.

Postscript

This discussion highlights several disturbing characteristics of modeling algorithms. First, some algorithms don't provide much insight to *how* they do their jobs or *what* important factors were used to form their predictions. Second, most of the algorithms require extensive tuning of their hyperparameters to yield the best models. These drawbacks of data analytics modeling point to the truism that data analytics is as much an art as a science. Good models are not so much about what the modeler does as it is about what the modeler does *right!* The challenge before the modeler is to "map" the usage

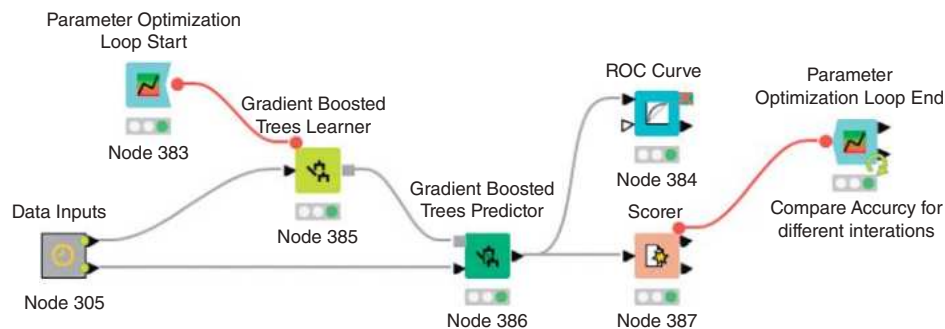


FIG. 14.15 A KNIME workflow for a parameter optimization loop for maxDepth and eta (learning rate).

of the right algorithms in the right way to work with data in a specific data domain. Chapters 15–17 provide a deep dive into more sophisticated algorithms that give additional background for the presentations in Chapters 18–19, which tackle the need for explainable algorithms and the need to keep the human in the modeling “loop.”

Keywords and phrases

Regression, Classification, Algorithms, Linear, Non-linear, Relationships, Assumptions, Hyperparameter, Interactions, Machine learning, Neural network, Decision tree, Predictor variables, Normal distribution, Parametric statistical.

References

- Denison, D., Hansen, M., Holms, C., Mallick, B., & Yu, B. (Eds.), (2003). *Non-linear estimation and classification*. Springer, 474p.
- Fisher, R. A. (1921). On the “probable error” of a coefficient of correlation deduced from a small sample. *Metron*, 1, 3–32.
- Fisher, R. A. (1922). On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society (A)*, 222, 309–368.
- Fisher, R. A. (1924). On a distribution yielding the error functions of several well-known statistics: *Proc. Internat. Congress mathematicians*. Univ. Toronto Press. (1928). pp. 805–813.
- Foot, K. 2021. A brief history of machine learning. <https://www.dataversity.net/a-brief-history-of-machine-learning/>. (accessed 4/6/2023)
- Gartner. (2012). What is Data and Data Analytics? <https://www.gartner.com/en/topics/data-and-analytics>
- Jørgensen, B. (1987). Exponential dispersion models (with discussion). *Journal of the Royal Statistical Society: Series B (Methodological)*, 49, 127–162.
- Pednault, E. 2001. Transform Regression and Klotmogorov Superposition Theorem. Proc. 6th SIAM Int. Conf. on Data Mining. http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/p/Pednault:Edwin_P=D=.html
- Rosenblatt, F. (1958). The perceptron: a probabilistic model of information storage and organization in the brain. *Psychological Review*, 65(6), 386–408.
- Shrestha, N. (2020). Detecting multicollinearity in regression analysis. *American Journal of Applied Mathematics and Statistics*, 8(2), 39–42. <https://doi.org/10.12691/ajams-8-2-1>.
- Wright, S. (1921). Correlation and causation. *Journal of Agricultural Research*, 20, 557–585.

Model evaluation and enhancement

Preamble

One of the most common questions asked by beginning data miners is “How do I know when my model is any good?” This chapter will introduce you to a number of model metrics that you can use to measure the “goodness” of your model. We will provide a checklist of actions you can employ to improve model performance. Using a reliable technique for model assessment is essential for finding the best model and being confident in its performance. Some of the model enhancement techniques presented in this chapter include:

- Cross-validation resampling method
- Bootstrap resampling
- Jackknife resampling
- Boosting
- Bagging

Evaluation and enhancement: part of the modeling process

In [Chapter 3](#), we presented the CRISP-DM process model, which includes an Evaluation phase. The overall process must be expanded to show where model enhancement fits in. As maintained in [Chapter 3](#), this modeling process is not linear but *iterative*. It is very rare that the first model trained will be the best. Often, the evaluation process will raise some issues that can be resolved by making changes in some tasks in the data preparation or modeling phases. These changes may help to enhance the predictability of the model or its usability in deployment.

For example, the initial model may not include a variable that is needed in the deployment process (e.g., Zip Code, necessary to partition the sales contact lead). Most modeling packages can pass along variables like Zip Code not used in training the model. Some of the responses, however, might be related to Zip Code.

A summary of the modeling process that includes evaluation and enhancement is shown in [Fig. 15.1](#).

This process may consist of many iterations; thus we must view them as a single integrated operation. That is why the activities of model evaluation and enhancement are included in the same chapter.

When you use evaluation criteria, they may lead you to make changes (enhancements) in the model and then rebuild it. We will discuss a number of common model evaluation metrics, some of which will suggest immediate enhancements of the model. Afterward, we will discuss higher-order enhancements, which may have no direct connection with any evaluation metric.

Types of errors in analytical models

Errors can be classified into two general types: Random error and Systematic error. Random error is attributable to random variation in the target value “signal” caused by factors that cannot be accounted for by the predictor variables of even a perfect model. But no model is perfect. All modeling algorithms have strengths and weaknesses, which cause errors in the predicted values. Systematic error is caused by weaknesses in the random sampling and processing operations of the modeling algorithm or errors in model specification.

We will consider evaluation methods based on:

- Analyses of the predictive power of the model, apart from error
- Analyses of random error in the model
- Analyses of systematic error in the model

Evaluation of models based on predictive power

For many purposes (e.g., direct mailing campaigns), the most important consideration is not the error of the model but its success in predicting the target value. In addition, overall (global) accuracy of prediction may not

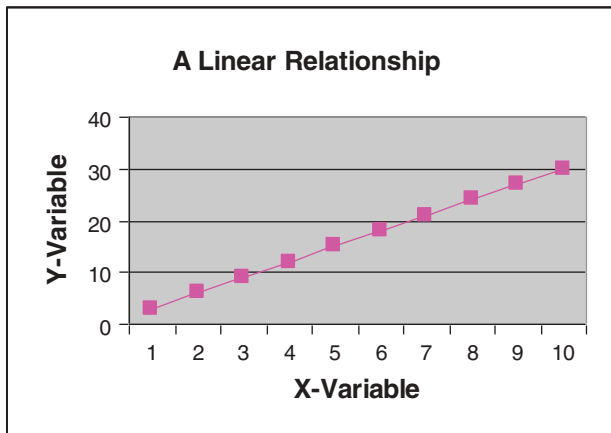


FIG. 15.1 The relationship among model training, model evaluation, and model enhancement composed into an iterative modeling process.

be as important as the accuracy of predicting the positive state of the target variable (e.g., those customers who bought a service in the past). For example, many customer relationship management (CRM) models seek to classify customers into two groups: those who responded and those who did not. This is a *binary* classification problem in that there are only two outcomes.

Evaluation of classification errors

In addition to total error, we need to know the error rates for each class in a classification analysis. These errors (and other metrics based on them) are all derived from the coincidence (or confusion) matrix.

Coincidence matrix

The coincidence (confusion) matrix is a table showing the relationship between observed and predicted values in a classification problem. The coincidence matrix was introduced in [Chapter 9](#), but we will review it here to provide a basis for the definition of some additional metrics used commonly in the evaluation of predictive models.

Consider a manufacturing operation in which 92 out of 2911 units were returned as failures. A model was built to predict these failures from product specifications and results from the manufacturing and testing operations. [Table 15.1](#) shows the coincidence matrix of the observed and predicted failures.

TABLE 15.1 Coincidence matrix for industrial failures.

	Predicted	
	Y	N
Observed		
Y	72	20
N	21	2798

The coincidence table is the basis for calculating a number of accuracy metrics. The four error metrics calculated from a confusion matrix were also introduced in [Chapter 7](#) (Classification). In review, each of the following accuracy metrics is defined in terms of four prediction categories:

True-Positives (TP) = Number of correct positive predictions (Y) in [Table 15.1](#).

This situation occurs when a case with an observed category = Y is predicted as Y (this is the number 72);

True-Negatives (TN) = Number of correct negative predictions (N) in [Table 15.1](#).

This situation occurs when a case with an observed category = N is predicted as N (this is the number 2798);

False-Positives (FP) = Number of cases where the predicted category is Y and the observed category is N (this is the number 21);

False-Negatives (FN) = Number of cases where the predicted category is N and the observed category is Y (this is the number 20).

- Global accuracy
- The overall (or global) accuracy of prediction is equal to the total number of true predictions divided by the total predictions, expressed as:

$$\text{Global accuracy} = \frac{\#TP + \#TN}{N}$$

The global accuracy is an overall measure of predictive power and is the most commonly used to display model accuracy in data mining tools. Problems with global accuracy are:

1. Assumes that the error rates among classification classes are equal.
2. An accuracy of 99% can be good or it can be really bad, depending on the nature of the problem and the differential costs of both errors.

Other measures based on the four situations above permit a more refined view of predicted categorical values.

Precision (P) = $\#TP / (\#TP + \#FP)$. This metric expresses the proportion of positive predictions that are correct. The precision of a model reflects the degree of confidence that we can attribute to the positive predictions. Sometimes this is the only metric that can be used reasonably in a project in order to serve the goals of the analysis. For example, in insurance risk analysis, false-positive predictions are those people you insure who had large claims caused by factors you might have been able to include in the initial risk analysis. Insurers carefully study those insurance contracts that cost the company a lot of money. They will try to find some commonalities among large payees to gain some insight

into modifying insurance contracts. Insurers will track model precision and hope to see increases through time.

Recall (R) = $\#TP / (\#TP + \#FN)$ aka *True-Positive Rate*. This metric expresses the proportion of the observed positive values that were predicted as positives. The $\#FN$ predictions were labeled as positives but were predicted as negatives. Therefore this metric reflects the proportion of total positive observations that were predicted correctly by the model.

Sensitivity (S) = $\#TP / (\#TP + \#FN)$ - aka *True-Positive Rate*. This metric expresses the proportion of the observed positive values that were predicted as positives. Note that Recall and Sensitivity are terms for the same error but used in different application fields.

Specificity = $\#TN / (\#TN + \#FP)$ - aka *True-Negative Rate*. This is a very useful metric to express the effect of the number of negative observations that were not predicted correctly.

F-value

This metric is often used in statistical analysis and can be calculated as the ratio of the variance between groups and the variance among groups. In predictive analytics, the among and between variances are expressed in effect by the metrics of Precision and Recall calculated as:

$$F\text{-measure} = 2 * [(P * R) / (P + R)] \quad (15.1)$$

where P is the Precision and R is the Recall.

F -values are used in statistical analysis for significance checking. In predictive analytics, the F -value is used as an overall measure of prediction error.

Receiver operating curve (ROC)

This metric was initially developed by the British in World War II. They built a series of radar detectors to identify incoming German planes (true-positives). The radar detectors, however, could also detect flocks of birds and other false-positive signals. They designed a metric to help them track the effect of false-positive radar signals. They expressed the metric as the plot of the true-positive (TP) rate over the false-positive rate. The false-positive rate is calculated as $1 - \text{specificity}$ and plotted in Fig. 15.2.

The total area under the ROC curve (AUC) is useful for showing graphically the relative predictive power of a model. It has, however, the same disadvantage of the global accuracy – misclassification costs are not considered (Hand, 2009).

The processing of binary classification algorithms generates two probability numbers ranging from 0 to 1, not a category. One of these numbers reflects the probability that the prediction belongs to the positive class (represented by 1), and the other number represents the

probability that the prediction belongs to the negative class. Analytical tools set a threshold for the positive probability (usually 0.5), equal to or above which the classification is set to 1, and below which the classification is set to 0. Most analytical tools can output the classification probabilities in addition to the classifications. These classification probabilities are used to generate a graphic (the *Gains curve*).

Gains curve. A gains table is generated by sorting the classification probabilities in descending order, and collecting them into 10 bins of an equal number of records, called *deciles*. The number of records in each decile that are TP predictions is calculated for each decile. The proportion of number of TPs in each decile is divided by the total number of TPs across all deciles and plotted against decile number (or proportion of the population). Then the proportion of responses captured is accumulated up to each decile and plotted against the decile proportion in a graph. Fig. 15.3 shows a cumulative gains curve.

The form of the gains curve in Fig. 15.3 is similar to the ROC curve in Fig. 15.2, but they are calculated differently and used in different applications. The ROC curve expresses the predictive power of a model with a categorical target. The Lift curve can express the predictive power of a real number target. Predictive power of models with categorical targets can be assessed also with gains curves if the predicted categories are associated with a probability of classification. The theory behind the gains curve is that most of the high probabilities for Target = 1 should be in the top 2 or 3 deciles. This means that marketers have to contact only 20% to 30% of the customers in such a scored list sorted on prediction probability in descending order to capture the large majority of responders. The diagonal line in Fig. 15.3 is the line of random expectation, which expects that 1/10 of the total number of true responders is expected by chance in each decile. You can use the noncumulative response numbers also to generate a *lift curve*, showing the actual proportion of total TP prediction in each decile are converted to lift index values (Fig. 15.4). In this case, the values are not summed to each decile level.

Evaluation of models according to random error

We can express the total of the random error and systematic error mathematically, but it is very difficult to *distinguish* between them in practice. For example, the general form of a regression model is:

$$Y = a + b_1X_1 + b_2X_2 + b_3X_3 + \dots + b_nX_n + \text{Error} \quad (15.2)$$

where a is the slope intercept, the X values are the predictor variables, and the b values are the coefficients associated with each X value.

Comparing ROC Curves

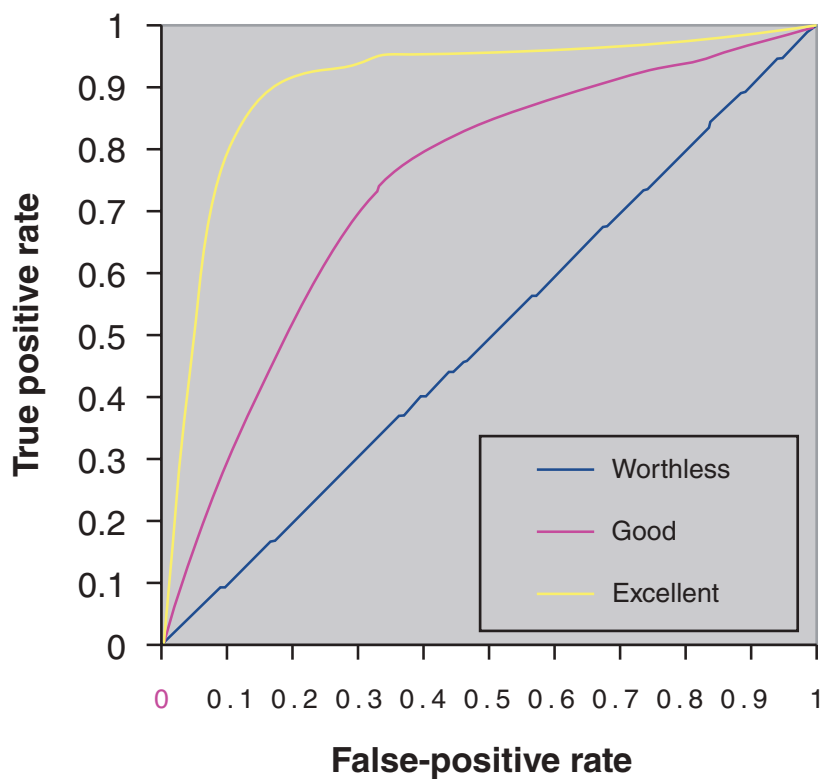


FIG. 15.2 Sample ROC curves. The area under the yellow curve ("excellent" model) and the diagonal line is greater than that of the blue curve ("good" model), reflecting a greater predictive power of the yellow model than the blue model. (From Erik Blasch <https://www.researchgate.net/profile/Erik-Blasch>)

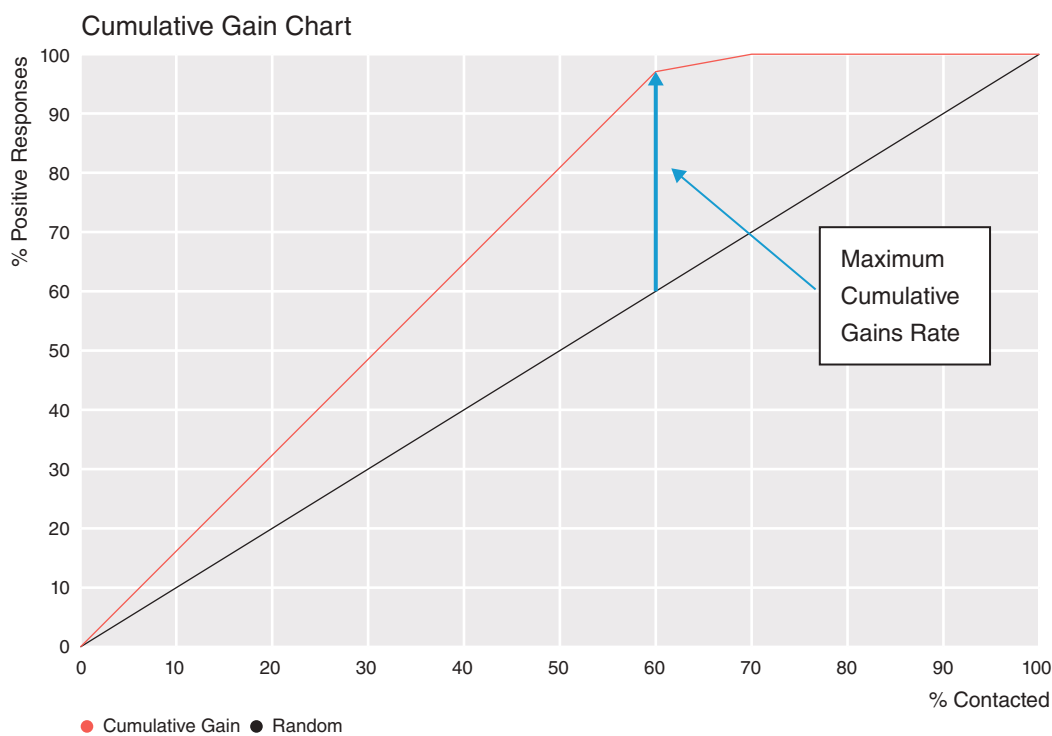


FIG. 15.3 A KNIME cumulative gains curve for a Random Forest model of student attrition (churn). The red line shows the cumulative gain for each decile of the student population. The blue arrow shows the point of maximum instantaneous gain.

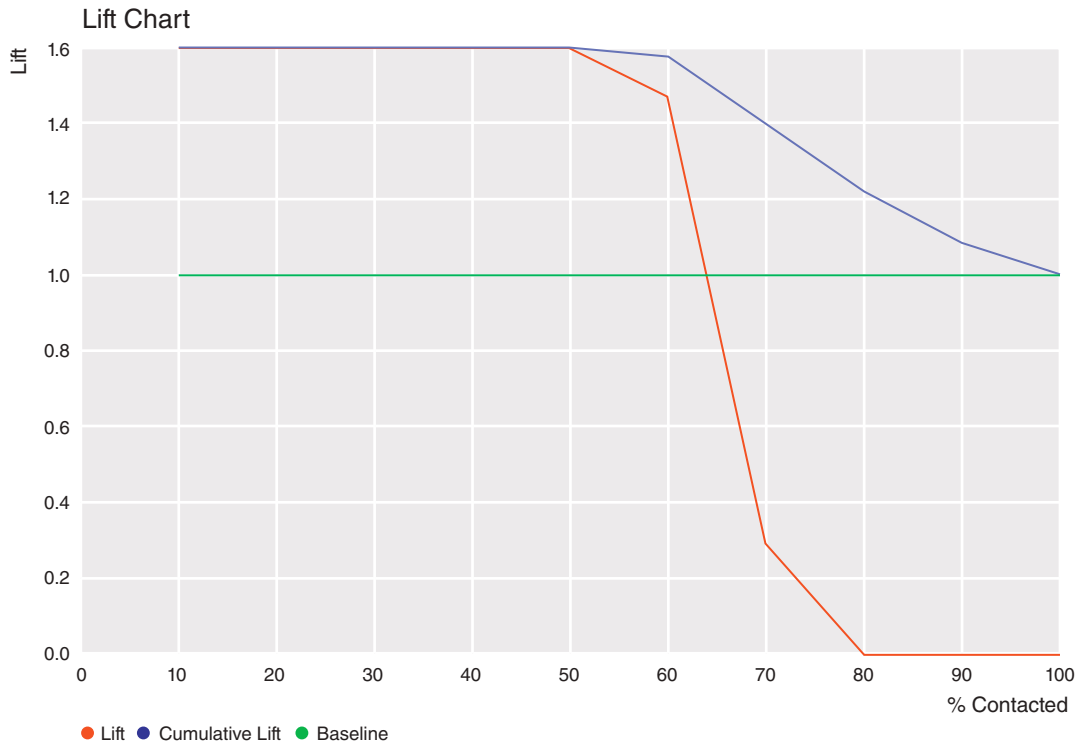


FIG. 15.4 A KNIME incremental lift curve, showing a lift of 1.6 times the random expectation through the first 5 deciles of a student population. This lift value is calculated as the ratio between the number of positive responses and the number of expected responses in each decile. Notice that the incremental lift line (red) crosses the orange line (random expectation) at between the 6th (60%) and 7th decile of the chart. This point of crossing of the line of random expectation indicates that interactions with students beyond that point are not worth the effort (they will leave no matter what interdictions are offered).

If the signal in the data set is faint, the error term will be relatively large. If the signal in the data is strong, the error will be relatively small. Unfortunately, the error term in Eq. (15.1) is a combination of random error and model error. Most model performance metrics do not distinguish between random error and model error. But there are some techniques that can be used to measure model error to some extent and correct for it. We will begin by discussing model performance metrics, which express the total combined error. Later in the chapter, we will present some common techniques for assessing model error, and show some ways to correct it (partially).

Assessment of random error

Many of the measures of error can be used with both numerical prediction and classification problems, but there are some differences in application.

Mean – squared error (MSE)

$$= \frac{\sum_{i=1}^N (\text{predicted}_i - \text{observed}_i)^2}{N} \quad (15.3)$$

Root mean – squared error (RMSE)

$$= \sqrt{\text{Mean-squared error}} \quad (15.4)$$

The MSE expresses how the predicted and expected values differed among all records (cases). The differences are squared to remove the negative signs and retain the numerical quantity. The RMSE transforms the MSE back to the scale of the original data. Use the mean squared error when comparing errors among runs because the differences between values provide more discriminating power. Use the RMSE when you must relate total error back to the scale of the data values which produced it.

Mean absolute error (MAE)

$$= \frac{\sum_{i=1}^N (\text{predicted}_i - \text{observed}_i)}{N} \quad (15.5)$$

The MAE expresses the sum of the positive and negative errors, as a measure of the overall accuracy of the model. However, this measure does *not* express any information on how variable the error was. The error values could have been very large positive and negative values, which nearly happened to balance out. The accuracy in this case would reflect the performance of the model across the scored population as a whole. This measure might be useful to show one dimension of the error, but it should be used in tandem with other measures to assess the predictive power of a model.

Relative squared error (RSE)

$$= \frac{\sum_i^N (\text{predicted}_i - \text{expected}_i)^2}{\sum_i^N (\text{expected value} - \text{expected mean})^2} \quad (15.6)$$

RSE relates the total squared error to the error using a simple predictor (the average of all expected values). This measure expresses the difference between the model and a simple model used as a baseline. RSE shown in Eq. (15.7) is analogous to MAE:

Relative absolute error (RAE)

$$= \frac{\sum_i^N (\text{predicted}_i - \text{expected}_i)}{\sum_i^N (\text{expected value} - \text{expected mean})} \quad (15.7)$$

Simple correlation coefficient. The simple correlation coefficient (also known as the Pearson product-moment correlation coefficient) is a classical statistical parameter which measures the degree of relationship between any two lists of data (e.g., observed and predicted values). A perfect correlation has a value of 1.0 and a value of 0.0 shows no correlation whatsoever. The correlation coefficient may show that two data sources are related, but it doesn't mean that one data source causes another; correlation does not necessarily indicate causality! The correlation coefficients are available in most data mining tools in the form of a matrix, showing values for each pair of variables.

R² value is related to the correlation coefficient between two variables in that it expresses the combined relationship between the target variable and all of the predictor variables. If there are only two variables, *R²* is the square of the correlation coefficient. It is sometimes called the *coefficient of determination* because it gives you some information on the "goodness of fit" of the model. This squaring operation is assumed by statisticians to "convert" correlation into causality. For more than two variables, the combined correlation coefficient (*R*) is squared. The squaring of the correlation coefficient reduces the value of the combined correlation coefficient, according to a long-standing convention in statistical analysis to covert correlation to a causal relationship. Like the correlation coefficient, an *R²* of 1.0 indicates a perfect fit. The *R²* value is a measure of the amount of variation in the data set that can be explained by the combined effect of all of the predictor variables.

Use of the *R²* value as a model evaluation metric is problematical because it changes as more variables are added to the model and because it depends on the specific selection of variables included in the model. To

account for this effect, an adjusted *R²* value is usually calculated and used to show the relationship. The adjusted *R²* value penalizes the value for each variable added to the model, which is a reflection of the additional uncertainty that the added variable is inappropriate (e.g., it might be highly collinear with another variable). Also, there is a built-in bias in the metric against variables left out of the model. If a variable is intentionally omitted from the list of variables (the *specification*) during a feature selection operation, its effect to explain the variation is missing from the model. This fact seems like it should be quite obvious, until it is viewed together with the fact that this bias is *not* necessarily present in the operation of machine learning algorithms. The reason is that the machine learning processing architecture can compensate to some degree for the lack of a variable in the specification by using *surrogate variables* (variables in the specification that have a similar effect to the missing variable). The automatic assessment of variable interactions can identify these surrogate variables, whereas the processing of parametric modeling algorithms cannot.

Kolmogorov-Smirnov (KS) statistic is a measure of the difference between the cumulative fractions of two data sets. For any number *x* in a set of data sorted on *x*, the cumulative fraction is the fraction of the data represented by values smaller than *x*. For example, consider a list of 8 numbers, 1, 3, 6, 8, 9, 20, 21, and 30. The cumulative fraction of value = 11 on the scale of this sorted list is 5/8 or 0.675. Given another list of numbers, 1, 2, 4, 6, 12, 15, 20, 25, the cumulative fraction at value = 8 on that scale is 4/8 or 0.5. The difference between 0.675 of one list (for value = 8) and 0.5 for the other list is a component of the KS statistic.

In effect, a plot of the KS statistic analogous to the area under the curve (AUC) in the ROC curve in Fig. 15.2 and gains curve of Fig. 15.3 reflects how different the two lists are. You can use this measure to compare the predictions of two models, or to compare the observed and expected values of the same model, or to compare a data distribution to a normal distribution. Many data mining tools provide this comparison statistic for your use in evaluating the predictive power of a model by applying the criterion to the observed and predicted values.

KS caveats:

1. KS works only on continuous data sets.
2. KS works well only near the mean of a distribution, and outliers can generate a significant bias.

Assessment of systematic error

Now we can turn our attention to the assessment of systematic error in models. The previous discussion showed techniques for assessing random error.

The most common systematic errors in a modeling project are:

- Improper algorithm use
- Inadequate experimental design
- Sampling errors

Improper algorithm use or inadequate experimental design

The first two sources of systematic error are rather complex in nature. Error caused by improper algorithm use might be minimized by studying [Chapter 13](#). Error caused by inadequate or faulty experimental or data analysis design is very difficult to assess, but it might be quite significant. [Murray et al. \(2008\)](#) studied 72 cancer trial papers indexed in PubMed and MedLine between 2002 and 2006. They found that over half of the studies published were flawed sufficiently in either algorithm choice or experimental design to largely invalidate their conclusions.

[Tinsley et al. \(2016\)](#) provide another example of how errors in experimental design could account for results that were only slightly better than random. They tested the pooled results of the perception of three subjects to a single photon light source. They found that the subjects could distinguish the presence of the light source correctly 51.6% of the time, where 50% represents the random expectation of guessing. The authors published the article containing a strong interpretation that the human eye could distinguish the presence of a single photon. These results were challenged by one of the reviewers, who claimed:

1. There was no discussion of the very weak results.
2. Many other factors that might have influenced the results were not considered in the experimental design:
 - a. Different genetic backgrounds
 - b. Different sleep-wave cycles
 - c. Differences in caffeine intake from tea or coffee prior and during the experiment

In addition, there was no consideration of the possible effects of differential perceptual abilities of subjects with and without contact lenses (one subject had contacts; the others did not). Also, there was no discussion of the assumptions and how they could have been violated. There are a number of assumptions that underlie Fisher's Exact Test used to analyze significant differences.

1. The subjects' behaviors are independent of each other. No consideration was given to possible effects of outside relationships between subjects, which might have caused bias in their response patterns.
2. Subjects were drawn from a population of potential subjects by random sampling.

3. The assumption is that each entry in the analyzed 2×2 table could only be positive or negative, not both (called the Directional Hypothesis). In the [Tinsley et al. \(2016\)](#) study, the pooling of the data for all three subjects violates this assumption.

The reason that different probabilities associated with a different set of outside influences may affect the results of the analysis is that the probabilities are assumed to be additive. If subjects represented different behavioral populations defined on the basis of different sets of influence factors in their backgrounds, adding their probabilities is analogous to adding apples and oranges available to derive a control factor in an apple pie factory.

These shortcomings in the experimental design of [Tinsley et al. \(2016\)](#) could easily invalidate their "weak" findings.

Sampling errors

Sampling errors are, however, much easier to minimize. When a sample data set is drawn from a larger population and used for analysis, the assumption is made that it is representative of the population. The degree to which this assumption is violated may cause a systematic error that might completely invalidate the model results built on the sample data set. Another source of possible sample error may result from the partitioning of modeling data sets into the Training, Testing, and Validation data sets. Sampling error of this type can be assessed by three techniques of drawing samples in slightly different ways: (1) Jackknife, (2) Bootstrap, and (3) K-fold cross-validation. Because multiple samples are drawn, these techniques are called forms of *resampling*.

Resampling/techniques

1. Jackknife
2. Bootstrap
3. K-fold cross-validation

Jackknife

The Jackknife is a method used to estimate the variance and bias of a large population. This was the earliest resampling method, introduced by [Quenouille \(1949\)](#) and named by [Tukey \(1958\)](#). It involves a leave-one-out strategy of estimation of a parameter (e.g., the mean) in a data set of N observations (or records). Ideally, $N-1$ models are built on the data set with different factors left out of each model. The estimates of all models are then aggregated into a single estimate of the parameter. The Jackknife gets computationally intractable as $N \rightarrow \infty$.

The success of the Jackknife in academics and research led to the development of the bootstrap method.

Bootstrap

The bootstrap method divides the data set with N cases into B samples of identical size with replacement. A separate model of some target variable is built on each of the samples, yielding an n number of predictions for each record in the data set. The mean (average) prediction can be calculated and used as the final prediction for each record. There is an art to the specification of the n number. The greater the B number, the closer the outcome will be to the ideal bootstrap ($n = N$ records), but the longer it will take to process the bootstrap. A compromise must be made between a theoretical maximum and a practical optimum. Efron and Tibshirani (1994) have argued that in some instances, as few as 25 bootstrap samples can be large enough to form a reliable estimate of the correct prediction. Many studies have shown that the bootstrap resampling technique provides a more accurate estimate of a parameter than the analysis of any one of the n samples. The bootstrap method is more common than the Jackknife in predictive analytics because it doesn't matter how many records are in the data sets (the N number).

K-fold Cross-validation

Consider the discussion in Chapters 9 and 10 about the partitioning operation prior to modeling. The data set is split (partitioned) into three data sets: the training set, the testing set, and the validation set. The model is trained on the training data set and tested on the testing data set iteratively, until a final model is built. The final model is in turn validated (assessed for accuracy) by using the model to predict the target values in the validation data set. There are several drawbacks to using this approach.

- The validation estimate of the test error can be highly variable, depending on precisely which observations are included in the training set and testing sets and which observations are included in the validation set.
- After the partitioning operations, only some of the records are used to build the model. A better model

might be built using all of the available data, but there would be no data left with which to validate it.

- If the model were to be built on the entire data set, it is probable that the validation set error may tend to overestimate the error of the model built on the entire data set.

Machine learning tools require partitioning of the data for the purposes of error checking. However, it is very likely that the partitioning operation itself will increase the model error of the data.

What can we do? Ah, there is a way out of this conundrum – use the technique called *K-fold cross-validation*.

The K-fold cross-validation method builds a number of models (K) on different complements of the total data set in order to correct (partially) for differences in target signal in different data samples of the total data population. This method is similar to the bootstrap method, except that each of the sampled subsets (folds) is used as a validation data set. The method follows these steps:

1. The entire data set is divided into K subsets (tuples), usually 10.
2. The model is trained on 9 of the 10 tuples by partitioning the data set into training and testing sets, and the 10th tuple is used as a validation data set.
3. Ten models are built following steps 1 and 2 above.
4. The predictions of the 10 models are aggregated to yield the final prediction, using some heuristics.
 - a. A common heuristic for models with continuous targets is to calculate the mean.
 - b. A common heuristic for models with categorical targets follows a voting method, where the majority vote wins.

Fig. 15.5 shows how a five-fold cross-validation modeling system is designed. The blue boxes represent the training samples and the tan boxes represent the testing samples.

Python script for cross-validation

Python has become the open-source tool of choice among many data scientists. KNIME is another open-source tool with a graphical programming interface that includes a Python Scripting node for running Python

Sample	Model 1	Model 2	Model 3	Model 4	Model 5	Legend
1	1 Fold	Fold 2	Fold 3	Fold 4	Fold 5	
2	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Train sets
3	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test set
4	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	
5	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	

FIG. 15.5 A five-fold cross-validation modeling design.

scripts within the KNIME user interface. Table 15.2 shows a simple Python script for performing a cross-validation operation on the commonly used Iris data set.

Notice the range of cross-validation fold accuracy scores in Table 15.2 (0.8 to 1.0, or 80% to 100% accuracy), which reflects the differences in data distribution of the values in the variables among the models of the five folds. The average prediction accuracy among the five models (0.9133 or 91.33%) is the estimated prediction accuracy of the data patterns in the entire data set.

Deployment of cross-validated models

Deploying a cross-validated model is not possible because the samples are drawn without replacement (using Jackknife sampling, not bootstrap sampling). Therefore there is a logical link in the definition of the data subsets between successive cross-validation submodels. A related resampling technique, bagging (discussed below), uses bootstrap sampling and aggregation of results of the submodels; therefore all of the data is available for the definition of each sampled data subset. Bagging can be used in deployment only if the original training set is available for use. In that case, the validation data set is substituted for the testing data set, and the average predicted value for each row among the

submodels is calculated. It is possible, however, to do a cross-validation operation within iterations of a KNIME bagging loop to reduce the sampling error caused by the internal splitting of the input data set into a training set and a testing set (50:50) within the operation of the algorithm. These resampling approaches will be explored more fully for ensembles in Chapter 16.

We might view the bootstrap and K-fold cross-validation techniques as model enhancement methods. In fact, the bootstrap approach is an element in the ADABOOST model enhancement method discussed below.

Model enhancement techniques

There are many model enhancement techniques that are used commonly to refine the predictability of models. In this section, we will consider:

- Meta-modeling
- Boosting
- Bagging
- Parameter adjustment
- Using different sets of predictors

Meta-modeling

A meta-model is an overall model composed of elements, which themselves are models. This organization is like a complex systems model in engineering, which is composed of submodels of a number of specific functions. The outputs from the submodels are combined in a particular sequence to produce the model of the entire system. There are two kinds of meta-models in data mining:

- Ensembles (or “bundles”)
- Complex models

Ensembles

An ensemble is a group of models “bundled” together to predict the same set of values (Fig. 15.6).

TABLE 15.2 A Python script for performing a cross-validation modeling operation on the Iris data set, with output.

```
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import KFold, cross_val_score
X, y = datasets.load_iris(return_X_y=True)
clf = DecisionTreeClassifier(random_state=42)
k_folds = KFold(n_splits=5)
scores = cross_val_score(clf, X, y, cv=k_folds)
print("Cross Validation Scores: ", scores)
print("Average CV Score: ", scores.mean())
print("Number of CV Scores used in Average: ", len(scores))
```

Output:

```
Cross Validation Scores: [1. 1. 0.83333333 0.93333333 0.8]
Average CV Score: 0.9133333333333333
Number of CV Scores used in Average: 5
```

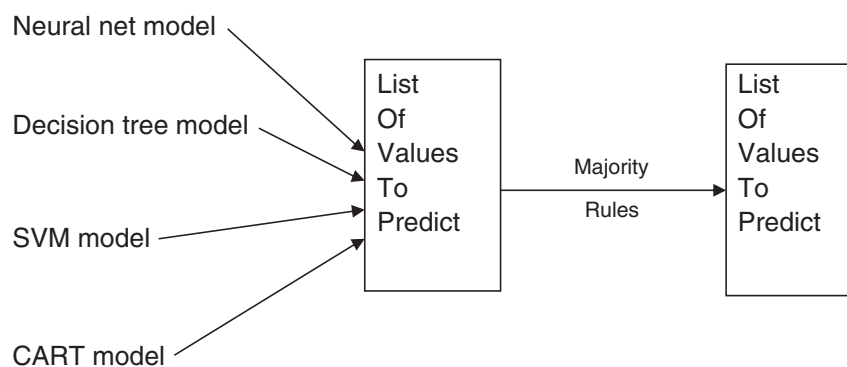


FIG. 15.6 An ensemble of models used to predict an outcome.

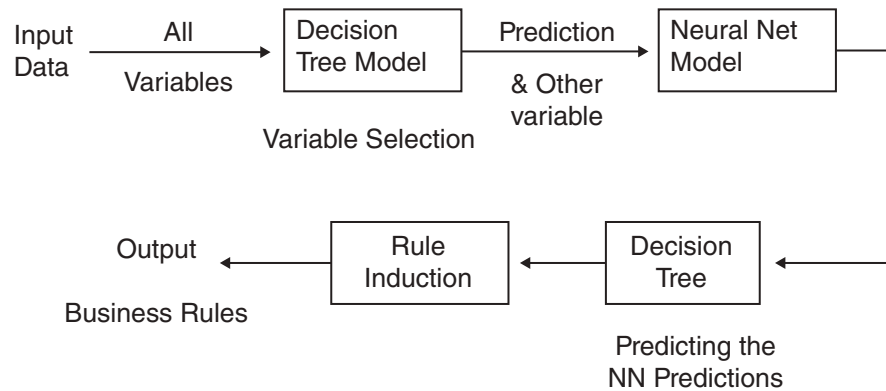


FIG. 15.7 A complex model in which the predictions of one model are used as input to a subsequent model.

If the predicted value is a category, each constituent modeling algorithm contributes a “vote” for a category, and the category with the most votes wins for a given record in the list (majority rule). Some analytical tools provide other heuristics to replace the majority rule. If the predicted value is a number, the mean value for the predictions (or some other heuristic value) is calculated across all model predictions.

Another form of ensemble modeling is performed with the *Boosting* and *Bagging* techniques. These techniques have become very popular in the last 10 years, particularly in combination with other algorithms, such as Boosted Trees.

Complex models

A complex model uses several models in tandem to create a prediction. Fig. 15.7 shows how constituent models are “ganged” together to produce a multistep model.

Boosting

The basic idea behind boosting is to add an outside processing loop around a weak learner to convert into a stronger learner. Between boosting iterations, a parameter like a split-point in a decision tree is adjusted by an amount proportional to the false-negative error rate. Various forms of boosting algorithms focus on false-negative forces, the algorithm on resolving errors in classification models (with categorical targets), and numerical errors in regression models (with continuous targets).

There are many types of boosting algorithms, but the most common are:

- ADABOOST
- Gradient Boosting
- Stochastic Gradient Boosting
- XGBOOST

We will describe briefly only the ADABOOST algorithm for binary classification problems.

ADABOOST

The algorithm’s name is an acronym for “Adaptive Boosting,” which was introduced by Yoav Freund and Robert Schapire (Freund and Schapire, 1997), for which they won the Nobel Prize in 2003. The processing of the algorithm associates equal weights to all variables in the first iteration. After one iteration of the model:

1. The weak classifier trains a model and assigns weights w to each variable i proportional to the probability associated with the classification of each variable in the model. For example, the classifier might calculate the classification probability as 0.3 for a case with an observed value of target = 1. This case would be assigned a binary classification of target = 0. This classification would represent a false-negative outcome (because the classification probability was below 0.5).
2. The weighted sum of the classification rate E is expressed by:

$$E = \Sigma(\text{error}_i * w_i) / \Sigma w_i \quad (15.8)$$

where error_i is the calculated error for each case i and w_i is the weight for assigned by the classifier for each case.

3. The weight of each case is updated proportional to the error of each case if the case was classified incorrectly; otherwise the weight is unchanged.
4. The classifier is run again, through many iterations, until the overall misclassification rate is minimized or some other stopping function is satisfied (e.g., time or number of iterations).
5. This technique serves to focus the algorithm training on maximizing the correct classifications. Note: this is a gross simplification of the processing of the ADABOOST algorithm, but it serves to show the general flow of logic.

Fig. 15.8 shows the logic flow of the ADABOOST algorithm.

The ADABOOST algorithm has been incorporated into various forms of Boosted Trees (Gradient Boosted

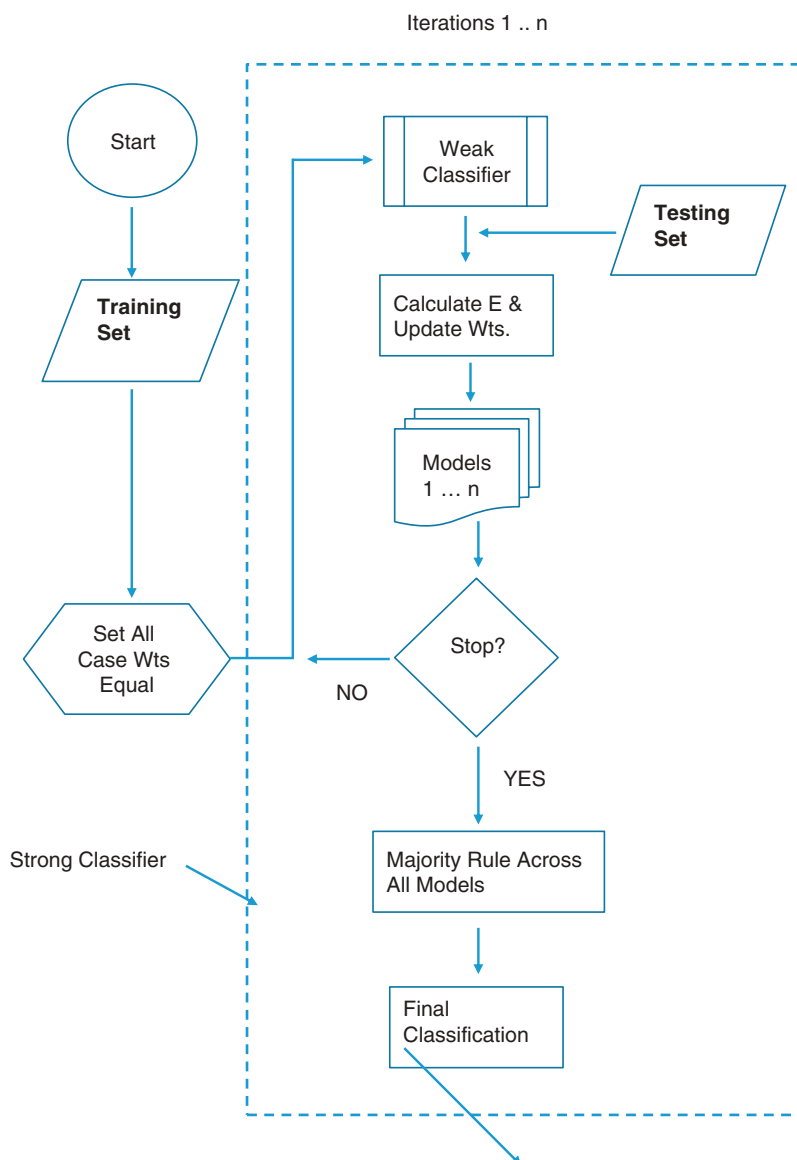


FIG. 15.8 Flowchart of the logic in the ADABOOST algorithm.

Trees and Stochastic Gradient Boosted Trees). These elaborations of the ADABOOST algorithm will not be discussed in detail here. Suffice it to say that these algorithms have been incorporated into many modern analytics tools. It is not uncommon for some form of Boosted Trees to be the most predictive compared to other algorithms used in a study.

Bagging

Bagging is an acronym for “bootstrap aggregation.” A group of samples (folds) are drawn from the entire data distribution with bootstrap sampling (sampling with replacement). The bagging modeling process is similar to cross validation shown in Fig. 15.6, except for the

bootstrap sampling method. But this is a big difference because it generates no logical link between successive samples. Therefore models trained on each bootstrap sample can be aggregated as independent samples of the prediction landscape inherent in the entire data sample. Also, aggregated bagged models can be used in deployment, simply by using the trained model to predict the validation data set. The bagging option will be discussed more fully for ensembles in Chapter 16.

Algorithm parameter adjustment (hyperparameter optimization)

Machine learning algorithms have several to many parameters (called hyperparameters) that control how

the algorithm functions. For decision tree algorithms, common hyperparameters include controls for the number of sequential branches that can be made before processing stops, the learning rate (how fast the error minimization process happens), and what strategy is used to minimize prediction error (e.g., gini score, entropy, etc.). Other hyperparameters of the XGBoosted Trees algorithm, for example, include controls for the type and degree of regularization (variable value transformation to aid in finding the best solution).

Some analytical tools like KNIME provide a special parameter optimization loop for finding the best levels of a set of hyperparameters. The Python sklearn library provides a GridSearch method for semiautomated hyperparameter optimization. Table 15.3 shows a snippet of a Python classification script that includes the GridSearch method.

TABLE 15.3 A Python routine that finds the optimum levels for seven hyperparameters of the XGBoost algorithm.

```
# Import GridSearchCV
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV, StratifiedKFold
kfold = StratifiedKFold(n_splits=5, shuffle=True, random_state=2)
def grid_search(params, random=False):
    kfold = StratifiedKFold(n_splits=5, shuffle=True, random_state=2)
    if random:
        grid = RandomizedSearchCV(classifier, params, cv=kfold, n_iter=20, n_jobs=-1, random_state=2)
    else:
        # Instantiate GridSearchCV as grid_reg
        grid = GridSearchCV(classifier, params, cv=kfold, n_jobs=-1)
    # Fit grid_reg on X_train and y_train
    grid.fit(X_val, y_val)
    # Extract best params
    best_params = grid.best_params_
    # Print best params
    print("Best params:", best_params)
    # Compute best score
    best_score = grid.best_score_
    # Print best score
    print("Best score: {:.5f}".format(best_score))
# End of function definition
grid_search(params={'n_estimators':[100, 200, 400, 800]})
grid_search(params={'learning_rate':[0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5]})
grid_search(params={'max_depth':[2, 3, 5, 6, 8]})
grid_search(params={'gamma':[0, 0.01, 0.1, 0.5, 1, 2]})
grid_search(params={'min_child_weight':[1, 2, 3, 4, 5]})
grid_search(params={'subsample':[0.5, 0.7, 0.8, 0.9, 1]})
grid_search(params={'colsample_bytree':[0.5, 0.7, 0.8, 0.9, 1]})
print("GridSearch finished")
```

Note: The StratifiedKFold method does a stratified-random sampling operation in subsampling, which guarantees that the proportion of rows with negative and positive target values in the Kfold data subsets is the same as in the parent data population.

Miscellaneous model enhancement options

Akaike's Information Criterion is a criterion for comparing the relative predictive information present in a model. One common formulation of the criterion is:

$$AIC = \ln(S_m^2) + 2m/N$$

where m is the number of parameters in the model, S_m^2 is the sum of the squared residuals (observed – expected values), and N is the number of observations.

This criterion can be maximized over various values of m by adjusting one of the algorithm parameters. This operation can help to evaluate a trade-off between the fit of the model and model simplicity (i.e., reduction in complexity)

Adding a cost matrix

Some tools allow you to associate costs with the false-positive and false-negative predictions (e.g., SAS Enterprise Miner). The cost values affect how the algorithm selects a predicted value. Modeling with a cost matrix forces the algorithm to try harder to resolve prediction errors associated with higher costs. The result can be to minimize the effect of error cost.

Using different sets of predictors

Some variables may have very similar predictive effects as other variables; these are called *surrogate* variables). One member of a surrogate pair of variables might generate less error than the other member. Sometimes several variables interacting together can function as a surrogate for another variable. Therefore it is good practice to train several models, each with a different set of predictor variables. These separate models could be combined together into an ensemble, a type of meta-model. Analysis of the results could be used to adjust (optimize) a modeling parameter (e.g., neural net learning rate).

Model enhancement checklist

1. Have you standardized your data?
 - a. Parametric statistical algorithms (e.g., Regression) work better if all variable ranges are similar.
 - b. A regression solution will be biased toward a variable with a relatively large range.
 - c. Machine learning algorithms (e.g., decision trees) don't "care," but some algorithms work better with standardized ranges than without them.
2. Did you remove the outliers?
 - a. For most models, outlier removal will increase the model accuracy.

- b. For some models, you MUST NOT remove outliers. Examples:
 - i. Frauds detection models
 - ii. Intrusion detection models
 - iii. Any model that searches for anomalies rather than common patterns
3. Did you reduce the number of variables?
 - a. This operation almost always increases the model accuracy.
 - b. Some modeling tools can handle a lot of variables (e.g., SVMs), so it doesn't matter if you have a lot of variables. Therefore the option to reduce the number of variables may be appropriate for some algorithms and not for others.
4. Did you consider running the data through an algorithm initially that uses variable selection as part of its logic? (e.g., Step-wise Regression)
 - a. Step-wise Regression will output a model with only those parameters that had significant effect in building the model.
 - b. This can be used as a form of variables selection before training a final model with a machine learning algorithm.
5. Can you segment your data set?
 - a. Often, you can find a variable to use in dividing the entire data set into several parts.
 - i. Your business knowledge may suggest to you that the pattern you are modeling might be significantly different among these data segments.
 1. Rural versus urban phone callers
 - ii. Zip Code ranges, which you know are related to specific socioeconomic segments of the general population
 - iii. Train separate models on each data segment.
6. Have you imputed all of the missing values?
 - a. Check your data variables carefully to make sure that all missing values are filled with appropriate values.
 - b. Machine learning algorithms will delete any case (row) that has even one missing value in one of the variables.
 - c. If there are missing values in your modeling data set, you may find that the size (# of rows) in your final data set is significantly reduced, and that effect can lower the model accuracy greatly.
7. Are there any more ways to combine data variables into higher-level groups?
 - a. This is a form of feature reduction, which will probably increase the accuracy of the final model.
8. Have you decomposed all the categorical variables into dummies?
 9. Try binning some of the continuous variables to reduce the range of the values in them.
10. If you have historical data available, derive lag variables from them, along with other appropriate temporal abstractions (e.g., time-since variables).
11. Have you balanced your data set that has a rare case in the target variable?
12. Have you tried to model your data set with an ensemble?
 - a. After all else is done, the most effective way to increase model accuracy is to use several algorithms to predict the target and then combine the predictions to yield a final prediction.
 - b. Ensemble model predictions will almost always outperform single-model predictions.
 - c. Some tools provide ensemble of algorithms:
 - i. Boosted Trees
 - ii. Random Forests
13. Are there differential costs in the business application for false-negatives versus false-positives? If so:
 - a. Enter the co24sts into a cost matrix if your analytical tool provides one.
 - b. Evaluate the separate accuracy values for target = 1 and target = 0.
14. Have you tried cross-validation to get a better estimate of model errors?
15. Try different groups of predictor variables.
16. Get more data if you can.

Postscript

This chapter discussed some common techniques of model evaluation and some basic techniques of model enhancement. [Chapter 16](#) will continue with the theme of model enhancement by discussing deeper aspects of modeling ensembles and their impact on model complexity.

Key words and phrases

Sensitivity, Specificity, Sampling, Bootstrap, Cross-validation, Boosting, Bagging, Parameter, Global accuracy, Correlation coefficient, Random error, Systematic error, ROC curve, Gains curve, Area under the curve.

References

- Efron, B., & Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC Press.
- Freund, Yoav, & Schapire, Robert E. (1997). "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of Computer and System Sciences*, 55, 119–139.
- Hand, D. J. (2009). Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Machine Learning*, 77, 103–123.

- Murray, D. S., Pais, J., Alfano, Blitstein, C., & Lehman, J. (2008). Design and analysis of group-randomized trials in cancer: a review of current practices. *Journal of the National Cancer Institute*, 100, 483–491.
- Quenouille, M. (1949). Approximation tests of correlation in a time-series. *Journal of the Royal Statistical Society. Series B*, 11, 68–84.
- Tinsley, J., Molodtsov, M., Prevedel, R. et al. (2016). Direct detection of a single photon by humans. *Nature Communications*, 7, 12172. <https://doi.org/10.1038/ncomms12172>.
- Tukey, J. W. (1958). Bias and confidence in not quite large samples (abstract). *Annals of Mathematical Statistics*, 29, 614.

Ensembles and complexity: a practical approach

Preamble

After a preliminary model is built, how do we know that our chosen modeling approach is best? This chapter discussed an elaboration of the modeling operation: an ensemble. A modeling ensemble is a group of models trained by different methods or algorithms, the results of which are combined to produce a set of final predictions. In this chapter, we will show that ensembles can outperform single-algorithm models. But this fact appears to contradict the principle of Occam's razor, which maintains that the simplest solution is often the best. The direct translation of the 14th-century Latin text of Occam's razor is "Don't elaborate the nature of something beyond necessity." We will discuss this apparent paradox and show further that some complexity is necessary, which does not violate the principle of Occam's razor. This chapter is a prelude to [Chapter 23](#), where an ensemble model is combined with boosting and bagging to predict the presence of precancerous colon polyps from clinical data.

Modeling: How to choose the best modeling option?

Building a model is a lot like choosing a president of the United States of America (or any other official). How do you do that? Different people and groups of people have different ideas about who would be the best candidate to put into office. Totalitarian systems install the strongest person into office (or the one with the most military power). This system leads almost always to repressive and unilateral governments from the perspective of one person. Demographic systems form more representative governments by giving all or part of the people a vote for the winner. In the U.S. system, all eligible people have an equal vote, but the "super-voters" (Electors) representing each state in the Electoral College can reverse the popular vote. This foray into civics is an

introduction to the philosophy behind creating models that come as close as possible to representing the significant "voices" in a data set (aspects of the target signal) and selecting the best model to reflect them all. No single model can do it. We must let groups of equal "super-voters" (like the Electoral College) cast their votes and add them up to decide the "winning" predictions for any given case in the data set.

We can follow this philosophy in data analytics by creating an *ensemble* of models, composed of a set of different mathematical algorithms to predict an outcome. These algorithms "look" at the data in slightly different ways, just like the different states of the union view a presidential candidate differently. And ensembles may be less complex in *behavior* than single models; that is, ensembles are more flexible in their adjustment to arbitrary changes in the training data and thus can be more "general" in their ability to perform with new data more accurately.

Ensemble models combine multiple models built with the same or different algorithms to output a prediction composed from outputs from the separate algorithms. [Fig. 15.7](#) in [Chapter 15](#) showed the general design of an ensemble model and included a general description of how an ensemble works. Also in [Chapter 15](#), several methods of model enhancement were discussed (e.g., boosting and bagging). Boosting and bagging are forms of ensemble modeling. This chapter will combine boosting and bagging into an ensemble modeling structure in KNIME to illustrate the combined power of these model enhancement techniques.

The technical challenges of building a single estimator model include:

- High variance in the data set. Variance in a data set results from:
 1. Normal variation in the values of variables across all rows in the data set.
 2. Random selection error. Machine learning algorithms require partitioning input data sets into a training subset and a validation subset. This

partitioning operation involves a random selection process, which injects a source of random error into the model outcome.

- One mathematical “perspective” of the patterns in the data set may not be sufficient to characterize the data patterns in the presence of high data variance.
- One mathematical perspective of present data patterns may not be “general” enough to fit future data patterns well.

After an ensemble is built, predictions from the individual algorithms must be combined somehow. This combination must consider two objectives:

1. Increasing model accuracy.
2. Maintaining or increase model “generality.” Model generality is the ability of the trained model to perform well on new data from a different source or time.

Most often, the combination approach is used to calculate the mean of all predictions for a given example for numerical predictions or the mode for classification predictions. However, sometimes the maximum value is selected where the modeling objective is to optimize the Sensitivity accuracy metric.

Do ensembles violate Occam’s razor?

For most applications, ensemble models will outperform single algorithm models. By their very design, ensemble models are more complex than single-algorithm models. Is this a violation of Occam’s razor? John Elder thinks not (Elder, 2017). Elder views that the concept of complexity in analytical models is measured more by *function* than by *form*. That is, the behavior of the model is characterized by its prediction accuracy as well as its architecture. Occam’s razor recommends that we do not complicate a model beyond “necessity.” If

model accuracy and generality are components of the definition of necessity in the process of model building, ensembles may be necessary to accomplish both of those modeling outcomes.

Elder (2017) maintains that one criticism of ensembles is that interpretation of the model is now more difficult to accomplish than with single-algorithm models. For example, simple decision trees are so easy to interpret that they are very common among machine learning algorithm used despite having relatively low prediction accuracies. Combining decision trees into an ensemble algorithm like Random Forests retains the ease of interpretation and provides outcomes of much higher accuracy. This is why Random Forest has become one of the most popular modeling algorithms in many data domains. The “fly in the ointment” of tree ensemble algorithms like Random Forests and Gradient Boosted Trees is that interpretation of results is available only for the amalgamation of all predictions in the forest.

Elder (2017) cites another criticism of ensembles, which is that the increased complexity of decision tree algorithms is that they may “over-fit” the data. These and other critiques of Occam’s razor are discussed by Domingues (1998).

The power of ensembles

Fig. 16.1 shows the general structure of a stacked ensemble composed of four different modeling algorithms.

Types of ensemble modeling techniques

1. Bagging ensembles
2. Boosting ensembles

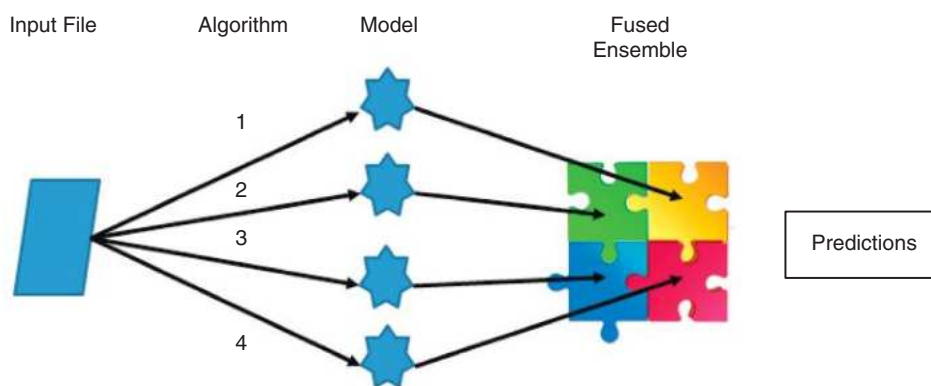


FIG. 16.1 Four different algorithm perspectives contribute to define the model.

3. Stacking ensembles

4. Meta-modeling

Fig. 16.2 shows the KNIME workflow that performs a stacking ensemble operation that includes boosting and bagging ensemble processes. Boosting and bagging operations were described in Chapter 15. They are considered as forms of ensembles because multiple models are trained and the best model is retained.

All of the green-colored boxes shown in Fig. 16.2 are KNIME metanodes. A metanode is a collection of processing nodes represented by a single icon. The function of each of these groups of processing nodes is described in detail in Chapter 23 on Colon Polyp Screening. In this chapter, we will confine our discussion to:

- The description of the general structure of the fused ensemble model.
- Show how the model enhancement processes of boosting and bagging fit into the ensemble design.
- Demonstrate the predictive power of ensembles.
- Show how meta-modeling can open up the “black box” of a machine learning model.

General structure of a fused ensemble model

The Data Input metanode in Fig. 16.2 accesses the Boston Housing data set available in the UC-Irvine Machine Learning Repository. The Initial Filtering metanode performs a correlation filtering operation to remove one member of each pair of variables correlated above the 0.8 (80%) level. The XGB Short-list Builder performs a feature selection to reduce the number of variables to be submitted to the modeling algorithms. The Prep and Filtering metanode performs type change operations and a partitioning operation to output 70% of the input data as the modeling set and 30% as the validation data set for use in validating the model. There are actually three data subsets used to train and validate the models:

1. The modeling data set (70%), which is split internally into two equal parts in the KNIME algorithms to form:
 - a. The training set
 - b. The testing set
2. The validation data (30%) set to be used in model assessment of prediction accuracy.

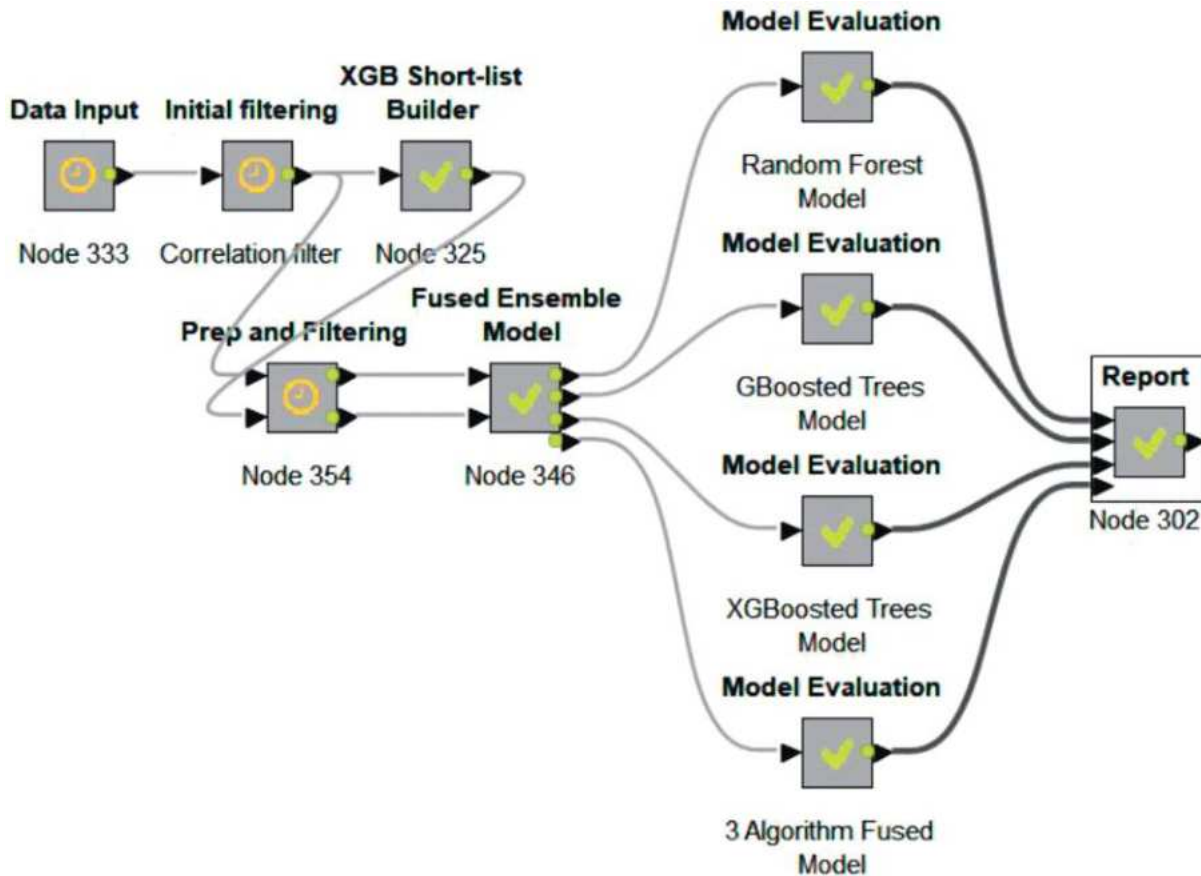


FIG. 16.2 KNIME workflow that includes an ensemble model with boosting and bagging processes.

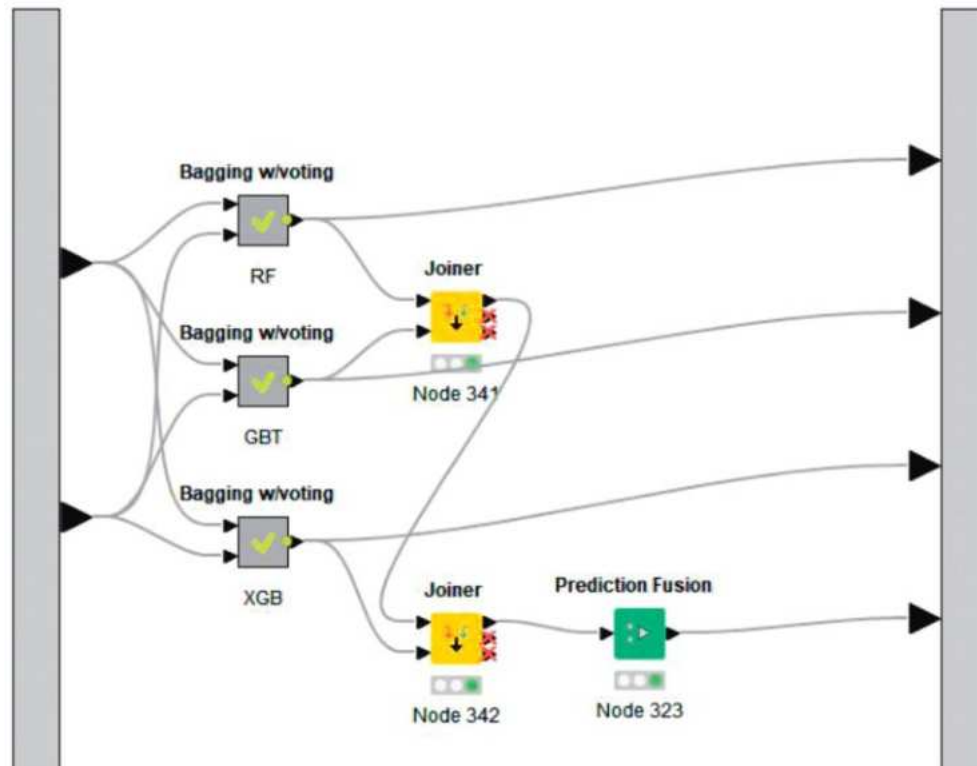


FIG. 16.3 The expansion of the Fused Ensemble Model node.

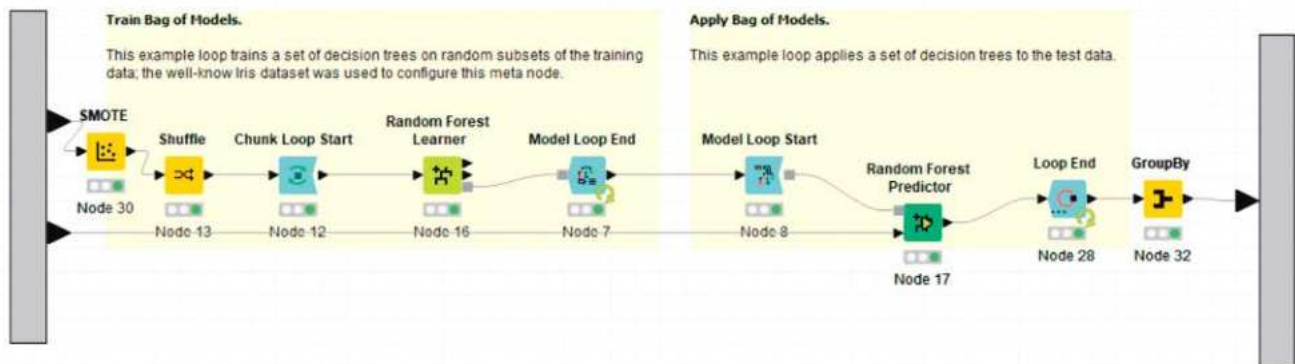


FIG. 16.4 The Random Forest bagging operation in the Fused Ensemble Model metanode shown in Fig. 16.3.

The Boosting operation is performed in the Fused Ensemble Model metanode by the XGBoosted Trees algorithm in Fig. 16.2. The configuration of three algorithm bagging operations is shown in Fig. 16.3.

The Prediction Fusion node in Fig. 16.3 calculates the mean of the predictions of the Random Forest algorithm (RF), the Gradient Boosted Trees algorithm (GBT), and the XGBoosted Trees algorithm (XGB).

The bagging operation for the Random Forest algorithm is shown in Fig. 16.4.

Notice that the bagging operation is performed in a Chunk loop in KNIME. The “chunk” size is set to build models on 10 data subsets, the predictions of which are averaged in the GroupBy node. Similar boosting loops

are contained in metanodes, as shown in Fig. 16.3 for the Gradient Boosted Trees algorithm (GBT) and the XGBoosted Trees algorithm (XGB).

The evaluation results are calculated in the Model Evaluation metanodes for each algorithm and the fusion of all three algorithms is shown in Fig. 16.2. These evaluation metanodes calculate the Sensitivity, Specificity, overall Accuracy, and AUC for each model.

The predictive power of ensembles

The accuracy metric results from the ensemble model in Fig. 16.2 are formatted in the Report metanode and displayed in Table 16.1.

TABLE 16.1 Accuracy metrics of the three modeling algorithms and the fused model.

S Metric	S Value
Bagged Models with voting	
The Random Forest Model	
Sensitivity	0.789
Specificity	0.529
Accuracy	0.682
AUC	0.733
The Gradient Boosted Trees Model	
Sensitivity	0.77
Specificity	0.461
Accuracy	0.643
AUC	0.631
The XG Boosted Trees Model	
Sensitivity	0.748
Specificity	0.494
Accuracy	0.643
AUC	0.677
The 3 Algorithm Fused Model	
Sensitivity	0.807
Specificity	0.506
Accuracy	0.683
AUC	0.772

The predictive accuracy values of three of the four metrics are higher for the fused model results shown in Table 16.1 than for any of the metrics of the three separate models. Only the Specificity accuracy of the fused model (0.506) is slightly less than the value for the Random Forests model (0.529). The area under the curve (AUC) of the fused model is the highest of all models. The metrics for Accuracy and AUC are similar in scope in that they pertain to the overall performance of the model. The difference between them is that Accuracy is calculated from classification confusion matrix values, while AUC is calculated from prediction probability values. Notice that for the fused model, the AUC value of 0.772 is considerably higher than the overall accuracy value of 0.683. Evidently, the set of prediction probability values reflects the true prediction accuracy better than the average accuracy calculated from values in the confusion matrix. AUC can also correct for the effects of this imbalance by

analyzing probabilities (true-positive rate and false-positive rate). Therefore it appears that AUC is a better indication of overall model performance than overall accuracy. These results show the predictive power of an ensemble model over single-algorithm models.

Ensemble modeling with KNIME and Python

KNIME is integrated with Python to permit the processing of Python scripts in a Python Scripting node. Ensemble models can be built in KNIME using Python modeling scripts instead of KNIME modeling algorithms. The Python Scripting node is shown in Fig. 16.5.

Fig. 16.6 shows three Python modeling scripts combined together in KNIME to form an ensemble.

Fig. 16.6 illustrates how KNIME integrates Python into building analytical models. KNIME provides the visual programming environment to generate a visual

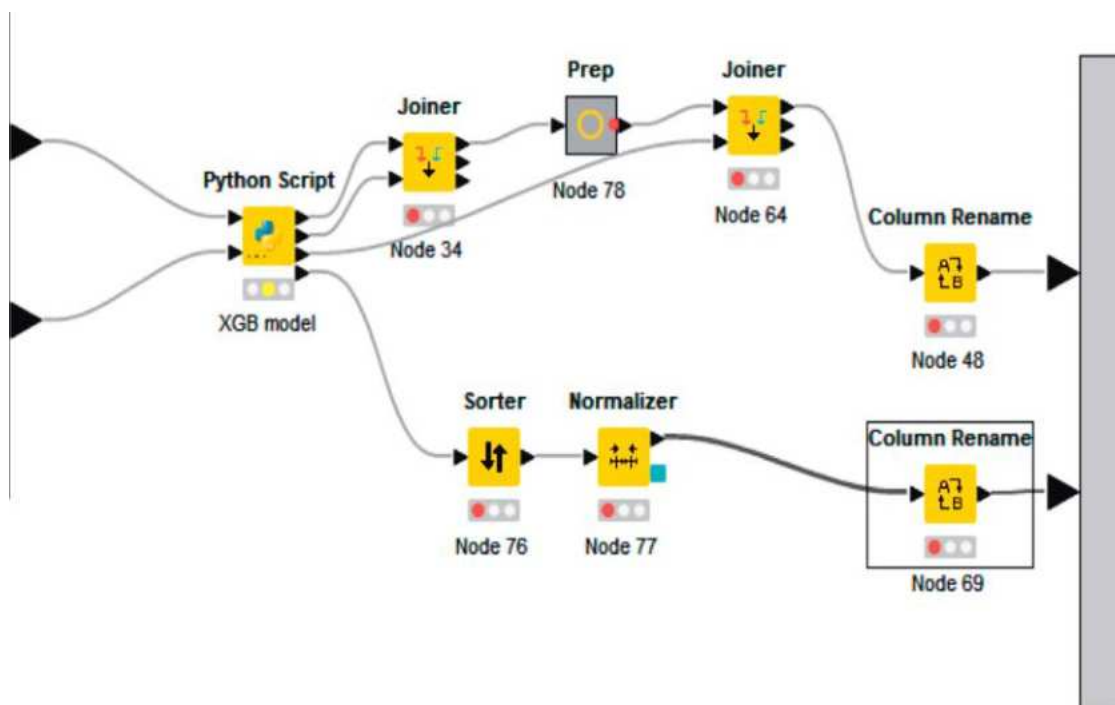


FIG. 16.5 Processing of a Python XGBoosting modeling script in KNIME.

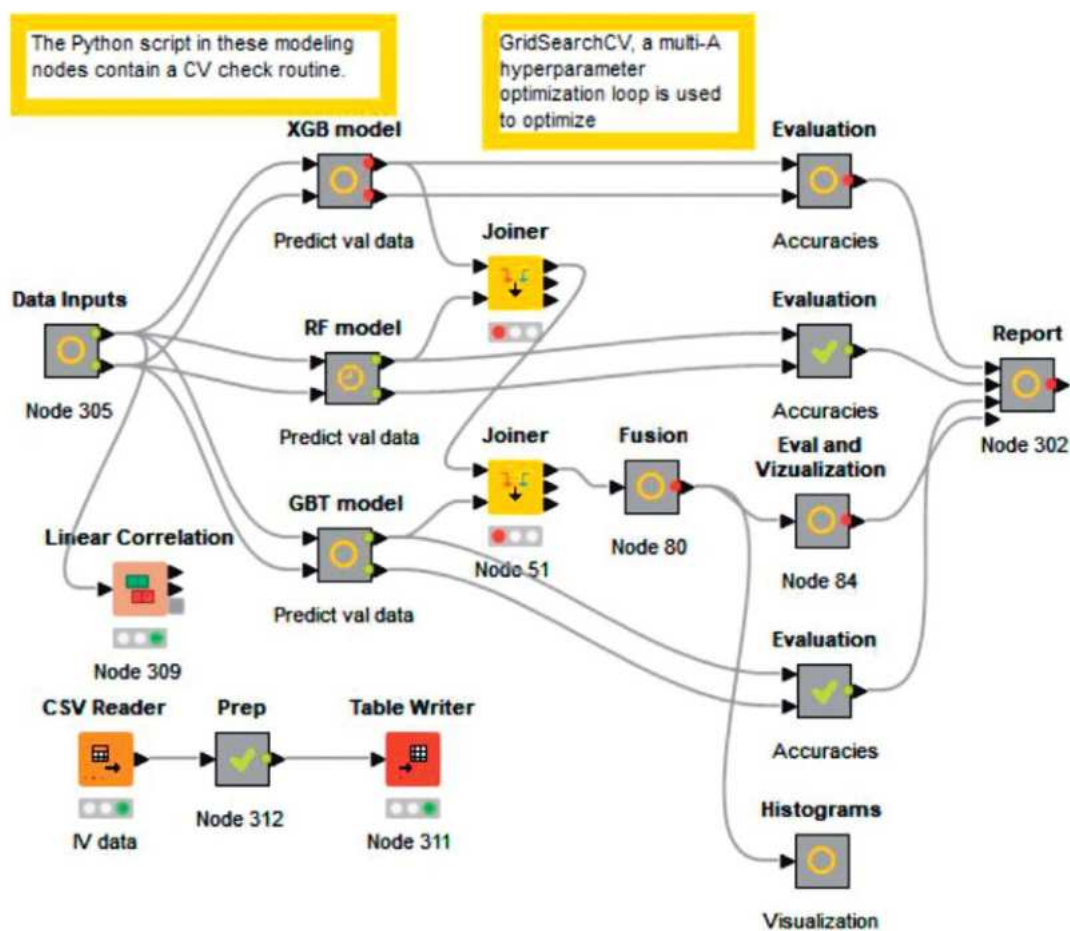


FIG. 16.6 A KNIME ensemble modeling workflow using Python scripts to build models with XGBoost (XGB), Random Forest (RF), and Gradient Boosted Trees (GBT) algorithms and compose them into a fused ensemble model.

TABLE 16.2 Accuracy metrics for three modeling algorithms and their fusion to form an ensemble.

S Metric	S Value
Python Models	
The XGB Model	
Sensitivity	0.761
Specificity	0.568
Accuracy	0.682
AUC	0.716
The Random Forest Model	
Sensitivity	0.723
Specificity	0.571
Accuracy	0.66
AUC	0.697
The Gradient Boosted Trees Model	
Sensitivity	0.73
Specificity	0.568
Accuracy	0.663
AUC	0.717
The 3 Algorithm Fused Model	
Sensitivity	0.748
Specificity	0.568
Accuracy	0.674
AUC	0.717

display of the systems of operations to train the set of models. Python provides the flexibility in configuration and processing superior to similar algorithm implementations in KNIME.

Fig. 16.6 shows that the output ports on the right side of the XGB, RF, and GBT metanodes direct relevant variable values as inputs to the evaluation metanodes, where they are appended to a CSV file for subsequent processing in the Prep metanode in the bottom-left corner of the workflow. The top right output ports of the three modeling metanodes direct target expected and predicted

value to the evaluation metanodes for calculation of prediction accuracy metrics. Outputs of all of the evaluation metanodes are directed as inputs to the Report metanode, which generates the output table shown in Table 16.2.

Meta-modeling – Opening up the machine learning “black box”

A meta-model is a model of model. One form of meta-modeling is to enable explainable AI solutions.

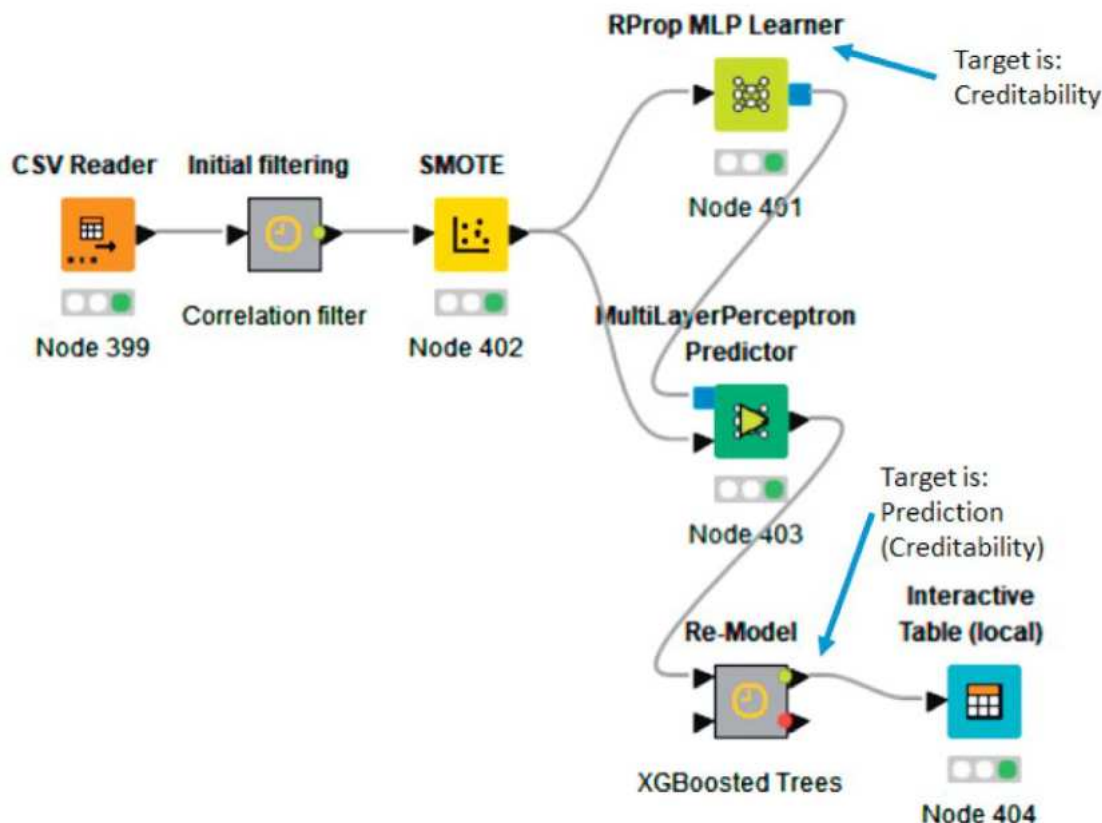


FIG. 16.7 A meta-modeling workflow that uses a decision tree to open the “black box” of a neural network model.

Additional information on this subject is presented and discussed in [Chapter 18](#). In this chapter, we will discuss the use of a decision tree ensemble (XGBoosted Trees) to identify the important variables in a neural network model.

[Fig. 16.7](#) shows a KNIME meta-model workflow that uses a decision tree algorithm (XGBoosted Trees) to predict the predictions of a neural network algorithm (RProp MLP) for the German Credit Scoring data set, available in the UC-Irvine Machine Learning Repository.

Predictor variables

The problem with many machine learning algorithms (like the MLP neural network algorithm in KNIME) is that they don’t provide any output of predictor variable importance. This lack is a serious limitation to their use in applications where the predictor variables must be incorporated into business interdiction strategies to address problems like customer attrition (see [Chapter 21](#) on customer response modeling). This lack of interpretability can be removed by using a second modeling algorithm (with important variable reporting) to predict the

predictions of the first modeling algorithm (without important variable reporting). The output of the first model (a neural network algorithm – a multilayer perceptron) shown in [Fig. 16.3](#) is used as the target variable for the second model (XGBoosted Trees) shown in [Table 16.3](#). As such, the second model is a model of a model, or a meta-model. The list of the top 10 important features (variables) in the XGBoosted Trees model is shown in [Table 16.2](#).

The Gain column is the normalized average information gain across all splits in the decision tree that use a given feature. Higher Gain values reflect the higher importance of a given feature for generating a prediction. This capability of the meta-model is very useful in conjunction with many modeling algorithms that might otherwise be rejected for use in a given business application.

Postscript

The premise of this chapter was the statement that an ensemble is an elaboration of a simple modeling operation. We showed that this elaboration helps to combine

TABLE 16.3 The top 10 predictor variables for the German Credit Scoring data set.

Row ID	S Feature Name	D Gain
Row0	>=200 DM_Balance	1
Row1	no running account_Balance	0.217
Row2	no savings_Savings_stocks	0.181
Row3	guarantor_Debtors_Guarantors	0.176
Row4	<= 200 DM_Balance	0.165
Row5	paid back previous credits at this bank_Payment	0.161
Row6	less than 20_Installment_%_available_income	0.132
Row7	new car_Credit_purpose	0.099
Row8	no balance_Balance	0.097
Row9	other_Credit_purpose	0.095

outputs from different modeling approaches to produce the optimum outcome. Chapters 17 to 19 discuss several other elaborations of modeling: elaborated neural networks (using “deep” learning), opening the “black box” further (Explainable AI), and the need to combine perceptions of human brains with those of machine learning algorithms (Human-in-the-loop).

Keywords and phrases

Ensemble, Accuracy, Boosting, Bagging, Enhancement, Python, meta-model, Processing loop, XGboosted trees, Random Forest, Gradient boosted trees, Predictor variables.

References

- Domingues, P. (1998). *Occam's two razors: The sharp and the blunt*. *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining* (pp. 37–43). New York: AAAI Press.
- Elder, J., & Peterson, A. (2017). Chapter 16. The apparent paradox of complexity in ensemble modeling. In R. Nisbet, G. Miner, & K. Yale (Eds.), *Handbook of statistical analysis and data mining applications* (2nd ed., pp. 705–718). San Diego, CA: Academic Press (Elsevier).

Deep learning versus traditional machine learning

Preamble

With the dominance of deep learning and Large Language Models (LLMs) in public discourse, it may seem surprising that they are not discussed at great length in this book. This issue goes beyond the book's structure and get to the heart of the difference between deep learning and traditional machine learning. They increasingly represent two different world views with fundamentally different approaches to problem conceptualization, data preparation, and modeling. Many of the chapters of this book would have to be written twice, with a version for each approach. In the foreseeable future, AI and traditional machine learning will likely split into two disciplines, and the early indications of that split are already visible.

Introduction

This chapter will discuss:

- The differences between deep learning and traditional machine learning
- How the history of deep learning led to those differences
- How those differences affect the model building lifecycle

Differences between deep learning and traditional machine learning

Deep learning has historical roots that go back several decades. Once the idea of a hidden layer took root, experimentation with more than one hidden layer followed. As early as the 1990s, Data Analytics tools such as Clementine from Integral Solutions, Limited (ISL) in the UK (later acquired by SPSS Inc. and then IBM) and SAS Enterprise Miner modeling algorithms included

neural network implementations that permitted the configuration of more than one hidden layer. The number of hidden layers was severely limited (by today's standards) both for scalability and because researchers did not anticipate that more than two or three would be helpful. Fear of producing overfit models was also a consideration. Another problem was that these algorithms used the original design used in the Multilayer Perceptron (see [Chapter 13](#)). The other major problem with working with neural network algorithms with multiple hidden layers was that processing time increased exponentially as the number of hidden layers increased.

The entire landscape of neural network algorithm design changed with the article published by [Krizhevsky et al. \(2012\)](#). The article described how the team won the 2012 ImageNet competition with the AlexNet. Sutskever, who has since become a high-profile leader at OpenAI, frequently describes in interviews and conference presentations what made AlexNet revolutionary. In short, it is scale; specifically, more data, dramatic increases in computing power in the form of GPUs, and more hidden layers ([OpenAI, 2022](#)). Notably, the authors also innovated using a preprocessing routine that used a convolution to provide spatial analytical capability. [Chapter 13](#) describes what a convolution is and what it does. The result was to train a model to recognize images in terms of spatial data relationships, not just overall relationships. This technique has completely revolutionized computer vision. Within several years, this “deep learning” architecture, in which “deep” simply means a large number of hidden layers, became the “rage” in the popular press, appearing on the front page of the New York Times. (<http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html>).

In *Deep Learning: A Critical Appraisal*, Gary Marcus argued cogently that within a few years of the 2012 ImageNet results, it was clear that it represented a sea change. Deep learning was not “just another algorithm”—it represented something new and different.

TABLE 17.1. Differences between traditional machine learning and deep learning applications.

	Traditional ML	Deep Learning
Data structure	Mostly structured data in rows and columns	Mostly unstructured: image, video, text
Sample size	It can be low thousands of cases but applicable to very large	Must be very large to extremely large
Data	Typically, data generated through the normal course of business	Often crafted specifically for a particular modeling task
Labels and annotation	Typically, labels (outcomes) are known in the historical data	Annotation must be added or supplemented manually (this process is discussed in Chapter 19)
Data exploration	Central to the process as it prepares for data preparation	Deemphasized relative to traditional ML
Feature engineering	Feature engineering is central to data preparation	Feature engineering is minimized in favor of architecture engineering
Modeling	Multiple algorithms and algorithmic settings are used in an experiment to find the optimum one	Only one algorithm is considered although multiple model architectures are considered
Model transparency	Sometimes functions as a black box	Always functions as a black box
Common use-cases	Loan default, marketing response, fraud, predictive maintenance	Translation, autonomous vehicles, speech recognition, image recognition

Although deep learning has historical roots going back decades, neither the term “deep learning” nor the approach was popular just over five years ago, when the field was reignited by papers such as Krizhevsky, Sutskever, and Hinton’s now classic 2012 ([Krizhevsky et al., 2012](#)) deep net model of ImageNet. ([Marcus, 2018](#))

The dramatic change was, in part, because of increases in computational power and data, the first time that deep learning truly became practical.

Although Marcus is often cited as a skeptic of deep learning, he could also be interpreted as issuing a caution of pursuing deep learning at the cost of abandoning traditional machine learning research. Because the application areas of the two approaches differ significantly, it would seem reasonable to encourage them to continue to work in parallel. We can draw two key points from Gary Marcus’s assessment:

1. While connected to a history that goes back to the mid-20th century, the events in 2012 were almost immediately recognized as milestones.
2. The increases in computational power were almost certainly a necessary development to allow multiple layers to be effective.

But do active deep learning practitioners also recognize the difference in style between the two approaches? The answer is yes, as clearly indicated by [Krohn et al. \(2019\)](#) in a book written for practitioners, in which the authors state:

1. Feature engineering often predominates in the implementation of traditional machine learning algorithms.

2. Deep learning algorithms do not usually involve much feature engineering, while the majority of modeling time is spent on algorithm configuration. This overview of the distinction is from the point of view of a book for deep learning practitioners.

These and other differences are summarized in [Table 17.1](#).

With these differences in mind, we can explore the path that led to contemporary deep learning that results in these differences.

The journey to deep learning

Virtually all histories of neural networks begin, quite appropriately, with McCullough and Pitts ([1943](#)). It is undoubtedly true, and we’ve explored that history in this book. However, if you start with the contemporary advances in deep learning and work backward in time, a slightly different timeline emerges.

The beginning of this story begins with McCullough and Pitts ([1943](#)), who focused on the physiology of the neuron and the analogous structure of perceptions. This was followed by the failure of perceptions, identified by Minsky and Papert (1969, 1972), which ultimately led to the development of the multilayer perceptron. But that narrative can get quite complex with failed attempts and the discussion of the so-called “AI Winters.” Even the moderate success of the multilayer perceptron failed to be as predictive as practitioners expected. With the success of deep learning, however, it relegated the original architecture of the multilayer perceptron to become viewed only as a “shallow” neural net ([Aggarwal, 2023](#)).

The alternate version of the timeline, inspired by the perspective of Krohn et al. (2019) in the book *Deep Learning illustrated*, focuses not on the physiology but rather on the *behavior* of neurons in the brain. Hubel and Wiesel won the Nobel Prize in 1981 for their work begun by Hubel and Wiesel (1959) that uncovered the nature of edge detection in the visual cortex in cats. This early work led Fukushima (1979 in Japanese) to propose his concept of neocognition. This work was published in English by Fukushima (1980). Neocognition employs a hierarchical, multilayered artificial neural network to recognize Japanese handwritten characters. This approach served as the inspiration for the design of convolutional neural networks, a deep learning technique used later for pattern recognition of image data. LeCun et al. (1998) used the same edge detection approach to read handwritten English text characters.

The connection of deep learning to the work of Hubel and Wiesel is not as frequently discussed, but the connection was not lost on those investigators responsible for the contemporary success of deep learning. Even though deep learning was not yet in view, the significance of what Hubel and Wiesel did was not lost on the Nobel Prize Committee either. Here is what Hubel and Wiesel found that led to their winning of the Nobel Prize in 1981.

Hubel and Wiesel discovered that the neurons that receive visual input from the eye are in general most responsive to simple, straight edges. Fittingly then, they named these cells simple neurons. Hubel and Wiesel determined that a given simple neuron responds optimally to an edge at a particular, specific orientation. A large group of simple neurons, with each specialized to detect a particular edge orientation, together is able to represent all 360 degrees of orientation. These edge-orientation detecting simple cells then pass along information to a large number of so-called complex neurons (Krohn, 2019).

Fig. 17.1 shows the timeline of a series of developments that led to the winning of the Turing Prize by Hinton, LeCun, and Bengio, whose work is documented below.

Fig. 17.1 is based on the rendition of the history of deep learning by Krohn et al. (2019). This rendition begins with the work of Hubel and Wiesel and goes on to explain another piece of largely unheralded work by the Japanese electrical engineer Kunihiro Fukushima who “referred to Hubel and Wiesel’s work explicitly.” Fukushima tried to produce layers of abstraction in artificial neurons in imitation of those in the brain (Fukushima, 1979, 1980). This work had a direct influence on the development of the LeNet-5 model by LeCun et al. (1998). In the interim between 1980 and 1998, Rumelhart et al. (1986) developed the very influential backpropagation algorithm, which is used to update the factor weights between successive iterations of a multi-layer perceptron. This updating process is discussed in summary form in Chapter 13.

While most of the machine learning community was focused on traditional machine learning, Rumelhart and his colleagues and LeCun and his colleagues continued to work on what we now call deep learning. The critical turning point was the development of the ImageNet database by Fei-Fei Li’s group at Stanford University (see Dong et al., 2009). Beginning in 2010, the ImageNet team held an image classification competition. Competitors had to correctly classify images and detect different objects and scenes across a *trimmed* list of 1000 ImageNet categories. In 2012, the team working with Hinton won the competition, and dramatically so, with AlexNet (see Krizhevsky et al., 2012). Hinton was Krizhevsky’s PhD advisor at the University of Toronto. Hinton took the position of the last listed coauthor on the Krizhevsky et al. 2012 paper, but his seminal influence was obvious. In particular, over the decades that Hinton worked on neural networks, he was also credited with helping establish backpropagation as the primary technique for adjusting the weights in neural networks (ACM, 2018).

Upon this foundation, deep learning began the process of moving from the academic realm to the broader

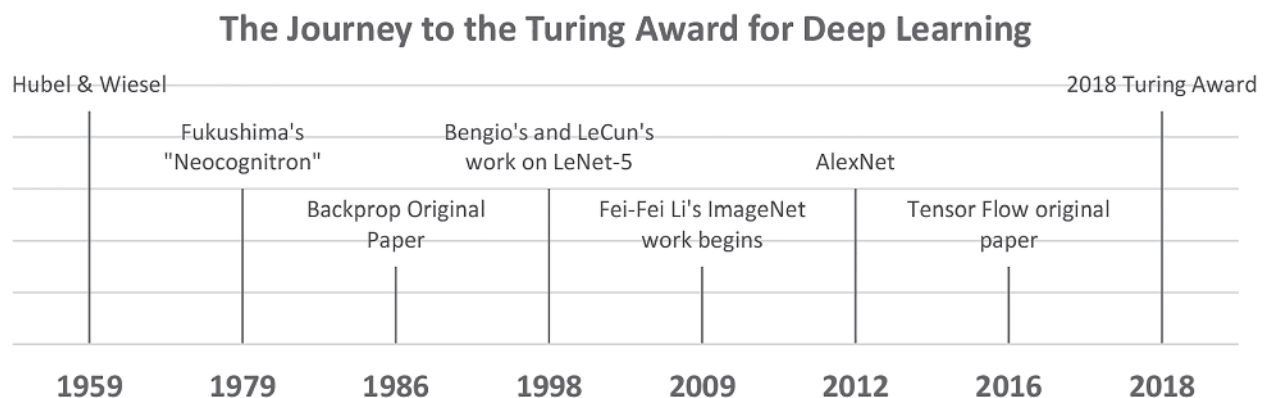


Fig. 17.1 The timeline of development of deep learning that led to the Turing Award in 2018.

data science community with the popularization of open-source libraries to do the work, notably with Tensorflow in 2016 (see [Abadi et al., 2016](#)). In 2018, Hinton, LeCun, and Bengio shared the prestigious Turing Award.

When announcing the award, ACM described how their collaboration led to their sharing the prize:

Hinton, LeCun, and Bengio recognized the importance of building deep networks using many layers—hence the term “deep learning.” The conceptual foundations and engineering advances laid by LeCun, Bengio, and Hinton over 30 years were significantly advanced by the prevalence of powerful graphics processing unit (GPU) computers, as well as access to massive datasets. In recent years, these and other factors have led to leap-frog advances in technologies such as computer vision, speech recognition, and machine translation. Hinton, LeCun, and Bengio have worked together and independently. For example, LeCun performed post-doctoral work under Hinton’s supervision, and LeCun and Bengio worked together at Bell Labs beginning in the early 1990s. Even while not working together, there is a synergy and interconnectedness in their work, and they have greatly influenced each other (ACM, 2018).

Since 2018, the three have made numerous appearances including the popular press, with Hinton often being described in the media as the “godfather” of AI. Bengio is an academic in Canada at the University of Montreal. Hinton recently retired from both academia and Google. LeCun is an academic at NYU and an executive at Facebook.

Why is feature engineering so important

A common thread of the use-cases famously associated with deep learning, computer vision, speech recognition, and machine translation, is that they all involve unstructured data. Silicon Valley-based research start-ups in industries like autonomous vehicles, natural language processing, and image detection overwhelmingly have adopted deep learning. It is less clear whether deep learning offers similar advances in modeling structured data. Even if it eventually does prove more effective in structured data, the necessary loss of transparency is a tradeoff that not all organizations will be willing to make. The use of traditional machine learning and statistically based algorithms is still dominant in use-cases involving structured data in established industries.

Before explaining how traditional machine learning and deep learning address the issue of feature engineering differently, its importance should be clarified. Arguably, the most straightforward gateway into the topic is to discuss an extremely simple multiple linear regression model with just two independent variables. The advantage of a very simple model is that it allows a

focus on the concept of interactions without addressing a complex algorithm.

The risk is that it may seem that by using an algorithm as simple as regression, we aren’t in the category of machine learning at all. As Galit Shmueli has described in her article *To Explain or to Predict?* ([Shmueli, 2010](#)), academics and data miners use techniques like regression in different ways. Academics to test hypotheses, and data miners (data analytics) to search for patterns that can be incorporated into predictive models. This discussion, and indeed this entire book, will focus on the latter. The algorithm chosen is not as material as the goal of the analysis. Note that since this approach is a search for previously unknown patterns, the importance of variables is not known in advance.

In this context, the subject of feature engineering focused on what are called interactions and interaction terms in regression. In the opening section of what is the classic text on the subject, [Aiken and West \(1991\)](#) provide an example that illustrates how easily a regression analysis can go astray.

Consider the following example. A researcher wishes to investigate the effects of life stress (X) and the amount of social support (Z) on physical illness (Y). The researcher highly reliable continuous measures of each ...

$$Y(\text{hat}) = B1 * X + B2 * Z + B0 \quad (17.1)$$

But is this the regression model that the researcher really wished to test?

Many data analysts might be tempted to answer “yes,” but [Aiken and West \(1991\)](#) conclude that the researcher almost certainly is hypothesizing that life stress is positively correlated with physical illness but that social support mitigates it.

Indeed, for individuals with high levels of social support, the researcher that there would be very little, if any, relationship between stress and physical health. This hypothesis implies that there should be an interaction of stress and social support in predicting physical illness, a relationship represented by this equation (Eq. 17.2).

$$Y(\text{hat}) = B1 * X + B2 * Z + B3 * X * Z + B0 \quad (17.2)$$

Therefore the (original) equation (Eq. 17.1) does not test the researcher’s hypothesis ([Aiken and West, 1991](#)).

One could easily, but falsely, conclude that this is only an academic concern. After all, in machine learning, we are rarely testing hypotheses, but that conclusion would be a mistake. This example points at the heart of the importance of feature engineering and while the details may occasionally differ, feature engineering is just as fundamental to traditional machine learning as it is to the

research example. The famous educator and entrepreneur Andrew Ng has the following to say about the subject:

Coming up with features is difficult, time-consuming, and requires expert knowledge. “Applied machine learning” is basically feature engineering (Ng, 2013).

While some of the “expert knowledge” is the practical experience of the analyst, a large part comes from subject matter experts (SMEs) interacting with the analyst. The combination of this expert knowledge results in additional variables being added to the dataset. Thus the discussions of data preparation operations in [Chapters 8–12](#) can be summarized into two general tasks:

1. Transforming the initial data set into a form that can be analyzed at all, and
2. Clarifying the available information by generating new variables to analyze

And as Tom Khabaza, coauthor of CRISP-DM and author of the *Nine Laws of Data Mining* puts it, feature engineering is also about “making the data more informative with respect to the business problem.” (Khabaza, 2010). Data preparation goes beyond data cleaning and the formatting necessary to meet the requirements of the analytical tool or coding library. It also requires using subject matter expertise to help the algorithm solve the specific business problem at hand.

Interactions

Interactions are the effects caused by two or more variables working together to produce a combined effect

that is different or greater than the separate effects of the components.

There are three approaches for including interaction effects in models:

1. Traditional ML approach: use trial and error combined with subject matter expertise to identify, create, and fine tune features
2. Auto ML approach: create all possible interaction terms and use various methods to keep some and discard others
3. Neural net approach: avoid interaction terms and allow the hidden layer to mimic them in the model

Auto ML does not hypothesize interaction terms and then test them in the model; rather, all conceivable interaction terms are generated, followed by a weeding-out process. We will discuss how neural networks (and by extension, deep learning) address the problem in the next section after some reinforcing examples.

In [Fig. 17.2](#), the college PSAT scores are plotted against the level of difficulty of classes taken (Average_Level) for two learning style groups, indicated by the blue and green lines.

There are two critical observations to make:

1. The gap between the lines is uniform; therefore the advantage of the blue learning style group is constant.
2. The slopes of the two lines are constant, so the increased score associated with the difficulty of classes taken is also uniform.

You could conclude that these two statements are essentially equivalent. However, the two statements can be thought of as referring to the two independent variables.

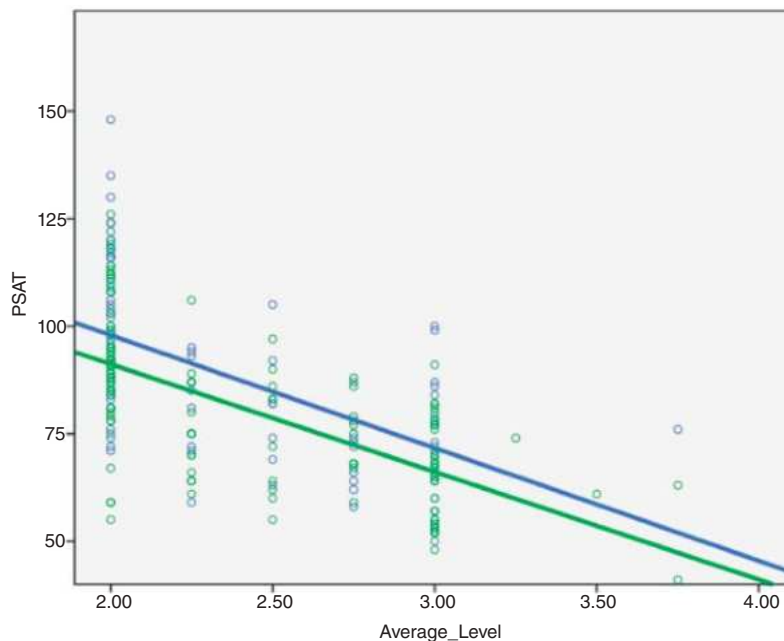


Fig. 17.2. Relationship of PSAT and average level of difficulty of coursework with trend lines for gender.

In this example, it would be OK to use the following regression formula. The third term in Eq. (17.3) is required because no interaction is present.

PSAT (predicted) = B1 * Average_Level

+ B2 * Learning Style + B0

(17.3)

Contrast that with the following plot shown in Fig. 17.3.

Beginning Salary (Adjusted) is the dependent, and Education Level is the first independent variable. The second independent variable is sex, with the steeper line belonging to the male employees and the more gently sloped line belonging to the female employees. The average salary is zero because that variable has been “centered,” a technique that is often recommended when creating interaction terms. (See Aiken and West, 1991, for a more detailed discussion of centering).

The difference in slope is important for two reasons. First, it is arguably the most visually striking characteristic

of the scatter plot. Second, it represents a potentially different kind of bias, and an important one. Women at the bank clearly have lower salaries overall, but the gap also widens as the education level increases. An interaction is present and a third term is required (much like the earlier example that required a third term). If one attempted the regression model with only two terms, it would force the model to have a uniform sex gap for all levels of education. Clearly, that is not the case, so the model would be flawed.

Consider the following regression results in Table 17.2.

With only two regression coefficients, this regression result would indicate that the sex gap in salary is always \$11,342.97. The scatterplot clearly indicates that this is not the case. Consider this alternative with a third regression term in Table 17.3.

First, note that the interaction term, ed_sex, which is Sex of Employee * Education, is significant, because the Sig. value in the far right column for that variable is below 0.05. The other coefficients are somewhat changed, and the third

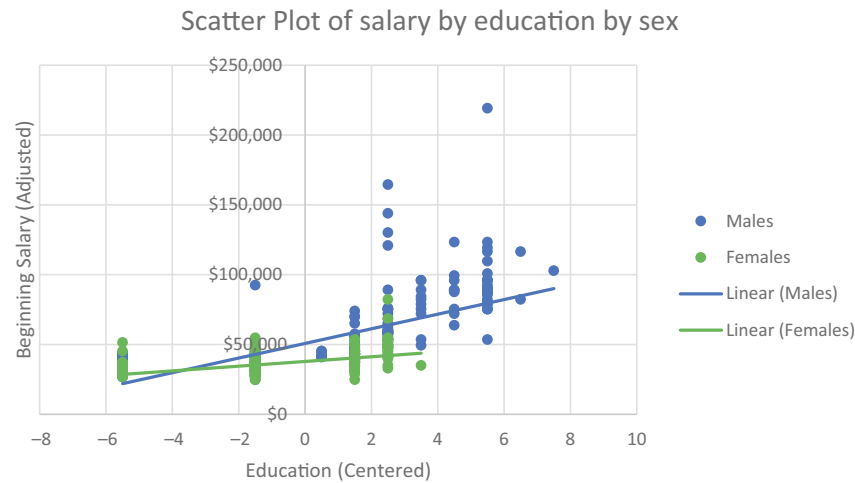


Fig. 17.3. Relationship of beginning salary and education level by sex for PSAT data.

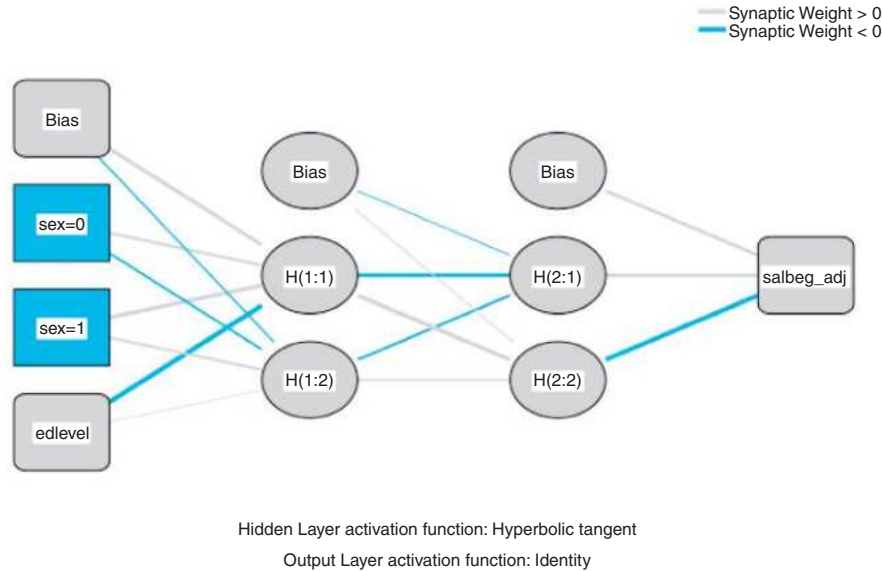
TABLE 17.2. Of bank salary data Regression results for the relationship between beginning salary and education level by sex.

Coefficients ^a						
Model		Unstandardized B	Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
1	(Constant)	51848.613	1017.984		50.933	<.001
	educ_cen	4060.501	271.212	.542	14.972	<.001
	Sex of Employee	-11342.972	1566.814	-.262	-7.240	<.001

^aDependent Variable: salbeg_adj

TABLE 17.3. Regression results for the relationship between beginning salary and education level by sex with an interaction term added.

		Coefficients ^a				
Model		Unstandardized B	Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.
1	(Constant)	50754.037	991.682		51.180	<.001
	Sex of Employee	-12911.659	1523.331	-.298	-8.476	<.001
	educ_cen	5237.170	318.289	.699	16.454	<.001
	ed_sex	-3551.412	552.962	-.271	-6.423	<.001

^aDependent Variable: salbeg_inflation**Fig. 17.4.** The architecture of a shallow neural network of bank salary data with only two “hidden” layers.

term is adjusting for the difference in slope between the lines. So it is no longer the case that the model is forced to be inconsistent with the scatterplot. This is also reflected in an improved model. The flawed model has an R^2 of 46.3% of the variance explained. The second model has a substantially better R^2 of 50.7% of the variance explained.

Interaction representation in artificial neural networks

So how does an artificial neural network address this problem? In order to answer that question, we will use the same salary bias example. Also, a deep learning model will make the example far too complex, so we will use a shallow neural network with only two layers, shown in Fig. 17.4.

The strength of the relationships (analogous to regression weights) among education level, sex, and the dependent variable is indicated by the thickness of the lines connecting the input variables with the hidden

layers. No single line indicates the interactions, which is what makes their manifestation in the model opaque. Note that several paths can be traced between education level and the dependent. Some of those paths include positive relationships (shown in blue), some include negative relationships (shown in gray), and some have a mixture of positive and negative weights. Answering a question as simple as “Is education level positively or negatively correlated with salary?” is challenging. So hidden layers may be capable of modeling interactions without the explicit addition of interaction terms, but it comes at a cost—the cost that the nature of those interactions is almost impossible to determine without additional analysis. In the next chapter, [Chapter 18](#), we will discuss one option for this—Explainable AI (XAI).

In summary, in traditional machine learning, feature engineering is critical to assist the algorithm in finding patterns relevant to the business problem. However, its shortcoming is that it is labor intensive and requires subject matter expertise. The advantage of feature engineering is that once identified, the knowledge of

interactions can help provide business context and inspire business solutions. Note that this chapter has only discussed this one example of engineered features—interactions—but there are many, as discussed throughout the data preparation-themed chapters in this text. Deep learning, in contrast, does not invest time in this activity, shifting the data scientists' attention to other concerns. While deep learning avoids feature engineering while making predictions, it also sacrifices transparency and the explicit identification of interactions.

Postscript

The discussion of the differences between traditional machine learning and deep learning algorithms may beg the question: "How can we know why the algorithms generated their predictions?" Chapters 18 and 19 will answer this question.

Keywords and phrases

Interactions, Regression, Algorithms, Perceptron, Feature selection, Data preparation, Deep Learning, Machine learning, Neural network, Hidden layer.

References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., & Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. *OSDI'16: Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, 265–283. November 2016.
- Aggarwal, C. (2023). *Machine learning with shallow neural networks: Neural Networks and Deep Learning*. Cham: Springer. https://doi.org/10.1007/978-3-031-29642-0_3.
- Aiken, L. S., & West, S. G. (1991). *Multiple Regression: Testing and Interpreting Interactions*. Sage Publications, Inc.
- Association of Computing Machinery (ACM), 2018. Fathers of the Deep Learning Revolution Receive ACM A.M. Turing Award. <https://awards.acm.org/about/2018-turing>
- Fukushima, K., 1979. 位置ずれに影響されないパターン認識機構の神経回路のモデル --- ネオコグニトロン [Neural network model for a mechanism of pattern recognition unaffected by shift in position — Neocognitron]. *Trans. IECE (in Japanese)*, J62-A (10), 658–665.
- Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4), 193–202. <https://doi.org/10.1007/bf00344251>.
- Galit Shmueli, 2010. To Explain or Predict. <https://projecteuclid.org/journals/statistical-science/volume-25/issue-3/To-Explain-or-to-Predict/10.1214/10-STS330.full>
- Hubel & Wiesel, 1981. <https://www.nobelprize.org/prizes/medicine/1981/summary/>
- Hubel, D., & Wiesel, T. (1959). Receptive fields of single neurons in the cat's striate cortex. *Journal of Physiology*, 148(3), 574–591. <https://doi.org/10.1113/jphysiol.1959.sp006308>.
- J. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Li, F.-F., 2009. ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, pp. 248–255, doi:10.1109/CVPR.2009.5206848.
- Khabaza, 2010. Nine Laws of Data Mining. http://khabaza.codimension.net/index_files/9laws.htm
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.
- Krohn, J., Beylerveld, G., & Bassens, A. (2019). *Deep Learning Illustrated (Addison-Wesley Data & Analytics Series)*. Pearson Education.
- Lecun, Y., Bottou, L., Bengio, Y., & P. Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324. <https://doi.org/10.1109/5.726791>.
- Marcus, G., 2018. Deep Learning: A Critical Appraisal. <https://arxiv.org/ftp/arxiv/papers/1801/1801.00631.pdf>
- McCulloch, W. S., & Pitts, W. , (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4), 115–133. <https://doi.org/10.1007/bf02478259>.
- New York Times, 2012. Scientists see advances in deep learning a part of artificial intelligence. <http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html>.
- Ng, A. (2013). *Machine learning and AI via brain simulations*. Stanford University. <https://ai.stanford.edu/~ang/slides/DeepLearning-Mar2013.pptx>.
- OpenAI, 2022. Deep learning theory session: The unreasonable effectiveness of deep learning. Ilya Sutskever, presented at OpenAI Week, February 7–9, 2022. <https://www.youtube.com/watch?v=BjyZcSiVg5A>
- Rumelhart, D., G. Hinton, and R. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323:533–536.

Interpretable machine learning and explainable AI

Preamble

Many machine learning approaches produce models that are impractical to interpret directly. Notable among them is deep learning, which we discussed in [Chapter 17](#), but there are others, including the ensemble techniques we discussed in [Chapter 16](#), especially XGBoost. This chapter will discuss some techniques to help explain model prediction in terms of the variables used to train them. These explanations are needed to help model users justify the predictions of their models that predict various outcomes that have enormous impacts on people (e.g., loan applications, fire risk, and prison sentences).

Introduction

Use of data analytics models is becoming rather commonplace in almost every aspect of our society. Many of these new models are generated by deep learning or large language model algorithms. The problem is that most of these models are very *opaque*; that is, predictions made by these models cannot be explained easily in terms of the variables or data sources used to train them. Four trends are making opaque models more commonplace:

- AutoML and its ability to rank hundreds of models in order of accuracy
- The rise of deep learning
- The popularity of ensemble techniques
- The visibility of ML modeling competitions based on model accuracy

One principle should be stated very clearly: accurate models have always been very desirable, and they will continue to be so. The failure, however, to weigh other factors together with prediction accuracy is a common mistake that inexperienced modelers often commit. Tom Khabaza's Value Law ([Khabaza, 2010](#)) warns us that "The

value of data mining results is not determined by the accuracy or stability of predictive models." Often, the elucidation of important predictors is far more important than the predictive accuracy level of the model. Sometimes relationships that are not found are just as significant as those that are found. This mapping of causes and effects in the output of a model is referred to as "transparency."

Concurrently, forces at work are making model transparency more critical than ever. Model transparency has always been required in regulated industries, but now, we are beginning to see regulations specific to machine learning. The General Data Protection Regulation (GDPR) has already been passed into law in the European Union (EU) and has clauses relevant to AI and ML modeling. We will discuss this further in the next section. The AI Act is a proposed European law on artificial intelligence (AI) and will be the first significant law of its kind. ML model biases in hiring, criminal sentencing, lending, and other use cases have been increasingly newsworthy and common. See <https://artificialintelligenceact.eu/> for more information on the AI Act.

Recognizing these trends, the Association of Computing Machinery dedicated a cover story to the problem in their membership publication ([Mengnan et al., 2020](#)). The authors state that:

Machine learning is progressing at an astounding rate, powered by complex models such as ensemble models and deep neural networks (DNNs). These models have a wide range of real-world applications, such as movie recommendations of Netflix, neural machine translation of Google, and speech recognition of Amazon Alexa. Despite the successes, machine learning has its own limitations and drawbacks. The most significant one is the lack of transparency behind their behaviors, which leaves users with little understanding of how particular decisions are made by these models.

How do we reconcile these two trends in AI: the growing need for AI applications, and the lack of transparency of its products? Academics, practitioners, and software vendors have tried to address the problem with a wave of activity in developing new approaches to

producing post hoc explanations of opaque models. Collectively, these efforts are referred to as explainable AI (XAI). *“The AIOps platform market size is expected to grow from \$2.83 Billion in 2021 to \$19.93 Billion by 2028; it is estimated to grow at a CAGR of 32.2% from 2021 to 2028” (The Insight Partners, 2022).*

Comparing interpretable machine learning and XAI

It is helpful to distinguish between two overall strategies for providing transparency. The first strategy of building and deploying models using algorithms that produce inherently interpretable models is still possible. This approach may be underutilized. Chasing after the perceived assurance of higher accuracy may seem appealing in the short term, but opaque models face challenges during deployment and defense of results. This practice may be a contributing factor in the oft-quoted fact that only a small minority of models reach deployment. The second strategy is to build sufficiently complex and accurate models that are nonetheless interpretable (aka interpretable machine learning – IML). Interpretable models provide a method (usually specific to the algorithm) to identify two things: (1) variables

important to specific predictions (local explanations) and (2) variables of overall import to the model (global explanations). The distinction between these two criteria will become clearer in the discussion of local and global explanations in the next section.

Interpretable machine learning

Classic examples of inherently interpretable models include linear regression and single decision trees, although there are many more. The common feature that all these options share which makes them directly interpretable is that they all have a method, usually specific to the algorithm, that allows the easy determination of variable importance. For example, in linear regression, the standardized beta coefficient allows for the direct comparison of the importance of independent variables.

Linear regression

The data processing operations for linear regression result in estimating a parameter (called a “coefficient”) for each predictor variable, which ranges from 0.0 to 1.0. The magnitude of this coefficient is a reflection of two factors:

- The scale of the predictor variable data (the x values).

TABLE 18.1 KNIME logistic regression coefficients and statistical scores for a wine color model.

Row ID	S Logit	S Variable	D Coeff.	D Std. Err.	D z-score	D P> z
Row0	red	volatile acidity	1.18	0.125	9.452	0
Row1	red	total sulfur ...	-2.163	0.159	-13.564	0
Row2	red	Constant	-3.53	0.236	-14.973	0
Row3	red	density	1.572	0.207	7.594	0
Row4	red	chlorides	0.816	0.108	7.53	0
Row5	red	residual sugar	-1.376	0.192	-7.156	0
Row6	red	sulphates	0.745	0.12	6.214	0
Row7	red	pH	0.785	0.128	6.142	0
Row8	red	fixed acidity	0.8	0.157	5.1	0
Row9	red	alcohol	0.345	0.147	2.351	0.019
Row10	red	quality=mod...	-0.463	0.266	-1.739	0.082
Row11	red	quality=slig...	0.379	0.224	1.694	0.09
Row12	red	citric acid	-0.2	0.127	-1.58	0.114
Row13	red	free sulfur d...	0.113	0.166	0.679	0.497
Row14	red	quality=neu...	0.142	0.218	0.65	0.516
Row15	red	quality=slig...	0.047	0.251	0.187	0.851
Row16	red	quality=mod...	-0.028	0.295	-0.095	0.925
Row17	red	quality=extr...	-0.002	0.316	-0.005	0.996

(Data from the UC-Irvine Machine Learning Repository.)

- The importance of the variable in calculating the value of the predicted outcome (the y variable).

The sign of the coefficient indicates whether the relationship is direct (positive) or indirect (negative).

If we want to isolate the effect of the importance of individual variables, we must convert all variables to the same scale. This process is called “normalization.” One normalization approach is to convert all variable scales to range proportionately from 0.0 to 1.0. However, the most common normalization approach is to use a z -transform to convert the x values to a common scale, which is more compatible with the probability distribution of data composing a Normal curve, also used by the regression algorithm. Table 18.1 shows an output table of a linear regression model of the Wine data set in the UC-Irvine Machine Learning Repository.

Assuming that the level of significance is set to 0.05 (the 95% confidence level), the top 10 variables (Row0 – Row9) have $P > [z]$ values less than 0.05. These variables constitute the important predictors of wine color in this data set.

Decision trees

Variable importance determination is slightly more complicated in decision trees. It is an accepted rule of thumb to consider that variables selected near the root node are the most predictive. Some algorithm implementations like the KNIME Random Forest algorithm provide an output table that can be related to variable importance. Table 18.2 lists the count of the number of times a variable is used in the first three splits in building the decision tree. Variables not used at all are determined by the decision tree processing to be unimportant in building the decision tree. Evaluation of models trained with the KNIME Random Forest algorithm was discussed previously in Chapter 15. Also discussed in Chapter 15 was the output table of the KNIME XGBoosted Trees algorithm that expressed variable importance in terms of information gain. The output table of the Random Forest algorithm is repeated here to emphasize the importance of XAI operations in developing transparent models.

TABLE 18.2 KNIME Random Forest decision tree split statistics for the German Credit Score data set.

Row ID	#splits (level 0)	#splits (level 1)	#splits (level 2)
>=200 DM_Balance of current account	16	12	18
paid back previous credits at this bank_Payment of previous credits	10	14	12
6 or more_Number of previous credits at this bank	9	3	4
no running account_Balance of current account	6	6	7
vacation_Purpose of credit	5	5	4
no savings_Value of savings or stocks	5	1	3
rented_Type of apartment	5	3	7
problematic running accounts_Payment of previous credits	4	1	7
items of furniture_Purpose of credit	4	4	4
greater than 1000 DM_Value of savings or stocks	4	4	6
yes_Foreign worker	4	3	5
<= 200 DM_Balance of current account	3	5	6
hesitant payment of previous credits_Payment of previous credits	3	4	6
free apartment_Type of apartment	3	1	2
new car_Purpose of credit	2	4	6
between 100 and 500 DM_Value of savings or stocks	2	6	9
unemployed_Has been employed by current employer for	2	4	11
ownership of house or land_Most valuable available assets	2	2	0
Amount of credit in DM	1	8	14
repair_Purpose of credit	1	3	4
other_Purpose of credit	1	0	8
business_Purpose of credit	1	7	5
between 500 and 1000 DM_Value of savings or stocks	1	5	5
greater than 7 years_Has been employed by current employer for	1	3	1
between 25 and 35_Installment in % of available income	1	3	4
male:married/widowed_Marital Status/Sex	1	1	4
executive/self-employed/higher civil servant_Occupation	1	6	7
0 to 2_Number of persons entitled to maintenance	1	4	4
no_Foreign worker	1	5	5
Duration in months	0	3	6

(Data from the UC-Irvine Machine Learning Repository.)

Data values in Table 18.2 have been sorted on Split0 (the node closest to the root node). If we use this criterion, 29 of the 72 input variables are important predictors of creditability in the model. The strength of the importance metric is reflected in the magnitude of the split count, thus the most important variable in the German credit score model is $>+ 200$ DM_Balance of current account. Another important metric could be defined as the total number of times a variable was used across the first three splits in the model.

The approach described above “opens up” the decision tree model to show how the predictions were determined. A set of if-then-else rules can be induced by many decision tree algorithms, which can be translated into business rules to show to management staff. This approach can also be used to open up a “black box” algorithm like a neural network by building a decision tree model to predict the predictions of the neural network algorithm. This type of model is called a *surrogate model* and will be discussed below.

It may be that an opaque model will prove to be the most effective option among all modeling options for meeting project criteria. If an opaque model is to be used, it requires a second step to produce local XAI explanations (e.g., a list of important predictor variables). XAI refers to making calculations (or even producing another model) to explain a complex, opaque model. In the next section, we will discuss the various types and techniques of XAI-produced model explanations, but first let’s provide some guidelines for choosing between inherently interpretable models and XAI.

Researcher and academic Cynthia Rudin of Duke University believes that the association with higher accuracy with opaque models is overemphasized and often untrue (Rudin, 2019b). Rudin remarked, “Perhaps we are using complicated models when we don’t need them.” She emphasizes two issues. First, the increased performance (effectiveness) of opaque models over inherently interpretable models is often small or non-existent. Second, a new generation of data scientists is being trained only in opaque modeling techniques. Despite her important reminders, it is unlikely that the trend toward opaque models will change, nor will the movement toward increased transparency requirements. The combination of these trends with the readily available technology to provide explanations makes modeling without explanations a violation of best practices.

Therefore our recommended approach is the following:

1. Carefully identify your requirements for model transparency, including regulatory requirements and those specific to your company and your project.

2. Always consider an inherently interpretable option and compare its accuracy to opaque modeling options.
3. Explore deployment considerations before committing to a modeling strategy.
4. If an opaque model best meets the project criteria, produce model explanations using XAI.

Comparing two types of XAI: global and local explanations

Global explanations tell us which input variables are most important to the model overall. Therefore global explanations are analogous to standardized beta coefficients in multiple linear regression. Standardized beta values are associated with that specific technique, and *are not produced by* decision trees. Some decision tree algorithms can, however, provide the count of the number of times a given variable is used in the first few splits in building a decision tree. The list of predictor variables can be sorted in descending order on count number of the first split or the total count for several initial splits to yield a list of important predictor variables in the decision tree model. Other decision tree algorithms can output a list of decision rules induced from model data. In this sense, these explanations are *global*, *post hoc*, and *model specific*. An important difference with XAI is that global explanations in XAI are typically both *model-agnostic* and *post hoc*. In an inherently interpretable technique like multiple linear regression, standardized beta coefficients are generated as part of building the model. Post hoc explanations are produced using the model predictions after the model is built. Intermediate products of the regression calculations (e.g., sums of squares) are necessary inputs to generate the variable coefficients, which are specific to the regression algorithm. Therefore regression explanations are *model dependent*. Rather, *global* explanations are model independent (model agnostic), meaning the same technique can be used with any modeling technique. One only needs the original input variables and the model predictions.

Common global explanation techniques include the following:

- *PDP: Partial Dependence Plot*
- *Surrogate models*
- *Permutation Feature Importance*

These three techniques will be briefly described in the section “Techniques for global explanations.”

Local explanations indicate which inputs were most important for a specific *individual* prediction. Like global explanations, XAI-produced local explanations are typically both post hoc and model agnostic. The primary reason that one customer was declined for the loan

might not be the same as the primary reason that another customer was declined for the loan. Local explanations allow you to explore this and to provide reason codes to customers.

Common local explanation techniques include the following:

- **SHAP:** Shapley Additive exPlanations
- **LIME:** Local Interpretable Model-Agnostic Explanations

These local explanation techniques will be explained further in the section “Techniques for local explanations.”

It is important to note that the purpose of local explanations is not only to meet regulatory requirements (like those mandated by GDPR; see below) but also to provide a service to customers. Model explanations are crucial even when no external forces require them. Marco Ribeiro developed LIME because he ran into a model challenge. He was trying to debug his model. He was experiencing a very stubborn issue with the failure of his model to generalize to new data. The purpose of LIME is to find a locally faithful linear explanation even when a globally faithful linear explanation is not possible. To borrow the title of his original paper on the subject, it attempts to answer the question “Why should I trust you?” We will explain LIME and the other two common local explanation techniques in the section “Techniques for local explanations.”

“It took me forever to figure out what it was ... The model had learned to distinguish between how we had collected positive and negative data rather than picking up on what it should be picking up on.” Marco Ribeiro, developer of LIME (<https://arxiv.org/abs/1602.04938>)

His experience is worth exploring. He used hold-out validation, as is commonly done. Since the hold-out represents “unseen” data, the model’s ability to predict accurately the instances of the hold-out data set is a measure of model “robustness” and the stability of model for use in the future. Yet, Ribeiro’s model failed to do so. Only *after* he had local explanations *in hand* did he discover the problem. An artifact of his data preparation affected both his train and test data... but was not affecting his new data. His work on LIME grew out of an attempt to identify the flaw.

This anecdote provides two critical reminders. If we are modeling “blind” with opaque models, we might not be able to rely on foundational techniques like hold-out validation. It is an incomplete picture of modeling performance. Also, we may be experiencing high accuracy for the wrong reasons (e.g., a variable related to the target variable by definition is used as a predictor of it). In his case, the problem was in his data preparation.

XAI explanations help us determine when our models are making predictions based on inappropriate clues in our data that will be unavailable to the model at deployment.

Li and Gadyatskaya (2023) define global XAI in terms of:

- **Stability** – Outcomes are unchanged during perturbations (e.g., stock market downturns).
- **Robustness** – Outcomes are unchanged by data not used to train the model.
- **Effectiveness** – Metrics that express how well a model works (e.g., Sensitivity).

Metrics of model effectiveness were discussed in Chapter 15. Criteria of model stability and robustness can be evaluated only when external conditions change (e.g., new data become available).

Techniques for global explanations

Partial Dependence Plot

Partial dependence plots (PDPs) are commonly available and easy to generate and interpret, but they have limitations. They are model agnostic and can be run post hoc on any model. PDPs show the relationship of one independent variable against the dependent variable for the entire range of the independent variable, which is what makes it useful for global explanations. If the relationship is linear and monotonic and relatively independent of other predictors, the graphic will be a faithful representation of the relationship. An example generated in KNIME is shown in Fig. 18.1.

Notice that at relatively low alcohol content levels, the wine quality probability is relatively low, but the wine quality probability level is relatively high (~0.9) at alcohol content above 11.5%. Therefore the alcohol content is probably one of the important predictors of the wine quality model.

According to Molnar, the biggest concern is the independence assumption.

The assumption of independence is the biggest issue with PD plots. It is assumed that the feature(s) for which the partial dependence is computed are not correlated with other features. For example, suppose you want to predict how fast a person walks, given the person’s weight and height. For the partial dependence of one of the features, e.g. height, we assume that the other features (weight) are not correlated with height, which is obviously a false assumption. (Molnar, 2022)

It is also notable that when the relationship is more complex than a linear and monotonic one, it makes it difficult to make a global interpretation because Y may

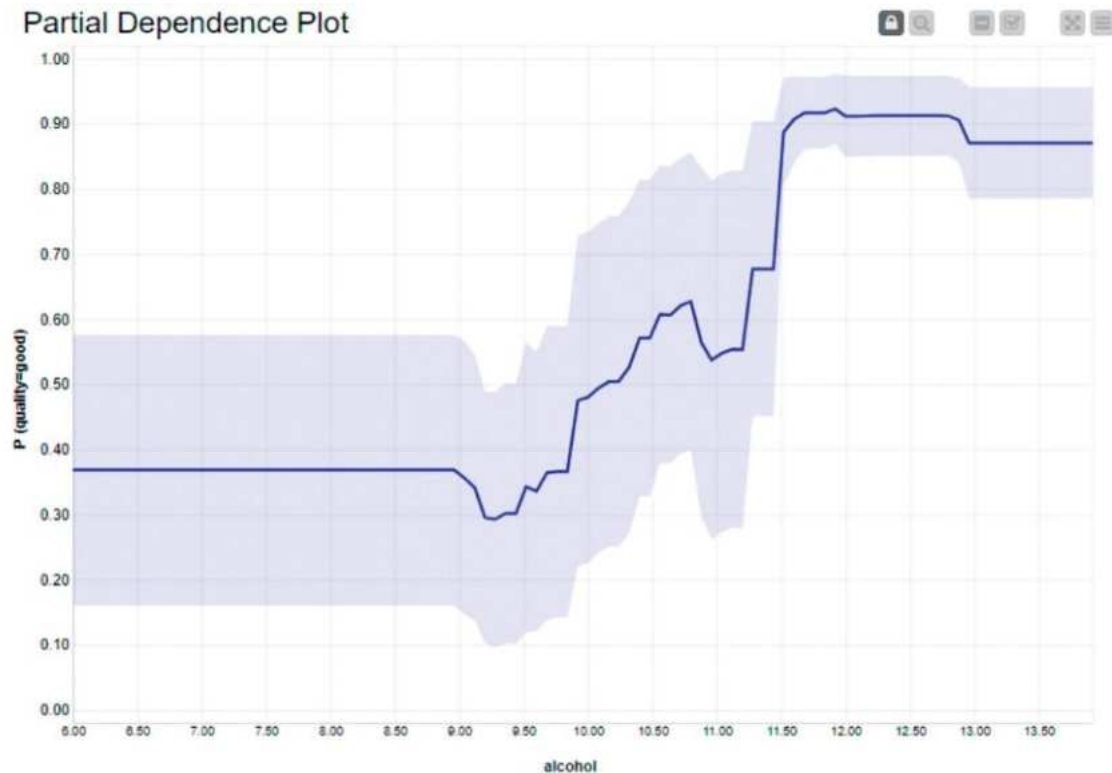


FIG. 18.1 Partial Dependence Plot of the effect of % alcohol in red wine on wine quality, using a KNIME Gradient Boosted Trees model.

sometimes go up as X goes up but also go down as X goes up. This is precisely the situation that makes LIME attractive as it attempts to make locally faithful linear explanations for small ranges of X . LIME is discussed in more detail in the next section.

Surrogate models

The concept behind surrogate models is fairly straightforward – build a transparent model with the predictions of an opaque model as the dependent variable. It is model agnostic, as all you need are the model predictions and the original inputs. Given those variables, you don't need any special algorithm or XAI routine, and you can simply choose an inherently interpretable algorithm and declare the black box model predictions as the target variable. A common choice is a single decision tree. The KNIME Random Forest algorithm provides an output of variable statistics that can be used to express variable importance. You could even build more than one surrogate with different algorithms. If you were predicting a scale target, you could build a surrogate regression tree and a surrogate regression model.

Here are some observations and practical advice about using this technique. First, the surrogate is often very accurate in predicting the original model because the black box model predictions have less noise than the original target variable. Second, and for the same reason,

the surrogate model tends to be less complex than a typical model of the same type. If a tree, for instance, may have just a couple of layers on the same settings, you typically see a deeper tree. You may want to set parameters to produce a more flexible model than you normally would. For example, if using a decision tree as a surrogate, consider allowing for a deeper tree with smaller terminal nodes.

But it's important to remember that it isn't a second version of your model, as the target variables of the original model and the surrogate model are not the same. The surrogate is an attempt to explain the first model. Rudin (2019a) warns us about an implication of this situation.

An explainable model that has a 90% agreement with the original model indeed explains the original model most of the time. However, an explanation model that is correct 90% of the time is wrong 10% of the time.

Rudin goes on to make a specific critique of surrogate models.

A more important misconception about explanations stems from the terminology "explanation," which is often used in a misleading way, because explanation models do not always attempt to mimic the calculations made by the original model. Even an explanation model that performs almost identically to a black box model might use completely different features, and is thus not faithful to the computation of the black box.

This is a reminder of the advice that we considered at the top of the chapter – if you think that an inherently interpretable model could be built that could completely replace the black box model, you should do so. Therefore you should use a black box model combined with a surrogate only when it is not possible to build an equally effective simple model, and even then, only when some lack of transparency is tolerable.

These considerations should be made clear to stakeholders in the analysis as they may come to believe that the simpler model is the model that is being deployed. Once this critical distinction is made clear, most stakeholders will find the surrogate model approach and the surrogate model to be intuitive and easy to interpret.

Permutation Feature Importance

The idea of introducing randomness to measure variable importance has a long history, certainly as far back as the introduction of Random Forests by Brieman, but also with a resemblance to some ideas in the Bootstrap (Efron). In Permutation Feature Importance, each input variable is permuted or “shuffled,” and the model error is measured again each time. As Molnar explains:

A feature is “important” if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction. A feature is “unimportant” if shuffling its values leaves the model error unchanged, because in this case the model ignored the feature for the prediction. (Molnar, 2022)

Note that the model is not retrained. For this reason, this technique is also model agnostic and post hoc. The model is not needed, but the predictions, the original inputs, and the target variable are all needed. At first glance, it may seem more natural to drop each variable and retrain the model, but that would require rebuilding the model as many times as there are inputs, which would be computationally expensive. Also, it would produce a myriad of models with all the complexity that entails. It would also fail at the intended mission, which is to determine the variable importance of each variable on a single existing static model.

Techniques for local explanations

SHAP

Shapley Additive exPlanations (SHAP) by Lundberg and Lee (2017) is based on a *concept borrowed* from game theory, Shapley values. Let’s discuss Shapley values first and then discuss how Lundberg and Lee adopted the concept for XAI. Shapley values are named after Lloyd Shapley, who introduced the concept in 1951 and won the Nobel Memorial Prize in Economic Sciences for it in

2012. Shapley values attempt to measure the individual contribution of various players in achieving a common goal. Sharing a taxicab has become a classic illustration of the concept.

Imagine that Ann, Bob, and Charles have just attended an event together at Lincoln Center in New York City and are now sharing a cab home. All are headed in the same general direction, south toward downtown, but they vary in how far they have to travel:

- Ann is only traveling as far as 42nd Street and Broadway, and if she were traveling alone, the fare would be \$15.
- Bob is traveling about twice as far down Broadway and would pay \$30 if he were traveling alone.
- Charles is traveling the farthest, to downtown, and would pay nearly \$60 *after dropping off Ann and Bob*, and then continues via Broadway. *If he were alone*, however, he would likely take the West Side Highway instead of Broadway, *saving* \$15.

The idea behind Shapley values is to list all of the possible “coalitions” and then to average across the scenarios. See Table 18.3.

One can use online calculators, including <http://shapleyvalue.com/>, to help with the math. In this case, we discover that the fair way to split up the taxi is: \$7.5 for Ann, \$15 for Bob, and \$37.5 for Charles.

Lundberg and Lee adopt this approach for providing local explanations and apply it to “coalitions” of variables (instead of coalitions of players), or, as they phrase it:

When the model is non-linear or the input features are not independent, however, the order in which features are added to the expectation matters, and the SHAP values arise from averaging the values across all possible orderings.

SHAP values attribute to each feature the change in the expected model prediction on an individual case when conditioning on that feature. The aspect of additivity is noteworthy – the sum of the SHAP values is equal

TABLE 18.3 Use of SHAP values to evaluate the fair share of the taxi fare for each passenger.

Coalition	Taxi fare
Ann traveling alone	\$15
Bob traveling alone	\$30
Charles traveling alone via the highway	\$45
Ann and Bob together	\$30
Bob and Charles together via Broadway	\$60
Ann and Charles together via Broadway	\$60
All three via Broadway	\$60

to the model prediction. So the magnitude of the SHAP value allows for a direct comparison of variable importance for an individual prediction (Lundberg and Lee, 2017.)

LIME

Local Interpretable Model-Agnostic Explanations (LIME) was first proposed by Ribeiro et al. (2016). As the acronym indicates, LIME is model agnostic, and its explanations are produced post hoc. The authors explain the goals of their work in the following way:

Despite widespread adoption, machine learning models remain mostly black boxes. Understanding the reasons behind predictions is, however, quite important in assessing trust, which is fundamental if one plans to take action based on a prediction, or when choosing whether to deploy a new model. Such understanding also provides insights into the model, which can be used to transform an untrustworthy model or prediction into a trustworthy one. In this work, we propose LIME, a novel explanation technique that explains the predictions of any classifier in an interpretable and faithful manner, by learning an interpretable model locally around the prediction.

The premise of LIME is that curvilinearity and variable interactions can often make global explanations difficult, but even when this is the case, “locally faithful” explanations are possible. This is not to limit LIME to models like linear regression. It has also been applied to computer vision and natural language understanding (NLU), but we will use curvilinearity to explore the concept. For instance, imagine a curvilinear and parabolic relationship between x and y , where y increases as x increases until x is near its median, and then y starts to decrease as x increases. Interpolating a straight line to represent a globally faithful relationship across the entire range of x is not possible, but interpolating a straight line for smaller ranges of x is possible. Therefore we are only interested in understanding the output associated with a specific value of x and the “neighbors” of that specific value of x , using the term neighbors in the same sense as it’s used in k -nearest neighbors. The insight of the developers of LIME is to perturb the values of the input variables to see which changes have the greatest impact on the outcome. Then LIME fits a linear surrogate model to the area near the input value. That approach allows the local linear model of our parabola example to have a positive slope for some values of x but a negative slope for other values.

Counterfactuals

The idea of Counterfactuals, as they are used in XAI, comes from the work of Judea Pearl. It represents an exploration of what could cause an alternative

outcome – if not for X , then Y would have been different. Pearl (1999) claimed:

*According to common judicial standard, judgment in favor the plaintiff should be made if and only if it is more probable than not that the defendant’s action was the **cause** for the plaintiff’s damage.*

When this concept is applied to the interpretative of machine learning models, it is not merely speculative. It can provide a systematic approach to the explanation of black box models.

*We simply change the feature values of an instance before making the predictions, and we analyze how the prediction changes. We are interested in scenarios in which the prediction changes in a relevant way, like a flip in predicted class (for example, credit application accepted or rejected), or in which the prediction reaches a certain threshold (for example, the probability for cancer reaches 10%). A counterfactual explanation of a prediction describes the **smallest change to the feature values that changes the prediction to a predefined output.** (Molnar, 2022)*

Recall that this is a local explanation technique, so it is used to identify explanations about individual cases. For example, one might be interested in a specific loan applicant who was denied a loan. Then one searches for cases with the desired outcome (credit approval) but that otherwise have the closest distance on the same set of inputs. Therefore the difference between the credit denial and credit approval is those input values that “tipped the scale” in favor of approval.

Regulatory considerations

While much of the information about AI and the regulation of AI models is of recent origin and is quickly evolving, regulatory requirements for predictive modeling are not new. The Equal Credit Opportunity Act was originally passed into law in the United States in 1974. Creditors are required to provide “specific reasons” if credit is denied, and this information is typically provided in the form of “reason codes.” For instance, in a disclosure document, a consumer might be informed that the reason that a credit line increase was denied was “85: You have too many inquiries on your credit report.” More examples of the codes used by the three U.S. national credit reporting companies (CRCs) – Equifax, Experian, or TransUnion, can be found at <https://reasoncode.org/>. The provision of these reason codes requires that the creditor’s model has sufficient transparency. Therefore compliance requires either an inherently interpretable model, which would have been the dominant choice during decades since the law was passed, or the newer approach of XAI-based explanations.

GDPR in the EU

The General Data Protection Regulation (GDPR) was enacted by the EU in 2016 and took effect in 2018. Although not a law in the United States, its influence has already been felt, because domestic products developed for sale in the EU also must be GDPR compliant. For example, *Article 25* grants the right to data subjects to contest any automated decision making that was made on a solely *algorithmic* basis. This right requires the analytical software developers to provide output that can be used to justify how and why a decision is made based on the analysis results. This requirement means that analytical software developers of products to be sold in the EU must be able to show that their products are transparent.

Another compliance issue commonly associated with GDPR is the requirement to disclose the use of cookies to website visitors. However, there are clauses directly related to disclosure.

Such processing should be subject to suitable safeguards, which should include specific information to the data subject and the right to obtain human intervention, to express his or her point of view, to obtain an explanation of the decision reached after such assessment and to challenge the decision. <https://www.privacy-regulation.eu/en/r71.htm>

Much like the disclosure requirements under the Equal Credit Opportunity Act, providing an “explanation of the decision reached” requires either IML or XAI.

The proposed AI Act in the EU

While the *AI Act* has not yet been passed, the EU has made public some of its considerations for the act. In their “preferred option,” there would be “a regulatory framework for high-risk AI systems only, with the possibility for all providers of non-high-risk AI systems to follow a code of conduct.” They provide an example of *a chatbot as a lower-risk application with only a “minimal transparency requirement.”* By implication, many applications of import to predictive modelers would require some transparency and “trustworthy AI.” (*define “trustworthy”?*)

One extended passage from the proposed AI Act states the following:

The requirements will concern data, documentation and traceability, provision of information and transparency, human oversight and robustness and accuracy and would be mandatory for high-risk AI systems. Companies that introduced codes of conduct for other AI systems would do so voluntarily.

Until passage, we can’t be certain of the details, but there is already sufficient evidence in the proposal to

assume there will be requirements related to providing model explanations. ([European Commission, 2021](#)).

While the details are still evolving, it appears that the European regulators are trying to distinguish between two kinds of transparency. First, a right to “explanations of decisions”:

MEPs want to boost citizens’ right to file complaints about AI systems and receive explanations of decisions based on high-risk AI systems that significantly impact their rights. ([European Parliament, 2023](#))

Second, an attempt to enforce transparency into the source data used in foundation models:

Generative foundation models, like GPT, would have to comply with additional transparency requirements, like disclosing that the content was generated by AI, designing the model to prevent it from generating illegal content and publishing summaries of copyrighted data used for training. ([European Parliament, 2023](#))

The first of these is more closely aligned with the topics in this chapter, but foundation models will certainly be increasingly important to the field in the coming years.

Postscript

This chapter focused on explaining to humans the reasons why predictions of outcomes of great interest to them were made (e.g., loan disapprovals). [Chapter 19](#) will discuss why it is important not only to keep the human in the “loop” of understanding model outcomes, but also the critical importance of keeping the human in the loop of building the models in the first place.

Keywords and phrases

Interpretable; XAI, Transparent, Predictions, Counterfactuals, Global explanations, Local explanations, Black box, Variable importance, Opaque models, Surrogate models, Random Forest.

References

- European Commission, Proposal for a regulation of the European Parliament and of the Council, 2021.
- European Parliament, AI Act: a step closer to the first rules on Artificial Intelligence, press release 2023.
- Khabaza, Tom, “Nine Laws of Data Mining,” http://khabaza.codimension.net/index_files/9laws.htm
- Li, R., & Gadyatskaya, O. (2023). Evaluating rule-based global XAI malware detection methods. In S. Li, M. Manulis, & A. Miyaji (Eds.), *Network and system security. NSS 2023. Lecture Notes in Computer Science* (vol 13983). Springer Cham https://doi.org/10.1007/978-3-031-39828-5_1.

- Lundberg, Scott and Lee, Su-In, A unified approach to interpreting model predictions. NIPS Conference, 2017.
- Mengnan Du, Ninghao Liu, and Xia Hu, Techniques for interpretable machine learning, *Communications of the ACM*, January 2020.
- Molnar, C. 2022. Interpretable machine learning: a guide for making black box models explainable. Mucbook Clubhouse, Westenstrasse 78, München, Germany.
- Pearl, Judea, Probabilities of causation: three counterfactual interpretations and their identification, Vol. 121, No. 1/2, *Statistics and Causation* (1999), pp. 93-149 (57 pages). Springer.
- Ribeiro, M., S. Singh and C. Guestrin. 2016. "Why should I trust you?" Explaining the predictions of any classifier. arXiv:1602.04938v3 [cs.LG] 9 August 2016. See <https://arxiv.org/abs/1602.04938> for a PDF copy.
- Rudin, C. (2019a). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, May 2019. *Nature Machine Intelligence*, 1(5), 206–215. <https://doi.org/10.1038/s42256-019-0048-x>.
- Rudin, C. 2019b. Keynote address at the KDD 2019 Conference, VideoLecturesChannel, <https://www.youtube.com/live/wL4X4IG20sM>
- The Insight Partners, 2022, <https://www.globenewswire.com/news-release/2022/04/13/2421748>

Human in the loop machine learning and data annotation

Preamble

The concepts of artificial intelligence and human action may appear to be a contradiction in terms, but this is not actually the case. Consider the analogy presented in the preface that the practice of data analytic modeling is like driving an automobile. Modern cars have a number of automated systems in operation (e.g., climate control and mixture control of air and fuel), but human action is still required. Granted, the hype concerning self-driving cars would have you believe that automatic driving has arrived. (Elon Musk would have you believe that a Tesla automobile can do it.) A completely automated drive has not arrived yet, nor has completely automated analytic modeling. This chapter will support that premise by showing the critical necessity of keeping the human-in-the-loop of AI applications.

Introduction

AI is more powerful than ever before, but it still needs a lot of human support, perhaps more than is obvious. The evidence of that fact appears in many places in our society ... if you know where to look. When you enter many websites, you are faced with the little visual puzzles that you have to complete to prove that you are not a robot; these puzzles are created by the app called CAPTCHA. It stands for *Completely Automated Public Turing Test To Tell Computers and Humans Apart* (<http://www.captcha.net/>).

The term was coined by a team at Carnegie Mellon University in 2000. By identifying distorted text or which little images have buses or telephone poles or motorcycles, we're proving that we are not a computer program trying to scrape the data off a website. What is less commonly understood is that CAPTCHA is also a form of crowdsourced human-in-the-loop data annotation.

What Web readers do not know, however, is that they have also been enlisted in a project to transform an old book, magazine, newspaper or pamphlet into an accurate, searchable and easily sortable computer text file. (New York Times, 2011)

It's not just using that information to allow us to enter the site, but it's also gathering labeled data for use in future machine-learning applications. There is another example of information gathering that is very common – phishing, a particularly dangerous form of it (FTC, 2022). Phishing is an attempt to trick a recipient into revealing highly sensitive information like bank account information by pretending to be a service provider. It's often a lost package or an automated renewal for a service that we don't recognize. Spam filters use machine learning in an attempt to identify these phishing attacks. Email service providers, including Gmail, provide a routine to report this very specific and very problematic type of email. This information is used to improve the machine learning models that try to identify this kind of attack. This is also a form of human-in-the-loop data annotation.

As we've discussed throughout the book, the dominant form of machine learning is supervised machine learning, which requires “labeled” data. The labeling requirement appears in two distinct forms that look quite different at first. Let's revisit what is increasingly called traditional machine learning as was discussed in Chapter 17, which is the type of machine learning that has been performed in both academia and industry for decades, and is the subject of this book.

Take credit risk, for instance. FICO credit scores are calculated using many different pieces of credit data in your credit report. These data are grouped into variable categories like payment history, amounts owed, length of credit history, new credit, and credit mix. We can take all of that information at the time when someone attempts to take on a new debt, then wait a year to see if there is a default. The credit data are called input or predictor variables, and the eventual outcome is called a target variable. But this *target variable* is the same idea

as what is called data *labels* in a different context. These terms have been used in various previous chapters in this book. What should be made explicit is this: there is no need to “label” CAPTCHA data (for example) because all of this information is being captured digitally during the normal course of doing business. While this kind of machine learning has been practiced for decades, there has not been a large demand for human data annotation until much more recently.

In computer vision analysis, there are iconic examples like detecting whether the pet in a picture is a cat or a dog, or whether a bicycle, pedestrian, or stop sign can be detected in front of an autonomous vehicle. In these cases, the objects in images have to be labeled by a human until there are enough examples that a computer can learn to recognize them on its own. That is another kind of supervised learning, pairing images with data labels. This human labeling to create a data set is just one part, a very important part, of human in the loop. It is not a coincidence that this requirement has grown massively in scale concurrently with the rise of deep learning, which has provided the technology to “mine” specific visual objects in image files.

Therefore the growing data annotation industry has not focused on labels in traditional machine learning use cases like loan default, insurance fraud, customer loyalty, and marketing response, because data for these use cases are recorded digitally as the data is captured in online forms and other customer interactions. Good business record keeping naturally accumulates a wealth of data that enables the analyst to pair the input variables (like credit info) with the ultimate outcome (whether they defaulted or not). That’s not true with our dog and cat pictures, however. Some humans have to provide the labels for those data sets.

Defining human in the loop

A one-sentence definition of human-in-the-loop (HITL) in machine learning is simply this: HITL is *humans and computers working together*. A more comprehensive and detailed definition is helpful. In his excellent book, Robert (Munro) Monarch defines

“Human-in-the-Loop Machine Learning” as ...

the set of strategies for combining human and machine intelligence in applications that use artificial intelligence. The goal is typically to solve one or more of the following:

Increasing the accuracy of a machine-learning model

Reaching the target accuracy for a machine learning model faster

Combining human and machine intelligence to maximize accuracy

Assisting human tasks with machine learning to increase efficiency
(Monarch, 2021)

Although Monarch’s book is the most comprehensive currently available, one should be a bit careful with the phrase “Human in the loop” because not all experts use it the same way. Everyone who uses it means some kind of human/computer collaboration, but there’s more than one kind. Note, however, that by human-in-the-loop, we don’t generally refer to the actions of a model builder like a data scientist except in the data annotation tasks that they might perform. In other words, we aren’t contrasting human-built models with AutoML. That is a different issue.

For some, the phrase only means that a human has to be in the *middle or end* of the process with a computer passing information to a human, but only with the human’s input or approval does the process complete. Certainly, that does happen, but we will use a more comprehensive definition in terms of four use cases across the machine learning lifecycle.

First, to create training data to build models, we need *Data Annotation*. This can take the form of labeling images, but it often also includes detailed tasks like drawing a bounding box around the labeled image to clearly identify where in the image the cat, dog, stop sign, or bicycle appears.

Second, once you have a model, you’ll begin to be able to identify where your model makes errors. This may prompt you to increase the quantity or quality of the data to address the need. This often involves a technique called *Active Learning*.

Third, the users of your model can also provide feedback on when your model is working well. This technique, often called *Reinforcement Learning from Human Feedback*, has gotten a lot of attention recently with the progress of large language models (LLMs).

Fourth, sometimes a model gets stuck. It could be something as simple as an illegible receipt or bank deposit. A model can route a particular instance to a human, sometimes in real time, to address the problem. This kind of exception processing is sometimes called *Human Computation*, and it’s more common than you probably think. It may be at work in apps that you thought were 100% automated.

This isn’t a comprehensive list, but these four use cases represent some of the most important among them. Even this rather short list reinforces the notion that human in the loop machine learning uses human input at all stages of the machine learning lifecycle.

Concurrent rise of human in the loop and deep learning

In 1950, Alan Turing proposed the so-called Turing Test, which he called the Imitation Game. Machine learning techniques like traditional neural nets, which many

now call “shallow neural nets,” and decision tree algorithms, started to be applied in business settings at least as far back as the 1980s and 1990s. Academics were studying them for much longer than that. If supervised machine learning needs human-in-the-loop and is, in fact, decades old, you might wonder why some of the considerations we are discussing in this chapter are not more widely discussed.

It’s worth pausing for a moment to revisit the recent history that brought us here, some of which we discussed in [Chapter 17](#). In 2006, an academic at Princeton University, Fei-Fei Li, started assembling a famous data set filled with millions of images and labels. She used crowdsourcing to add the labels to the images, specifically Amazon’s Mechanical Turk crowdsourcing platform. In 2009, she presented a paper on her work, and starting in 2010, a competition was held yearly to build computer vision machine learning models to identify the images, now called the ImageNet Large Scale Visual Recognition Challenge ([Li, 2017](#)).

That set the stage for the rise of deep learning, and in the third year of the competition, 2012, everything changed. A team working with Geoffrey Hinton won the competition. What was noteworthy was that they won by a wide margin ... and they won it with deep learning neural networks. Neural networks have a feature called hidden layers. For decades, including the first two years of the Image Net competition, these neural networks had just a couple of hidden layers. In 2012, the winning model had eight hidden layers.

It may sound at first like a minor detail of interest only to specialists, but it changed everything. Ever since use cases for deep learning have exploded, neural networks have gotten bigger and more complex, and the need for labeled data has grown exponentially.

The kinds of data on which deep learning has outperformed everything else include images, like the ImageNet competition, and also video footage for things like driverless cars and natural language processing (NLP) like Amazon Alexa.

In response, an entire cottage industry has arisen to meet the need. Some organizations also have created data annotation teams in house. The more widespread deep learning applications become, the more organizations need to understand human in the loop.

Common human in the loop use cases

Medical imaging and radiology

Perhaps the most famous AI use case in healthcare analytics is medical imaging and radiology. About 8 years ago, Geoff Hinton, the deep learning pioneer, made a remarkable statement:

It’s just completely obvious that in five years, deep learning is going to do better than radiologists. It might be ten years. (*New York Times*, 2023)

Hinton concluded that we should stop training radiologists. In a recent interview, he stood by the prediction, but agreed that it would be closer to his 10-year prediction. Not surprisingly, radiologists at the time disagreed and still disagree now (*CBS News*, 2023).

Most experts agree that AI will eventually be more accurate than humans. Yet most radiologists believe that we will still have human doctors “in the loop” for decades to come. As one expert put it: “Artificial intelligence (AI) is here to stay in radiology—and so are radiologists” (*Stanford*, 2017).

First, there are some important differences between medical imaging and the challenge of recognizing if a social media image is a cat or a dog.

- What is often called “ground truth” is difficult to obtain. Unlike cats and dogs, experts can differ on what the image shows.
- The medically significant portion of the image can be a tiny portion of that image.
- A doctor might conclude that the illness that was tested for is not present, but that something else is going on that requires followup or another specialist.
- The diagnosis is not an end in itself. Identifying evidence of a problem in an image is just the first step; trying to establish a treatment plan has to be done next.

AI model development is always hungry for more data, so these applications need a lot of labeled data, and you need that data at various stages of the clinical process, not just an initial yes/no diagnosis. We’re likely to see hospitals pairing up with AI startups to steward these huge data management tasks.

The replacement prediction isn’t getting a lot of traction in medical circles. First, radiologists will get faster, eventually a lot faster. AI will be able to label, color code, and preprocess images for more rapid interpretation. We simply don’t know yet what the implications of that will be. One very optimistic prediction from Dr. Matthew Lungren is that we might reduce the number of undiagnosed preventable diseases like tuberculosis in parts of the world where many simply do not have access to radiology. Perhaps in time a general practitioner would be able to do more initial diagnosis even without access to a specialist (*Lundgren*, 2017).

One very exciting possibility is that if presented with lots of examples of medical imaging with before and after contrasting dye, the AI can anticipate what the image with contrast will look like. The dye has to be in the vein of a patient, which is expensive, can be painful,

and is sometimes not possible for certain patients. (Lundgren, 2017).

This would be an exciting application of AI, but clearly, the human doctor still needs to be present to establish “ground truth,” to interpret the images more quickly, possibly to bring care to those who currently don’t have access to it, and to provide treatment after diagnosis.

Autonomous vehicles

Autonomous vehicles are one of the biggest use cases for data annotation companies because there’s been a rush to produce huge amounts of annotated data for autonomous vehicles. These are complex use cases and, despite perpetual optimism, full automation has not been achieved. Video is much more labor intensive to annotate. While what is often called Level 5 (as defined by the Society of Automotive Engineers), fully autonomous vehicles are still being researched, most rental cars have what is called Level 1 automation (SAE International, 2014).

Progress is being made, in part, because we are accumulating an enormous amount of annotated traffic video footage. According to the Waymo website, their cars have collectively driven 20 million miles (<https://waymo.com/>).

One of the most daunting unmet challenges seems to be rare events. For instance, there were two accidents in recent years that have gotten a lot of media attention. One was an autonomous vehicle that didn’t recognize an overturned truck, and another a vehicle that failed to recognize a private jet. Naturally, these are not obstacles that frequently, if ever, appear in video footage. A business model like Waymo’s can reduce the risk of something like this by choosing routes that are less likely to encounter a rare event. For instance, they can avoid a traffic circle in favor of the more common four-way stop with traffic lights. Also, they are operating in two cities, with a few more planned, so they are accumulating a lot of footage of very specific areas and challenges, for instance, the challenges of reduced visibility going up and down San Francisco’s very steep streets.

So what is in the near future, and what is the connection to human-in-the-loop? These researchers and companies need more annotated footage of a wider variety of circumstances. The good news is that now that these cars are on the road, some of this video data will naturally accumulate over time. A number of Tesla cars with AutoPilot are on the road. When a human driver operates the car manually on their neighborhood street, they are generating training data. Through the combination of selecting the footage that will most improve the

model, with a focus on what the model has not seen yet, improved auto-annotation technology, and plenty of manual human annotation, the training data is getting more and more complete. There’s another way that Tesla improves the training data. They prompt the user to comment on why they turned off autopilot whenever they do so. This kind of exception processing and human feedback is very helpful, especially since it is often associated with rare events.

NLP use cases

An extremely common NLP use case is the operation of smart devices in many of our homes. If you have a smart device, like Amazon’s Alexa, then at one point or another, you’ve encountered a mistake. Maybe, through persistence and rephrasing, you overcame it, or maybe you gave up. But all of those mistakes are generating extremely valuable data that can help improve the model.

It’s become a sensitive topic in the media, so it isn’t discussed very often, but there are thousands of individuals required to listen to recordings so that the mistakes can be annotated, eventually leading to an improved model. Given our discussion so far, it shouldn’t be surprising, because we know that supervised machine learning requires this kind of data, and lots of it.

In a statement, Amazon told Bloomberg, “We only annotate an extremely small sample of Alexa voice recordings in order to improve the customer experience. For example, this information helps us train our speech recognition and natural language understanding systems, so Alexa can better understand your requests, and ensure the service works well for everyone.” (Bloomberg, 2019)

I want to draw your attention to the phrase “extremely small sample” – they are using what is called “Active Learning” to determine which samples to annotate. Active Learning is a collection of sampling methods to prioritize labeling those cases that are best able to improve the model’s performance (Monarch, 2021).

The media has focused on the privacy concerns, but AI companies are quick to point out that they are always innovating ways to avoid them. In addition to the privacy concern, human in the loop on this scale is also cumbersome and expensive. Here’s something that Amazon has been working on:

From Amazon’s own blog:

Another promising approach, from my Alexa AI team, is something we call self-learning.

*An Alexa customer who’s unsatisfied with the response to a request may interrupt the response and rephrase the request. With our self-learning system, Alexa uses this type of **implicit signal to deduce** — without any human involvement — how to respond correctly to the initial request. (Guo, 2020)*

This is an interesting development indeed and it might reduce the number of human workers who have to evaluate these errors, but we still have a human in the loop: us, the consumers of the service. With this technique, whenever we provide feedback, we are improving the training data and thereby improving the model, even when it is done simply by repeating an instruction.

Large language model use case: Reinforcement Learning with Human Feedback

Very possibly, the most exciting recent application of human in the loop is the use of Reinforcement Learning with Human Feedback to improve LLMs. Some have wondered why ChatGPT-4 captured the world's imagination and became the conversation of nonspecialists in a way that the earlier version just a few months before did not. A number of experts have speculated on what the reason for this situation might be (Arize, 2023).

Ouyang (2022) reported some details about how his team at OpenAI used this technique, titled “*Training language models to follow instructions with human feedback.*” The authors described LLMs as a sophisticated and complex type of auto-complete. The abstract of the paper helps set the stage:

Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not aligned with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback.

The premise of the article is that ChatGPT-4 is not specifically designed to perform tasks like writing a summary, but it can sometimes do that successfully, building upon its primary mission, which is to predict the next phrase.

Ouyang (2022) stated that the research team tried to measure success in performing a task more directly. Success in this case was defined as satisfying the needs of a human who wants the summary, and that is why human feedback is so important to the technique. The question posed in this study was: “How do we optimize what we *really want* to optimize?” The approach is naturally complex, but it can be summarized in a three-step process, the first two of which required a large number of human contractors to perform human in the loop tasks.

First, a group of humans performed a task to briefly summarize a historical event. The summaries' data set was used subsequently to train a supervised machine learning model to produce these summaries without human help.

The second step was the critical one, and it was the most closely aligned with human in the loop. The humans were involved again, but this time, they provided rating scores on the quality of summaries. This is the step that ensures that future results will be aligned with the needs and expectations of humans.

Finally, a reinforcement learning model was built. The reinforcement constituted assigning a reward when the model performs well or assigning a penalty when it performs poorly over a series of attempts. As one of the researchers described it, their approach worked much the same way as using reinforcement learning to teach computers to compete in video games, but with the human ratings acting like points in a video game.

On numerous evaluation criteria, the new GPT-4 model outperformed the old GPT-3 model. This reinforcement learning methodology is widely credited with why many users find ChatGPT-4 results to be so surprisingly good compared to results of GPT-3. The central and most important element of this work was the body of many hours contributed by human contractors who acted as humans in the loop.

The data annotation ecosystem

An entire cottage industry has arisen to provide support for data annotation projects. When projects like this are planned, there are a number of options available for choosing those who will do the annotation work, four of which are listed below.

Choice #1: assign someone on the modeling team. These people could include data scientists, subject matter experts (SMEs), or some other dedicated in-house resource. This option requires that you develop the skills not only in building the models but also in managing a data annotation project.

Choice #2: Annotation software. There is software that uses AI to assist in creating training data to build an AI model. While it still requires human oversight, this so-called auto annotation can save a great deal of time.

Choice #3: Crowdsourcing. The idea behind crowdsourcing is to pay online users to perform simple tasks such as labeling the type of furniture in the picture or drawing a bounding box around a face. Experience has shown that there are millions of potential data annotators worldwide who participate online to perform these tasks. The most well-known service that provides this support is Amazon's Mechanical Turk.

Choice #4: A managed in-house data annotation workforce. Some companies have a different business model and don't use crowdsourcing. Instead, they hire, train, and manage the data annotators in house. Such a managed in-house workforce may be able to perform more complex tasks than a crowdsourced resource.

Which annotator would engender the greatest confidence to perform tasks in a medical informatics project: a medical student or an in-house data annotator with no medical experience? It is likely that the project manager would be tempted to pick the medical student. However, assume that further staff analysis revealed that the medical student was available to work on the project for only 2 weeks during a hiatus in medical training. Also assume that the data annotator was trained specifically on the task and had worked on the project for the previous 6 months. The final choice might very well be to use the in-house annotator. Therefore the decision of a wise project manager for the selection of annotator staff may be much more complicated than it might appear at the beginning of the selection process.

Consistency is key when it comes to data annotation. While you might want to use SMEs in the early stages of the project, it's not ultimately scalable. The ideal data annotator has the training and the patience to perform that task well many times.

SMEs usually can't do that. In short, the data annotation itself becomes a skill, one that is often unique to the specific aspects of the project. A very common practice is to use the SMEs to design and develop the processes; they then train the data annotators to produce a large data set at a scale necessary to train the AI model. It is important, however, to have feedback loops that include feedback from the data annotators back to the data scientists, feedback from the SMEs, and from the preliminary modeling process back to the data annotation process. What at first may appear to be a low-skilled task becomes, over time, a skill set not only in the project at hand but also one that is generalizable to other data annotation tasks.

The ethics of data annotation

No one has a definitive number, but most who have attempted to estimate agree that the number of data annotators in the world is well into the millions. Despite the fear that AI will eliminate jobs, here is a fairly new kind of job that is growing exponentially in response to AI developments. The controversy surrounds the working conditions for some members of this poorly understood group (*The Wall Street Journal*, 2023).

A high-profile article (*Wallstreet Journal*, 2023) and a book-length treatment (Gray and Suri, 2019) have shed some light on what large-scale data annotation is like. Some of the criticism has been quite harsh, but it's important to remember that conditions vary greatly. Data scientists and data professionals are potential consumers of these services and they should try to learn

about the conditions of the data annotators they might employ, however indirectly.

So why don't we know more about the world of large-scale data annotation? This description from Gray and Suri (2021) provides one possible explanation:

As one crowdwork research team observed, clients using these platforms "expect cheap, 'frictionless' completion of work without oversight, as if the platform were not an interface to human workers but a vast computer without living expenses."

This comment is as focused on "human computation" as it is on data annotation, but it is best to omit some of the details when delegating work to others, especially to a vendor.

The nature of the ethical criticism is focused on three general issues:

- Compensation
- Training and oversight
- Potential harm

Compensation

Some but not all data annotators participate in the gig economy in that they are paid by the task, per image, for instance. Although a nontrivial number of data annotators live in the United States, many live overseas, where the prevailing wage for this work can be under \$2 an hour. One of the criticisms that have been reported is that when structured as gig work, orientation to a new task is not compensated, so there can be a large number of hours that are not compensated, resulting in an even lower overall compensation. Also, since these workers are typically located in different time zones, surveys like the one conducted by Gray and Suri (2021) reveal a pattern where these workers have to monitor their apps all night and rush into action when a new requirement has been posted. They have to rush because once the tasks are complete, the opportunities will be gone.

Training and oversight

In addition to differences in compensation, vendors differ also in how they train and manage their teams. Gray and Suri (2021) mention two in particular: LeadGenius and CloudFactory have different management structures that vary in terms of longer-term commitments to workers and do not pay them by the task. According to Gray and Suri (2021),

"They prioritize workers' schedules, interests, and collaborations. Ultimately, these platforms show how worker-focused design can improve the quality of work produced and workers' experiences of their jobs."

Potential harm

Finally, there is another troubling issue without a clear solution. Whenever family members use an LLM to search for information or they surf the Internet directly, they might find some unsafe content (from an ethical, moral, or legal standpoint). To ensure safety for family members, the best approach for parents is to search for the information themselves to screen out harmful content. In a recent article in *The Wall Street Journal* (2023), “Cleaning Up ChatGPT Takes Heavy Toll on Human Workers,” this aspect was explored in regard to LLMs. It was revealed that some of these workers have experienced sleepiness and depression, and it is likely that they don’t have access to mental health care. Sometimes, these workers don’t even have access to others in a community because they work alone. For this reason, some companies (e.g., CloudFactory) won’t accept work that is likely to be harmful. However, someone has to do this work if troubling material is going to be kept off the Internet and out of our large language model results.

We should have some understanding of the ethical characteristics of the business model of our vendors. Also, we need to maintain those critical feedback loops from the data annotators who potentially spend more time in our data than we do. If these annotators are not anonymous and we are aware of their working conditions, then we can advocate for those vendors who manage this resource in the same way that we would if we managed them directly.

Those new to this kind of modeling are at risk of making two assumptions that might be often incorrect. First, the modeling is the hard part. Second, the data scientists will be able to handle this. The reason these assumptions are risky is that data science is a vast area, and even a very experienced data scientist might not have proper experience overseeing a data annotation.

Issues associated with managing a data annotation project

1. We can start with the issue raised above. *Does anyone on the team have experience doing this?* The answer to this question is complex, and determining this in the midst of a project very often will not work. Seek out help from high-level managers or HR representatives who can help you identify potential resources to fit your needs. Sometimes you may have to consider reaching out to an external resource for help.
2. *How complex is the data annotation task?* Is crowdsourcing an option? If so, is there a resource that you can find that you’re comfortable with in terms of quality, cost, and how their annotators are

trained and compensated? If the task is more complex, will you consider a managed workforce or perhaps training your own team? How much SME is required and at what stages of the project? Will you do some annotation in house, if only to estimate the level of effort to do it at scale?

3. *How will human resources be combined?* Should you establish more than one vendor relationship as a quality control step? Whose software will you be using: theirs, yours, or a software vendor’s? How will the data securely get back and forth?
4. *How will you establish feedback loops?* If you don’t have direct access to the data annotators, how will you get feedback about issues like confusing instructions, or questions about edge cases? How do you involve the SMEs during design but also to review cases?
5. *What is your ongoing plan?* What are your future plans for the long journey of **iteration**, improving the models and incorporating new data when errors or biases are identified in the model’s performance?

Becoming “data centric”

Andrew (2021) discussed the idea that organizations need to be more data-centric. (<https://datacentricai.org/>) According to Ng, traditionally, modelers see their data as fixed, so when they are disappointed in the performance of the model, they try to improve it by improving “the code.” He purposely kept his comments nonspecific so he could refer to the algorithm itself, its parameters, or even the algorithm architecture.

He doesn’t speculate in depth about why this has been a trend, but it could be that for years, modelers have prototyped solutions on famous publicly available training data set like ImageNet. After all, data set like this are time consuming and expensive to produce. What he proposes is that it’s time to reverse this trend. When we are disappointed in a model’s performance, we should invest more time and attention into increasing data size and, particularly, data quality.

In other words, he’s saying that we have to shift from focusing exclusively on our models to focusing more on the kinds of activities we’ve been talking about in this chapter. It is also noteworthy that in traditional machine learning operations, the primary subject of this book, there is a parallel notion. Many experts would suggest that when a model’s performance is lacking, redoubling one’s effort on data preparation and derived variables is more productive than merely attempting more algorithmic experiments. This issue was discussed in [Chapter 10](#) on Feature Selection.

What is Ng’s rationale in the context of data annotation and human in the loop? The most progress in AI

over the last 10 years or so has been made in industries that have a wealth of labeled data, like the big software companies in Silicon Valley, familiar to all of us. But a manufacturer who is trying to detect flawed products coming off an assembly line or an insurer who is looking at photos looking for evidence of fraud doesn't have a huge amount of data. Even if the total amount of data is large, there is not enough data of the rare instances that we are trying to predict, like the flawed product or the fraudulent claims.

Recognizing the importance of Data-Centric AI (DCAI), the Association of Computing Machinery dedicated a recent review article to the topic, *The Principles of Data-Centric AI* (Data Centric AI Hub, 2021). The authors identified six principles of DCAI. These principles underscore the importance of the activities and skills we've discussed in this course.

First: Systematic improvement of data fit. It's not enough to sample and label data at random. You have to systematically sample to ensure that all classes are represented. This is important for rare classes like fraud, but it's also important to combat bias.

Second: Systematic improvement of data consistency. Again, we are reminded that data annotation is a complex skill that can make or break the success of our models. Recognizing this will hopefully encourage effective training and appropriate compensation for data annotators. And also, recognize that managing a large number of data annotators is a challenging task as well.

Third: Mutual improvement of model and data through iteration. As we've seen, feedback is key. As Andrew Ng's original advice suggests, when we are disappointed with the model, iterate back to the training data set and see how we can improve it.

Fourth: Human-centeredness of data work. With this principle, the authors are trying to combat the notion that "if the data is big enough, it will tell everything." Instead, they encourage the idea that "humans are implicated in every step of data work and their choices have consequences for ethical and responsible uses of data in building AI models."

Fifth: AI as a sociotechnical system. Human-centered AI isn't only about the human who performs human-in-the-loop tasks, but also the humans who are the reason the systems are built. We can't focus exclusively on humans like SMEs, data scientists, and data annotators and forget the societal impact of the models we build. We have to think of our models as belonging to a broader collaboration of humans and computers, not only in the development of AI models but in their daily use.

Sixth: Continuous and substantive interactions between AI and domain experts. We've discussed that SMEs are often a critical part of the data annotation design, but because they are not a scalable resource, the data

annotation tends to be handed off to outside teams of data annotators. This principle is a reminder that this interaction can't be only at the beginning of a project; it must be "continuous and substantive" (ACM, 2023).

Postscript

Much of the current hype by analytical tool vendors focuses on automated operations. Some data analytics modeling tasks can benefit from automation (e.g., data cleaning), but human insights are necessary inputs for other operations (e.g., deriving new variables). This chapter has demonstrated that the same thing is true for large AI modeling activities. While our models certainly will improve over time and the size of the sample that needs direct human verification might diminish over time, we will continue to need the human in the loop for decades to come.

Key words and phrases

Human-in-the-loop, annotators, Feedback, Labeled, Data annotation, Deep learning, Large language models, Reinforcement learning, Labeled data, Image data.

References

- ACM, (2023). <https://cacm.acm.org/magazines/2023/8/274940-the-principles-of-data-centric-ai/abstract>.
- Andrew, N. (2021). NeurIPS Data-Centric AI Workshop <https://data-centricai.org/neurips21/>
- Hao, K., & Seetharaman, D. (2023). *The Wall Street Journal*, <https://www.wsj.com/articles/chatgpt-openai-content-abusive-sexually-explicit-harassment-kenya-workers-on-human-workers-cf191483>
- Bloomberg. 2019. Is anyone listening to you? Bloomberg Global Team Reviews-April 10, 2019. (https://www.bloomberg.com/news/articles/2019-04-10/is-anyone-listening-to-you-on-alex-a-global-team-reviews-audio?leadSource=uverify%20wall&in_source=embedded-checkout-banner)
- CBS Times (2023). <https://www.cbsnews.com/video/godfather-of-artificial-intelligence-talks-impact-and-potential-of-new-ai/>
- Data Centric AI Hub (2021): <https://datacentricai.org/>
- Gray, M., & Suri, S. 2019. Ghost work. HarperCollins. Kindle Edition. 288p.
- Gray, M., & Suri, S. (2019). Ghost work: How to stop Silicon Valley from building a new global underclass. Boston: Houghton Mifflin Harcourt.
- Guo, Edward Amazon blog. 2020. <https://www.amazon.science/blog/how-we-taught-alex-a-to-correct-her-own-defects>
- Arize, 2023. Ouyang and Lowe interview. <https://arize.com/blog/openai-on-rlhf/>
- Li, F-F. (2017). ACM Tech Talks, ImageNet, Where have we gone? Where are we going? <https://learning.acm.org/techtalks/ImageNet>
- Lungren, M. (2017). Stanford lecture. AI in Radiology at Stanford: Rise of the Machines. <https://youtu.be/Gigd1rkZTSE?si=sCSd-fxKufMrvCNW>

- Monarch, Robert (2021). *Human-in-the-loop machine learning*. Manning.
- New York Times. 2011. Deciphering Old Texts, One Woozy, Curvy Word at a Time. New York Times. March 28, 2011. <https://www.nytimes.com/2011/03/29/science/29recaptcha.html>
- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., et al. (2022). Training language models to follow instructions with human feedback, *Advances in Neural Information Processing Systems*. 2022. <https://arxiv.org/abs/2203.02155>
- FTC, (2022). <https://consumer.ftc.gov/articles/how-recognize-and-avoid-phishing-scams>
- New York Times (2023). <https://www.nytimes.com/2023/05/01/technology/ai-google-chatbot-engineer-quits-hinton.html?smtyp=cur&smid=li-nytimes>
- SAE International. (2014). Automated driving – levels of driving automation are defined in new SAE international standard J3016.
- Stanford, (2017). <https://aimi.stanford.edu/news/rsna-2017-rads-who-use-ai-will-replace-rads-who-dont-0>
- Karen Hao and Deepa Seetharaman, (2023). Cleaning Up ChatGPT Takes Heavy Toll on Human Workers.

Applications

Part IV of this book contains six application case studies to illustrate how data analytics modeling practices can be implemented in these data domains. This selection of case studies is not intended to be comprehensive or even representative of all examples in any of the data domains. Rather, the selection coincides with the breadth and general focus of research conducted by the authors of this book.

[Chapter 20](#) presents an overview of the practice of data analytics in the data domain of medicine. We include it here as an introduction to some of the problems associated with analytical modeling in this data domain.

[Chapter 21](#) covers the application of the technique of temporal abstractions to model customer demand. This technique permits predictive modeling in historical time-series data sets without resorting to complicated time-series analysis techniques. Temporal abstractions can extract temporal information from historical customer data sets that can be composed into predictor variables to be analyzed by traditional machine learning algorithms (e.g., decision trees and neural networks).

[Chapter 22](#) describes the data domain of learning analytics and focuses on building predictive models of student attrition. The implementation of various interdiction strategies based on results of these models can save educational institutions millions of dollars per year, represented by tuition monies lost when students leave an institution.

[Chapter 23](#) describes how clinical data in the electronic medical records (EMRs) of patients can be used to engineer powerful variables that can be used to predict type 1 precancerous colon polyps. The need for screening programs like this is defined by the fact that the incidence of colon cancer among people younger than age 50 (the age for recommending regular colonoscopies) is increasing at a rate of about 2% per year, and physicians don't know why. This screening program can be used to support the recommendation that a patient have a colonoscopy, regardless of age. Other screening tools are available for predicting the presence of type 1 colon polyps (e.g., the Fecal Immunochemical Test – FIT, and the Fecal Occult Blood Test – FOBT), but they are relatively inconvenient for patients compared with the completely noninvasive colon polyp analytical model. Whatever screening test is used to assign the risk that type 1 colon polyps are present in a patient, the colonoscopy is the only diagnostic tool available for determining the actual presence of type 1 colon polyps.

[Chapter 24](#) explains steps in the development of the FICO credit score and distinguishes it from the Vantage credit score. This chapter focuses on the case study of building a model to predict the credit risk. Both the FICO and the Vantage scores are based on credit risk models. The precise outcome that the case study focuses on is the risk of loan default. This case study will provide the reader with a window into the process used to develop the FICO and Vantage credit scores.

[Chapter 25](#) describes the newest type of analytical model, the Foundation model. The group of much-touted Large Language Models (LLMs) are a type of foundation model, which are trained on billions of online text documents (e.g., articles, eBooks, and blogs). Many of these LLMs are proprietary, but some of them are in the public domain (e. g. ChatGPT-3). Each online document is reduced to a number of small word sequences that mean something in the context of the subject of the document. These word sequences are called “parameters.” The open-source ChatGPT-3 has 175 million parameters, but NVIDIA's MT_NLG model has 530 million parameters. The ability to return comprehensive and correct answers to submitted queries is directly related to the number of parameters in the LLM.

This chapter provides a very brief introduction to the nature and variety of LLMs available today. It provides several short case studies in the form of returns by an LLM for several submitted queries. The reader can research the availability of several LLMs by clicking on the links provided. APIs are available from many of the vendors of LLMs like OpenAI, which can enable integration of LLM returns into another application by making the appropriate call to the LLM source through the API.

Overview of healthcare delivery and medical informatics

Prelude

Healthcare delivery and medical informatics disciplines are changing rapidly. New ways of storing and accessing information are evolving, along with the proliferation of large language models (LLMs). These new AI models are being incorporated into standard internet browsers. The Microsoft Edge browser already includes technology powered by ChatGPT-3. Google's Chrome browser will soon be powered by Bard, the Google LLM. Soon LLMs will replace the traditional SQL technology for many database queries. AI technology will become commonplace in medical sensors and recording systems. Soon AI applications will replace the body of best practice documents by providing front-line recommendations for treatments under a largely expanded option list.

Introduction

This chapter is designed to introduce the promise and problems associated with predictive analytics in healthcare delivery and medical informatics. In this chapter, we will introduce the subjects of healthcare delivery and medical informatics by providing short summaries of the chapter contents of the medical book and a case study describing the use of metabolomic data to build a model of lung cancer incidence.

Chapter 1: The purpose of the book on medical data analytics

There are many books available on the subject of medical data analysis; therefore you might wonder why so many authors spent so much time writing this one. It is not because the authors need books to bolster their academic positions (none of them are academics). All of

the authors are professionals, active either in the practice of medicine or the analysis of medical data. The primary reason for writing the book discussed in this chapter is to serve the great need for analytical information to support patient-directed medicine by showing how to do it and ways to do it right!

Chapter 2: History of predictive analytics in medicine and health care

This chapter traces the development of medical records over the last 7000 years and describes five examples of rulers mandating the gathering and storage of medical information, initially on clay tablets, then printed documents, and lately in digital format. The ancient Greek medical records preserved by Herodotus were destroyed almost completely during the Dark Ages. Before they were lost, copies were translated into Arabic and stored in Arabia for 1000 years. The Arabian works were translated into Latin during the Reformation, beginning with Erasmus in 1525, and remained as the standard in medical treatment until the rise of modern medicine at the beginning of the 20th century. Medical casebooks were written by some 17th- and 18th-century physicians to record medical information for specific patients. Former President Obama and the Affordable Care Act of 2009 mandated the conversion of written medical records (on physicians' bookshelves) to digital format by 2014. Such digital records have become the basis for the development of analytical methods of proactive health care.

Chapter 3: Bioinformatics

Past health care has been primarily reactive. When we are faced with a disease, we treat it; if the pain gets worse, we alleviate it; when someone stops breathing, we resuscitate them. One of the aims of predictive

analytics in health care is to diagnose problems in an early stage of development before they take a toll on the human body, but the role of predictive analytics does not stop here. Another aim of predictive analytics is to guide the selection and tailoring of treatments for individuals by predicting the course of events that is likely to occur with a number of candidate treatment options.

Chapter 4: Data and process models in medical informatics

This chapter lays the foundation for building classification systems for diseases and maladies in medicine. It shows how present classification systems developed in history and practice to form the ICD system used now for insurance and billing purposes. Finally, it describes the OMOP-DM data model used commonly to store clinical data from which analytical models can be built.

Chapter 5: Access to data for analytics – the initial issue in medical and healthcare predictive analytics

How much data does health care really have? Every second, a tremendous amount of healthcare data is being captured and stored in electronic health records in medical offices and from wearable devices. It is estimated that each patient generates about 80 megabytes of data per year. The sum total of all such medical data ranges into petabytes (millions of megabytes). This huge body of data is being mined to gain information (some of it in “real time”) and generate medical insights. It is estimated that today, 30% of the world’s data volume is coming from the healthcare industry, and that by 2025, the annual growth rate of data acquisition for health care will reach 36%. In this chapter, in addition to the size of data in our world, we also look at how difficult it is to obtain really good healthcare data for research, not because it doesn’t exist (it does), but because it is not available due to legal restrictions on personal medical information. Finally, the chapter discusses the benefit of using good data for developing personalized–precision medicine applications in healthcare delivery.

Chapter 6: Precision (personalized) medicine

Currently, personalized medicine has a very broad definition that spans a number of medical fields. A succinct definition is offered as “the tailoring of a treatment to an individual based on their unique characteristics.” The treatment may be a medication, an exercise, a diet, a

surgical procedure, or any other therapeutic measure used within the field of medicine. The treatments may be tailored to the individual by using genetic markers or other biological markers, environmental information, metabolomic and behavioral information, demographic information, or any other information that might define the way the patient responds to the treatment. Using these factors, physicians can determine the best type of treatment, duration of treatment, what dose to give, what kind of followup care is needed, or who is most likely to develop certain side effects. With this information in hand, the appropriate monitoring system can be assigned for each patient. Currently, some research organizations prefer the term “Precision Medicine” in place of “Personalized Medicine.” In this chapter, however, the term precision medicine is considered as synonymous with personalized medicine and as our knowledge increases over time, we can develop treatments catering to the individual as our precision increases.

Chapter 7: Patient-directed health care

The goal of this chapter is to provide information and media for the provision of patient-directed health care for:

- Patients who want to know how to take more control over their own health care;
- Medical residents who might be required to carry out a research study as part of their residency;
- Other interested researchers; and
- Physicians who might wish to understand their patients more and be prepared better to treat them.

Chapter 8: Regulatory measures – agencies and data issues in medicine and health care

This chapter provides a summary discussion of several regulatory agencies that exist to help maintain a consistency in health information and of several statistical quality control systems, including:

- Root-cause analysis
- Six Sigma

Chapter 9: Predictive analytics with multiomics data

The term *multiomics* refers to the combined body of information about human bodily function, including:

- Genomic data – information about the structure and function of the genetic system

- Proteomic data – information about the synthesis and usage of proteins in the body
- Metabolomic data – information about the relationship between certain products of biological metabolism and diseases
- Epigenomic data – information about the interactions among genetic products
- Transcriptomic data – information about various pathways of genetic transcription and effects of their errors

See [Miner et al. \(2023\)](#) for more information on analyzing multiomics data.

Medical science and practice are becoming increasingly data driven. These data are derived from studies of an increasing number of biological processes, generating many different kinds of data. The challenge for today is to develop effective ways to combine those data into compatible data sets that are prepared properly for analysis. This chapter will focus largely on the proper integration and preparation of diverse data sets necessary to support diagnostic analysis and the development of effective treatment protocols. An example is provided for the use of metabolomic data to predict the presence of lung cancer in patients. The model was 100% accurate because the measured levels of several metabolites in cancer patients were much higher and completely disjunct from those of normal patients.

A case study on modeling the presence of lung cancer with metabolomic data is presented in [Chapter 15](#) of this book.

Chapter 10: Artificial intelligence and genomics

Artificial intelligence (AI) is used in many tools that analyze genetic data prior to its direct clinical application. However, there is little direct clinical application of AI with genomics at the time of writing (January/February 2022). Genomics data presents a great opportunity for application of AI while also presenting significant challenges.

Chapter 11: Glaucoma (eye disease): a case study with suggested predictive analytics modeling for identifying an individual patient's best diagnosis and best treatment

This chapter presents the problem with glaucoma and the 12-year treatment history for one of the coauthors of [Miner et al. \(2023\)](#).

This pathway of diagnosis and treatment speaks clearly to the “lack of person-centered” care and directs

the ways needed in the future. Reading this personal history helps to illustrate the urgency of doing everything possible to get medicine and healthcare delivery to this next gold standard of person-centered medicine.

Chapter 12: Practical application example: using data science algorithms in predicting ICU patient urine output in response to diuretics to aid clinicians and healthcare providers in clinical decision making

This study was originally started based on research done for a customer in the healthcare space to assess the feasibility of usage of an algorithm to shorten the duration of time it takes to determine if a patient's urine output (UO) goals will be met (with or without diuretics) by a physician or other healthcare provider. This was intended to support diuresis for resuscitation of patients in the ICU by making it easy to initiate and monitor best practices for diuretic dosing, UO goal setting, and UO monitoring with ICU care teams.

Chapter 13: Practical application example: prediction tool development

This chapter describes the creation and adoption of robust predictive model metrics at the bedside for greatly benefiting the patient, like preterm infants at risk of bronchopulmonary dysplasia, using Shiny-R.

Chapter 14: Modeling precancerous colon polyps with OMOP data

The purpose of this chapter is to demonstrate the set of data preparation and modeling operations necessary to predict the presence of precancerous colon polyps using only common demographic data and clinical data (no images), with no data from special tests (e.g., FIT or FOBT). Age was not used as a predictive factor for two reasons: (1) because all of the patients in the data set were preselected for age in that they were older than the guideline age (50) for having a regular colonoscopy; and (2) because this study was performed to develop a screening tool for use for patients of any age who have not had a prior colonoscopy. This chapter shows how to build a predictive model directly on data available in normal patient files (e.g., birth date) or in the EPIC database (in OMOP format) at the University of California at Irvine Medical Center.

Chapter 15: Prediction of pancreatic and lung cancer from metabolomics data

This chapter is designed to present an overview of the modeling process in KNIME for modeling the presence of lung cancer or pancreatic cancer in patients from gas chromatograph-mass spectrometer (GCMS) analysis of their blood serum. The purpose of each node in the modeling workflow will be discussed but not documented in the form of a tutorial. This chapter represents a summary of the study only, with emphasis on the data preparation operations necessary to generate the modeling data set. The workflow built in this study is not available on the companion web page for experienced KNIME modelers to use because it uses data that include confidential information. However, the “case study” presented in this chapter allows readers to understand the overall process.

Case study on modeling the presence of lung cancer with metabolomic data

Metabolites have been defined as molecules less than 1500 daltons molecular weight which are produced endogenously by cellular metabolism (Fiehn, 2002). Cells altered by cancer or disease produce additional metabolites or significantly higher levels of existing metabolites. The quantitative analysis of these metabolites can be related to the presence/absence of cancer or disease in a patient.

After cancer and disease metabolites are emitted, they dissolve into organic fluids in the cell and bloodstream. Many of these volatile organic compounds (VOCs) are

subject to being ionized prior to being chromatographed in a GCMS. The number of ions in these VOCs can be counted in representative absorption peaks and related to normal and abnormal (diseased or cancerous) conditions. Specific compounds from which the ions are derived can be identified from standard tables. The presence of significantly higher than normal ion counts for a compound from a diseased or cancerous subject can tag them as markers for the presence of cancer. Various of these marker compounds have been used as predictors of cancer presence for lung cancer, breast cancer, liver cancer, pancreatic cancer, and colon cancer.

VOCs have been used to predict the presence of cancer with high accuracy in hundreds of studies during the last 10 years. For example, Nisbet (2023) analyzed metabolites in blood plasma from a group of lung cancer, pancreatic cancer, and normal patients. A GCMS system was used to count the number of ions of various molecular weights flowing through the system. Output data were preprocessed with a Fast-Fourier Transform (FFT) to convert data from the time domain to the frequency domain. A Python program was used to further preprocess the GCMS output data to generate the data sets for building the analytical models in the KNIME Analytical Platform. All data rows were randomized, and data values were normalized with a z-score function. Missing values were imputed appropriately, and the data set input to the Random Forest modeling algorithm was balanced with the SMOTE technique. Finally, important features were selected iteratively with a backward feature elimination technique using a Random Forest modeling algorithm. Fig. 20.1 shows the KNIME workflow used for developing the multicategorical model.

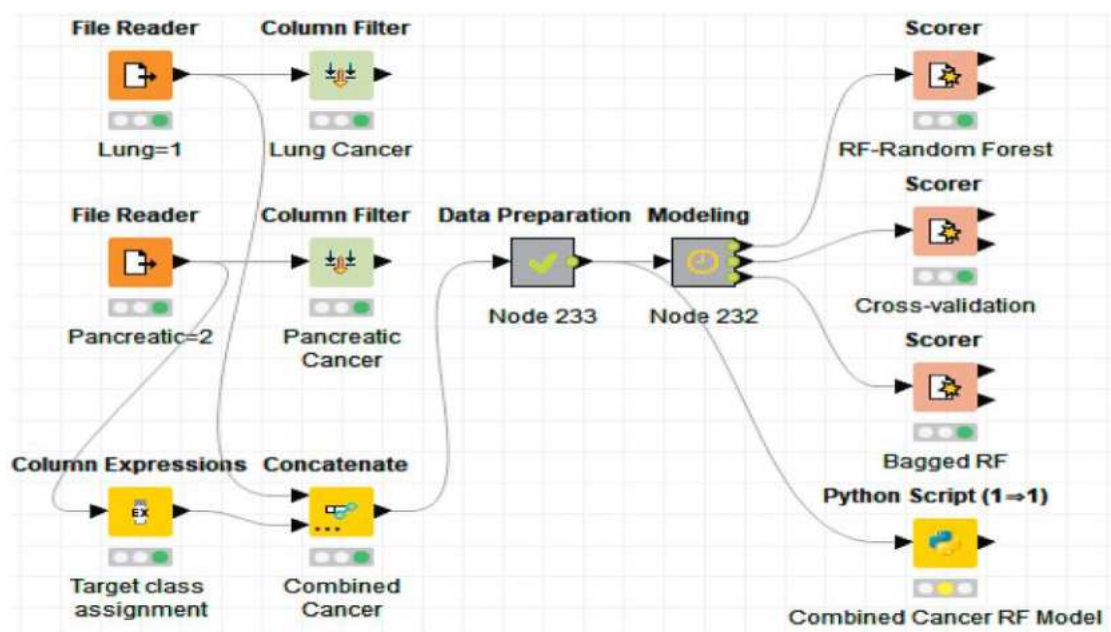


FIG. 20.1 KNIME workflow for developing random forest models with both KNIME and Python.

Results showed a perfect model to predict the correct cancer state of each row. How could this be? Past experience has shown that perfect models are usually the result of either mistakes in algorithm specification or variable definition. For example, the most common cause of a perfect model is using a target variable that is in some way related to one of the predictor variables *by definition*! This error generates a “leak from the future” (Nisbet et al., 2009).¹ These perfect models were not the results of leaks from the future, but rather, because the ion counts of specific metabolites for both groups of cancer patients range completely above those of normal patients. The modeling algorithm had no trouble producing a perfect model. Scouring of the literature uncovered four other cases where modeling of VOCs for cancer prediction produced perfect models (see Gordon et al., 1985; Westhoff et al., 2009; Turki and Wen, 2018; Qian et al., 2018). This example is presented as a full tutorial on the book web page.

Chapter 16: COVID-19 descriptive analytics visualization of pandemic and hospitalization data

This chapter will introduce readers to the kinds of data visualizations that can be made from a data set. From these data visualizations, readers can form some important insights about the nature of the data sets, even before the analysis begins. Based on these insights, analytical models can be designed to generate other insights valuable for understanding various phenomena in the data set.

Chapter 17: Disseminated intravascular coagulation predictive analytics with pediatric ICU admissions

This case study outlines the procedures we used to analyze the data for 11,569 pediatric patients at the Loma Linda University Medical Center. These children were admitted between January 1, 2009 and December 31, 2015 with a diagnosis of disseminated intravascular coagulation. An analytical model was developed to predict survival or death, given the measurements of various tests upon admission. The goal was to identify a smaller number of lab tests (variables) that could be used to predict survival or death. This case study also describes the processes used to analyze the data.

¹See Chapter 20 by John Elder.

Chapter 18: Challenges for healthcare administration and delivery: integrating predictive and prescriptive modeling into personalized precision health care

We face many challenges as we integrate predictive modeling into personalized precision health care. Some of these challenges raise questions that cannot be answered currently. Nevertheless, we need to consider them and strategize to develop methods to find answers.

Chapter 19: Challenges of medical research in incorporating modern data analytics in studies

We face many challenges as we design research to answer the medical questions facing us both now and in the future. Some of these challenges raise questions for which we don’t have answers at present. But we must continue to pursue the main goals: discovering and developing pathways of knowledge to make accurate diagnoses for the individual patient the first time presented and prescribing the most appropriate treatment. In this way, we give a patient the *right* diagnosis, and the *right* diagnosis the *first* time is the *right* way to practice precision medicine in the future.

Chapter 20: The nature of insight from data and implications for automated decisioning

The purpose of this chapter is to explore how some modern AI and machine learning (ML) technologies can facilitate the generation of new insights and inform better decisions in health care. More specifically, this chapter focuses on the nature of insights, causal inference, and explainability. It is widely understood that correlation does not imply causation. AI and ML methods are (almost always) applied to historical data in order to detect “relationships,” expressed as correlations. One of the challenges of some AI/ML models is that they are not very explainable (e.g., neural networks). The challenge is to make these methods explainable as far as possible. The first aspect is that much of acquired human expertise is acquired through interaction with complex historical and correlational data (e.g., the training of a surgeon). The second aspect described is the use of methods and algorithms now in common use to explain results and predictions from AI/ML models.

Chapter 21: Model management and ModelOps: managing an artificial intelligence-driven enterprise

It is becoming increasingly clear in almost all organizations that success in terms of effectiveness, the ability to compete, or profitability of enterprises in the future can be achieved only by incorporating the use of advanced analytical methods involving AI/ML methods. These methods are necessary in order to be effective and any successful organization must embrace ML, AI, advanced analytics, AI-infused applications, and automation. In practically all applications and domains, deep insight and accurate foresight are necessary to be effective and competitive, and also to adapt rapidly to changing and increasingly competitive environments. This is also true for healthcare providers, insurers, and practitioners. This chapter explores these issues from the perspective of Model Management and Operations (ModelOps). ModelOps is required tooling to achieve efficient and agile model deployment, monitoring, and recalibration. Any organization deploying advanced analytics and AI to optimize their services, processes, or products must implement demonstrated best practices with respect to the model-deployment life cycle, governance, monitoring, and risk management.

Chapter 22: Forecasts for advances in predictive and prescriptive analytics

Descriptive analytics presents data about what happened in the past in either textual or graphical form. Predictive analytics builds mathematical models based on past data to predict what might happen in the future and associated probabilities of occurrence (risk). Prescriptive analytics goes beyond predicting what will happen with associated risks of occurrence and provides some guidance for deciding what can be done about it.

Chapter 23: Sampling and data analysis

There are serious, real issues of measurements and modeling in health care. What is needed is a “decisioning system” that clearly says “Do This OR Do That,” depending on certain conditions, especially for situations where the variability in data may be much greater than the isolated selected data sampling points (e.g., pulse and blood pressure). We need to use measures that reflect reality expressed in terms other than those describing specific points in time. Such measures can include averages over a period of time, the amount of kurtosis, and the standard deviation. Using these three statistics

in concert may be a better indication of the central tendency within the range and thus lead to more accurate diagnoses and treatments.

Chapter 24: Analytics architectures for the 21st century

AI and analytics efforts are too often just extensions or isolated additions to medical practice and research. Often, they are just an extension of research efforts, part of a medical degree curriculum or residency program to extend scholarly pursuits. They could be additions to hospital systems wishing to gain operational efficiency and control. Or they could be just terminal activities by physician groups of practice trying to mitigate risk or improve financial performance. While isolated efforts may provide some benefit, leaders must understand that exceptional results require a paradigm shift in thinking about technologies and innovation in an integrative context and designing the infrastructure to support them.

Chapter 25 Predictive models versus prescriptive models; causal inference and Bayesian networks

Predictive models and prescriptive models in medicine are compelling technologies which are changing the way health care is delivered around the globe. Examples range from clinical to operational improvements. These include the power to predict the likelihood of a disease, readmission to the hospital, and the correct staffing and resource levels in a clinical setting. Additionally, these include the power to prescribe the optimal clinical therapy and, more importantly for person-centered medicine, the power to prescribe the best therapy for this patient rather than what is judged to be best for a general population. These advanced analytics methodologies are indeed changing the way medicine is practiced. Medicine is becoming less a “practice” and more of a “science.” While this is great news for medicine, misapplication of these models can lead to undesirable results. The formulation and application of predictive and prescriptive analytics require different philosophical and mathematical assumptions. This chapter is focused on revealing different model assumptions – especially the difference between predictive and prescriptive models.

Chapter 26: The future: 21st-century health care and wellness in the digital age

This chapter attempts to summarize much of the preceding book and to focus on the future. It was predicted

in the first edition of the book that dramatic changes and improvements in the field of health care and wellness within the next 6 years: a new 21st-century high-technology health care and wellness movement would shake the existing health care industry and institutions. The second edition of the book describes which predictions of the first edition have actually become reality and what may happen in the future.

Postscript

The chapter summaries of the book *Practical Data Analytics for Innovation in Medicine* provide a template to guide the development of AI applications to other disciplines. The case studies presented in [Chapters 21](#) to [Chapter 24](#) provide examples of past applications of predictive analytics in a few disciplines. Extensions of these case studies could include the incorporation of LLM technology to provide a much richer background upon which to build new applications, similar to the chapter summaries of the medical book.

Keywords and phrases

Clinical, health care, Patient, Medicine, Treatment, Metabolomic, AI, Medical data, Lung cancer, Predictive analytics, Prescriptive analytics.

References

- Fiehn, O. (2002). Metabolomics – the link between genotypes and phenotypes. *Plant Mol. Biol.*, 48, 155–171.
- Gordon, S. M., Szldon, J. P., Krotoszynski, B. K., Gibbons, R. D., & O'Neill, H. J. (1985). Volatile organic compounds in exhaled air from patients with lung cancer. *Clin. Chem.*, 31(8), 1278–1282.
- Miner, G., Miner, L., Burk, S., Goldstein, M., Nisbet, R., Walton, N., & Hill, T. (2023). *Practical data analytics for innovation in medicine*. San Diego, CA: Academic Press.
- Nisbet, R. G., Miner, & Elder, J. (2009). *Handbook of statistical analysis and data mining applications* (pp. 864). San Diego, CA: Academic Press.
- Nisbet, R. (2023). Prediction of pancreatic and lung cancer from metabolomics data. In Miner, L., et al., (Ed.), *Practical data analytics for innovation in medicine*. San Diego, CA: Academic Press. Chapter 15.
- Qian, K., Wang, Y., Hua, L., Chen, A., & Zhang, Y. (2018). New method of lung cancer detection by saliva test using surface-enhanced Raman spectroscopy. *Thoracic Cancer*, 9, 1556–1561.
- Turki, T., & Wei, Z. (2018). Boosting support vector machines for cancer discrimination tasks. *Computers in Biology and Medicine*, 101(2018), 236–249.
- Westhoff, M., Litterst, P., Freitag, L., Urfer, W., Bader, S., & Baumbach, J. (2009). Ion mobility spectrometry for the detection of volatile organic compounds in exhaled breath of patients with lung cancer: results of a pilot study. *Thorax*, 64, 744–748.

Customer response modeling with temporal abstractions

Preamble

Most organizations, whether for profit or nonprofit purposes, exist to develop and promote some things or ideas related to their organization. One of the major activities of these organizations is to appeal to people outside their organizations to join them, support them, or purchase their goods or services. Traditional means of doing this included offering goods and services in storefronts, by advertisements in appropriate venues, and by contacting a broad spectrum of people by phone or mail. These methods are rather passive. The philosophy was to build it, show it, advertise or promote it, and customers would come.

Since the early 1990s, many businesses have taken a more active approach by using various technological approaches to identify specific prospective customers and going after their business, rather than waiting for them to respond to the passive appeals. The key issue in this process is identifying *which* prospects are most likely to respond to the appeals. The activity of identifying prospects and quantifying their likelihood to respond is one of the earliest applications of data mining technology to business.

Early CRM issues in business

One of the early business issues to be addressed with data mining technology in the mid-1990s was customer relationship management (CRM). CRM systems were built to manage how a business relates to its customers. Customer-facing systems were built to manage call centers and to inform marketing and sales efforts. In support of the marketing and sales channels, analytical modeling systems were built by pioneers in data mining technology. National Cash Register (NCR) built some of the earliest analytical CRM product suites in 1998 in the form of *ChurnSentry* (for customer retention modeling) and *GrowthAdvisor* (for cross-sell and up-sell modeling). Both of these products included a data

discovery tool, a model manager, numerous canned reports (via Cognos), and a campaign management system. Soon, other CRM systems were built, notably by Siebel, Vantive, and Salesforce.com, to serve sales force automation, but they later were extended to cover call centers and some front-end office operations.

On the analytical side of CRM, the major foci were

- Customer response modeling with predictive analytics for customer acquisition
- Customer retention
- Customer up-sell (selling an enhanced product or service)
- Customer cross-sell (selling a different product or service)
- Customer Lifetime Value (LTV) modeling

The trend in marketing with analytics was to move from a broadcast marketing operation to a one-to-one marketing operation. Naturally, the key in this activity was predicting which products or services a particular customer was likely to respond to. The most common approach used to do this was to model customer actions in the past and use the model to predict actions in the future. This is a form of human behavioral modeling.

The following are the five popular CRM systems:

Monday Sales CRM (<https://monday.com>)
Pipedrive (<https://www.pipedrive.com>)
Salesforce.com (<https://www.salesforce.com/>)
HubSpot (<https://www.hubspot.com/>)
Oracle NETSUITE (<https://6262239.extforms.netsuite.com>)

Knowing how customers behaved before they acted

To be competitive in today's markets, we must capture and leverage information from historical detail records describing what our customers did in the past. This information can be used to predict customer demand for goods

and services (e.g., up-sell and cross-sell models). But a customer has to be a customer (and remain so) before any purchases can be made. It may cost a company 10 times the model to attract new high-value customers than it costs to keep the ones they have. Customer historical information can be very useful in defining patterns in the behavior of customers leading up to the decision to leave, the company. For a given customer, the decision to leave the company did not happen in a vacuum. Many factors, such as dissatisfaction with service, perception of the greater value of competitive goods and services, and changes in business needs, contributed to this decision. Some of these factors, such as customer satisfaction, can be tracked through customer care programs. However, most factors that contribute directly to attrition cannot be captured and stored in corporate databases. The only way to reflect these attrition variables is to relate them to customer behavior patterns that can be tracked from data in the data warehouse. The pattern of historical information of customers who have left the company can be used to predict which present customers have a high probability of leaving in the near future. How is this possible?

Transforming corporations into business ecosystems: The path to customer fulfillment

Ever since the Industrial Revolution, Western society has tended to view the world as a *machine*, composed of components that functioned like cogs, wheels, and springs.

Newton formalized this approach in science. However, it worked only within the range of Newton's instruments. Later discoveries by Einstein (relativity) and quantum physicists caused the Newtonian concept of the world to fall to pieces!

Businesses also picked up on this metaphor in the Industrial Revolution. The automobile assembly line of Henry Ford was viewed as the paragon of efficiency. As long as the product (a car in this case) was relatively simple in organization, this metaphor appeared to work. An efficient business became defined in terms of:

- A "well-oiled machine";
- "Having momentum";
- "Gaining steam";
- "Firing on all eight cylinders."

The primary business unit became the *corporation*. The prevailing attitude was "Us against Them." Only the strong competitors survived. For these corporations, the primary business activity was *production*. It was expected that revenue would be maximized as production was optimized. Generations of operations research practitioners sought to optimize processes that would maximize business revenue.

With the advent of fast computers, flexible communications, and the Internet, a new business paradigm has emerged: the *business ecosystem* (Inmon et al., 1998). Moore (1996) maintains that real competition in these business ecosystems is not dead (actually, it is intensifying); it has just changed its expression. The old expression of competition pitted offers and markets against each other. The products improved as companies listened to customers and made the products fit their desires. The problem with this approach is that it ignores the *environment* and the *system* in which those offers and markets are embedded. It also ignores the great benefit that can come with co-evolution with other "competitors" to satisfy customers more than if they operated separately ("co-opetition"). Moore stresses the importance of the environment and the system in which our businesses are enmeshed. This emphasis points also to the need to consider systems' effects in our analyses of customer behavior.

As businesses became more complex, the machine metaphor began to break down. In both science and business, it became increasingly obvious by the 1980s that we had to begin to look at the world in a different way. In these increasingly complex systems, there seemed to be important properties that did not emerge until the system was complete and operating as a whole. These *emergent* properties often controlled the major responses of the system. These influences are causing a profound shift in science and business toward viewing the world as an *organism*!

Petzinger (1999) remarks that the key characteristic of modern civilization is that of *economizing* and that our genes are programmed for business.

This view of business as an "organism" flowed out of the central concept proposed by Rothschild (1990):

There is a parallel between the response of natural systems to rapid environmental change and the response of business systems to rapid technological change. (p. xiii)

From this principle, it is argued that our view of the world-as-machine greatly hinders us from economizing very well in this age of rapid technological change. Why? Because the rules keep changing faster than our machinelike business systems can accommodate. Rothschild (1990) believed that it was time for a "new" science to help us understand life amid rapid change. In the intervening time, this new science (data science) has risen to prominence in the management of virtually all sectors of business in the world.

CRM modeling in business ecosystems

In the freewheeling business of today, companies try to build customer relationship management programs that aim to create the same kinds of relationships with

their customers. To build these relationships, companies must learn to understand their customers. To understand their customers sufficiently to build effective customer relationships, marketers must:

- Learn how to identify the right set of customers to do business with (segmentation);
- Learn how to identify valued customers;
- Learn how to recognize danger signals in their data relating to customer behavior that, if unchecked, might lead to decisions to leave the company;
- Use segments defined by attrition probability algorithms to strengthen and maintain relationships with valued members of the existing customer base.

The key principle in this approach is that the most powerful predictors of customer behavior in the future are customer behavior patterns in the past. Other customer characteristics are important also in defining patterns of customer behavior (i.e., demographic and firmographic information). However, unless we include in our models of customer behavior the patterns of past customer behavior related to their future actions, they will not be very powerful predictors of what customers actually do. When these patterns are combined with the more static customer information gathered by businesses in their day-to-day operations (e.g., the date a company started business), companies can take a quantum leap forward in understanding the customer and improving customer loyalty.

Differences between static measures and historical measures

The key difference between static measures of customer response and historical behavior patterns is that historical patterns enable us to track the *development* of the decision to leave, rather than just the decision itself. These evolving behavior patterns are very organic in nature and are driven by a number of significant nonlinear events (NLEs). Farrel (1998) maintains that bursts of customer demand (or “anti-demand,” like attrition) are driven by these NLEs. The evolutionary nature of these NLEs renders them much richer in predictive value than static characteristics alone because they can capture the mood preferences, and attitude of the customer, attitudes, and many clues that help you to understand why the customer did what they did. Some static characteristics are certainly related to the attrition decision, but they tell only a part of the story. To see the other half of the story, we must add variables that express this development of the decision to leave the company. This is a very organic view of customer behavior, similar to the way biologists view the complementary effects of intrinsic (organism-based) and extrinsic (environmental) influences on organism response. This viewpoint

represents a dramatic shift in mindset from the traditional way that many companies view their data.

How can human nature help us in modeling customer response?

If human nature is a common basis for human action, then, we must model human nature to predict the action of customer response. We must focus on variables available to us in our databases that reflect some aspect of human nature that leads to the response. These variables might include:

- Historical customer care data;
- Historical use of company services;
- Historical billing revenue data;
- Historical contract data;
- Selected demographic data.

How can we reorganize our data to reflect motives and attitudes?

The key to successful customer response modeling is to associate with each customer a historical time-series of fields (selected from those listed in the preceding section) that in some way *reflect* motives and attitudes that *caused* the customer decision. These motives and attitudes flowing out of our human nature are the reality behind the “shadows” of the action. Plato described this situation in his Allegory of the Cave in Book VII of *The Republic* (Rowe, 2012). Plato poses this allegory in the form of a dialog between Socrates and Plato’s older brother Gaucon. Socrates tells Glaucon that people who rely solely on their senses are like a group of prisoners who have spent their entire lives chained inside a cave, facing the blank back wall and unable to turn around. All they can see are the shadows projected onto the wall by things passing in front of the cave entrance. These shadows of things are their only reality.

To see the deeper reality of what causes these shadows, we must turn around, so to speak (like the prisoners in Plato’s cave), and look at data in a different way. To accomplish this task, we must abstract information from the time-series of customer responses in a form that is related to the customer action to be modeled. These abstractions are called temporal abstractions, or “lag” variables, because the effects of these variables on the response variable appear to lag one to several time periods behind the causes.

The use of lag variables has attracted widespread interest in medical and pharmaceutical informatics for predicting patient responses (Kahn et al., 1991; Haimowitz and Kohane, 1996; Kattan et al., 1997). Lag variables (aka temporal abstractions) are one type of data abstraction used to map data elements to some context environment.

Data abstractions can be classified into four groups (Lavrac et al., 2000):

- *Qualitative abstraction*: A numeric expression is mapped to a qualitative expression. For example, in an analysis of teenage customer demand compared with that of others, customers with ages between 13 and 19 could be abstracted as a value of 1 to a variable “teenager,” while others are abstracted to a value of 0.
- *Generalization abstraction*: An instance of an occurrence is mapped to its class. For example, in an analysis of Asian preferences compared with non-Asian, listings of “Chinese,” “Japanese,” and “Korean” in the Race variable could be abstracted to 1 in the Asian variable, while others are abstracted to a value of 0.
- *Definitional abstraction*: One data element from one conceptual category is mapped to its counterpart in another conceptual category. For example, when combining data sets from different sources for an analysis of customer demand among African-Americans, you might want to map “Caucasian” in a demographic data set and “White Anglo-Saxon Protestant” in a sociological data set to a separate variable of “Non-Black.”
- *Temporal abstraction (or Lag variable)*: A variable in a time domain with one reference is mapped to a time domain with a different reference. The response appears to “lag” behind the causal variables.

The first three types of data abstractions are usually referred to by analytical modelers as forms of *recoding*. The fourth type, temporal abstraction, is not commonly used. However, the methodologies of several analytical tool vendors have (or had) forms of lag variables integrated into their design:

- *SAS Enterprise Miner*: Ability to define “lag” variables
- *ChurnSentry* and *GrowthAdvisor* by NCR
- *KXEN*: Knowledge Extraction Engine, now SAP Analytics
- *Statistica Data Miner* from Tibco
- *KNIME*: Open Source (Konstanz Information Miner)

What is a lag variable?

Modeling customer behavior with lag variables involves rearranging all the modeling variables to reflect more clearly patterns of change in the customer response variable. Then the modeling tool can easily recognize the pattern that exists between the response variable and various states of predictor variables in the past with respect to the response variable. These time-series representations of each predictor variable are a form of lag variables. See Fig. 21.1.

Fig. 21.1 displays fields in six customer records lined up like beads on an abacus. The data on the left abacus represent information stored for one data element (e.g., number

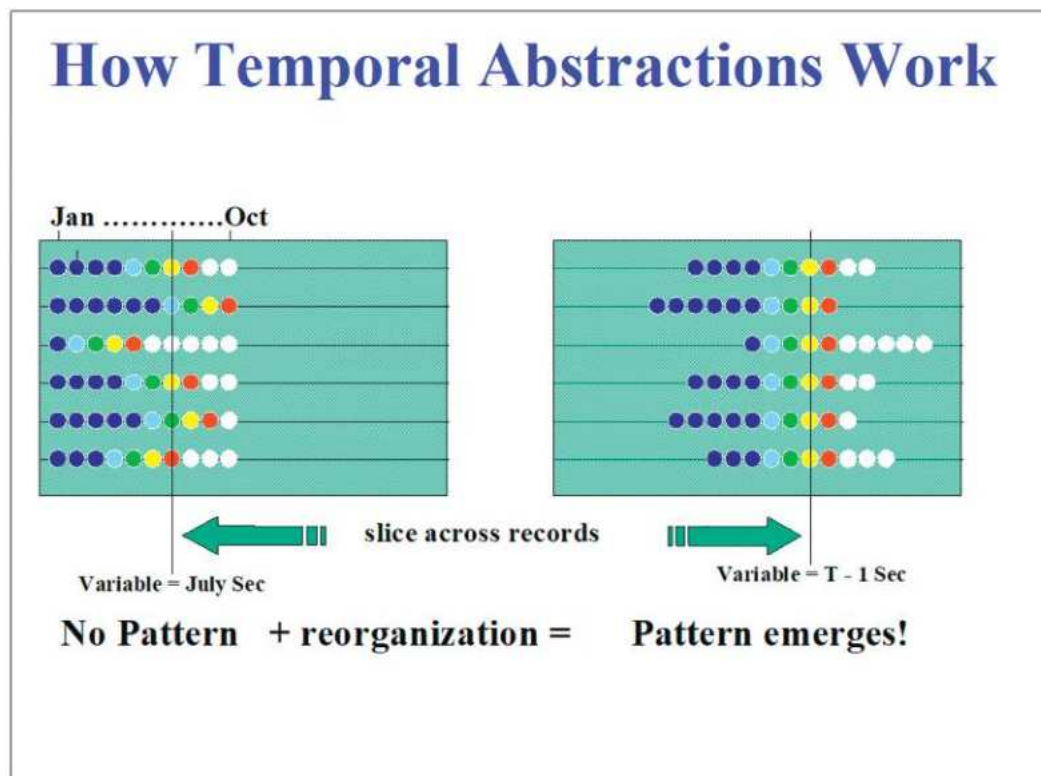


FIG. 21.1 Pattern emergence facilitated by a temporal abstraction.

of bank deposits in a given month) aggregated from multiple records in the database. This arrangement is similar to the format of the data extracted from databases into flat files to be submitted to the modeling tool for analysis. In the default configuration (left abacus), the yellow beads (a given state in the time-series) are scattered all over the abacus. The diagram on the right of Fig. 21.1 shows the rearranged data. Now the yellow beads are lined up. The pattern emerges to the physical sense of our vision and likewise to the mathematical senses of the modeling tool.

The same approach can be used to model customer fraud or propensity-to-buy to serve cross-selling and up-selling campaigns. In Chapter 1, we showed that we must include both Aristotelian and Platonic approaches to truth to model a complex system. Customer behaviors in the context of the business ecosystems within which they operate can be modeled successfully using this combined approach. This approach to customer behavior modeling will permit us to see the “shadows” of customer behavior (following Aristotle) and reflections of the causes (the deeper reality) of this behavior (following Plato). Such a combined perspective on the nature of customer behavior can provide much more powerful models than those based on one perspective alone.

Modeling with different sets of predictor variables

Three examples will be shown: (1) modeling with static variables only, (2) modeling with all variables, and (3) modeling with lag variables only.

Deriving lag variables

Deriving lag variables from a detailed historical data set aggregated initially to a given time grain can be performed either in SQL or by a data manipulation tool that contains a lag variable derivation function. The KNIME Analytics Platform was selected as the data manipulation and analytical modeling tool to derive lag variables and use them to build analytical models of anonymized banking data.

Fig. 21.2 shows the configuration screen for a KNIME Lag Column node used to derive three lag variables for a group of predictor variables.

The number in the Lag box shown in Fig. 21.2 is the number of sequential lag variables to be derived with a Lag interval of 1. This configuration will derive lag variables Num_Credits-1, Num_Credits-2, and Num_Credits-3.

The lag variable data were created by taking weekly snapshots of anonymized banking records for a given customer. These snapshots represent temporal objects in a temporal database (Jensen et al., 1996). The lag variables represent keys of this derived temporal database in which

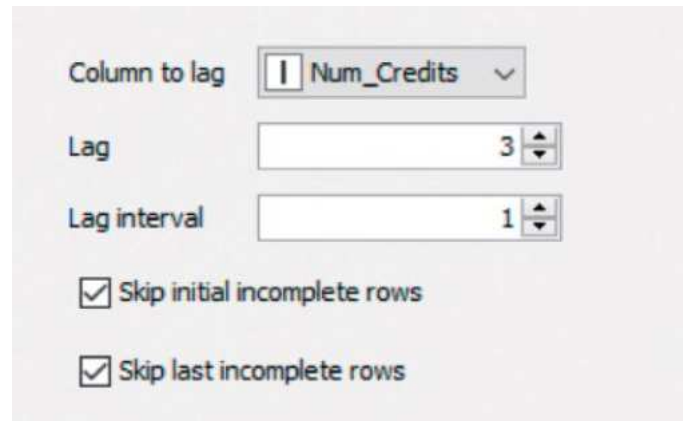


FIG. 21.2 The configuration screen of the KNIME Lag Column node used to derive three lag variables for the Num_Credits banking variable.

the temporal tuples are the response week (week of customer churn) and the state of various banking variables (e.g., number of deposits) 1, 2, and 3 weeks in the past. These snapshot lag variables follow Snapshot Dependency Theory as extended by Wijzen et al. (1993) and formalized by Wijzen (2001), and they represent keys for a sequence of snapshot relations in the customer banking history indexed in reference to the response week.

Three KNIME modeling workflows were built and trained on different sets of predictor variables:

- (1) *modeling with static variables (e.g., Age) and lag variables;*
- (2) *modeling with only static variables; and*
- (3) *modeling with only lag variables.*

Fig. 21.3 shows the KNIME workflow for building the static-only model.

Two other workflows similar to the static model workflow shown in Fig. 21.3 build models using all variables for one model and using lag variables only for the other model. The set of variables for each of the models is set in the green Filtering metanode (a set of nodes). Notice that Fig. 21.3 uses two algorithms, Random Forest and Multilevel Perceptron – MLP (a neural network) to build models in each workflow. These workflows and the anonymized data set used to train them are available on the book companion web page.

Modeling with static variables only

Accuracy results for the models that use static variables only as predictors are shown in Tables 21.1 and 21.2.

Modeling with all predictor variables

Accuracy results for the models that use all variables as predictors are shown in Tables 21.3 and 21.4.

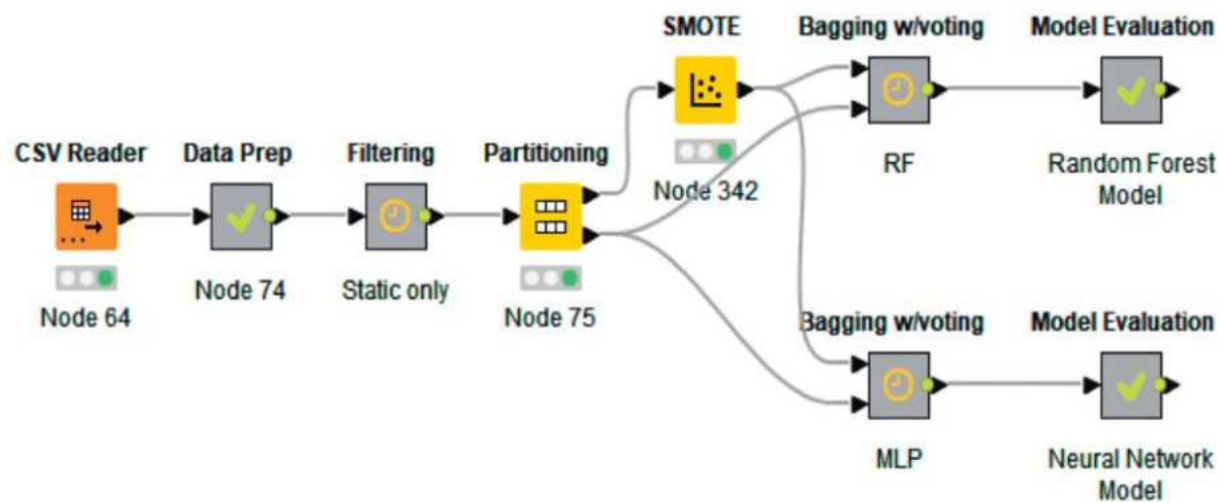


FIG. 21.3 KNIME workflow to build the churn model using only static variables.

TABLE 21.1 Prediction accuracy results of the Random Forest model using static predictor variables only.

S Metric	S Value
Sensitivity	0.252
Specificity	0.969
Accuracy	0.955
AUC	0.839

TABLE 21.3 Prediction accuracy results of the Random Forest model using all predictor variables.

S Metric	S Value
Sensitivity	0.575
Specificity	0.899
Accuracy	0.892
AUC	0.892

TABLE 21.2 Prediction accuracy results of the MLP (neural network) model using static predictor variables only.

S Metric	S Value
Sensitivity	0.838
Specificity	0.424
Accuracy	0.432
AUC	0.705

TABLE 21.4 Prediction accuracy results of the MLP-neural network model using all predictor variables.

S Metric	S Value
Sensitivity	0.821
Specificity	0.725
Accuracy	0.727
AUC	0.837

Notice that neither algorithm performs well for both Sensitivity (for Churn Flag = 1) nor for Specificity (for Churn Flag = 0).

TABLE 21.5 Prediction accuracy results of the Random Forest model using lag variables only as predictor variables.

S Metric	S Value
Sensitivity	0.702
Specificity	0.83
Accuracy	0.828
AUC	0.844

TABLE 21.6 Prediction accuracy results of the MLP-neural network model using lag variables only as predictor variables.

S Metric	S Value
Sensitivity	0.783
Specificity	0.775
Accuracy	0.775
AUC	0.846

Notice in Tables 21.3 and 21.4 that the prediction accuracies are much higher than for the model that uses only static predictor variables. Also notice that Table 21.4 shows that the MLP-neural network model is much more balanced in its performance as measured by Sensitivity and Specificity.

Modeling with lag variables only

Accuracy results for the models that use lag variables only as predictors are shown in Tables 21.5 and 21.6.

Both the Random Forest model results shown in Table 21.5 and the MLP-neural network algorithm perform even better than for the previous models, even though the static variables were eliminated.

Feature importance values

Table 21.7 shows the important predictor variables for the model built with all variables to illustrate the relative predictive power of static variables versus lag variables.

TABLE 21.7 Feature importance values for the Random Forest model using all variables as predictors.

S RF_Feature	I RF_Val
CreditNum-2	15
Credit_Num-1	11
Cons_Chk_Num-1	10
NSF_Amt-3	10
CreditNum-3	9
Last_credit	6
Ln_Acct_Num-1	4
DebitAmt-2	4
DebitAmt-3	4
Children_U	3
DebitAmt-1	3
NSF_Amt-1	3
U_MaritalStatus	3
Program_num	2
Debit_Num_Trend	2
Bill_pay	2
CreditAmt-2	2
DebitNum-1	2
Nearest_Branch_Disti	1
ProbableHomeowne_H	1
CreditAmt-1	1
Fee_Amt-3	1
M_Dwelling	1

Discussion

The biological metaphor appears superior to the machine metaphor in helping us understand the causes of customer behavior in business. Customers are biological entities that respond in a biological manner. This manner is the result of the complex interaction of many factors that operate as a system to influence the nature of the response. It seems reasonable to expect that the only way to create highly predictive models of customer behavior is to express some degree of this complex interaction in the design of the modeling methodology.

From this perspective, it is easy to see why static variables alone will detect only part of the signal of customer response. Most attempts at modeling customer response are confined to analyzing historical variables and their transforms as if they are time-independent influences on customer response. Occasionally, time-series analysis is applied to capture time dependencies. However, time-series analyses (like all parametric methods, as represented in Chapter 1)

suffer from many assumptions that are not satisfied in business data (e.g., linear additivity, variable independence, membership in a specific distribution such as a normal distribution). Therefore many analyses of historical data treat the data elements as if they were static variables with no temporal attributes. For example, the number of deposits by a customer for March, April, and May are part of a time-series of historical data for a given customer, but they are usually submitted to the modeling tool without regard to their sequence or relationship to the point in time of the customer response. This relationship constitutes a time dependency between the response date and the sequence in the state of each variable prior to that date.

Lag variables permit expression of this time dependency between an event and the influences affecting its occurrence. The new variables derived from these temporal expressions provide a rich source of predictability for customer response. They provide insights into why the customer acted and represent the “other half” of the story of customer response that we can see in our databases. Lag variables (and trend variables calculated from them) permit the data mining tool to capture much of the signal of customer response resident in the time-series of the historical data.

Conclusions

There are four conclusions that can be deduced from the model results shown in [Tables 21.1 to 21.6](#):

- (1) Occam was right! The discussion of Occam’s razor in [Chapter 11](#) – Feature Selection suggested that there is a minimum set of variables that are required as predictors in building most analytical models. The best model using lag variables used only 21 predictors, whereas the model built with static variables used only 37 predictor variables, and the model built with all variables used 57 predictor variables. The beneficial effect of using static plus historical variables in the model ([Tables 21.3 and 21.4](#)) can be seen in comparison to the use of static variables alone ([Tables 21.1 and 21.2](#)). In this case study, however, the effect of feature selection (limiting the predictors to only lag variables) outweighed the beneficial effect of using both static and historical variables as predictors in the model.
- (2) Some variable types are better than other types for use as predictors. Notice that the model with the highest prediction accuracy metrics and with the best balance between Sensitivity and Specificity accuracy metrics is the MLP-neural network model using only lag variables as predictors ([Table 21.6](#)). In [Table 21.7](#), we can see that the top five predictor

variables are lag variables in the model that used all predictor variables in the model.

- (3) For this banking data set, the MLP-neural network algorithm produced the best performance both in prediction accuracy and in balance between the Sensitivity and Specificity accuracy metrics. In many business data domains, decision tree ensemble algorithms like Random Forest often win. But in this banking data set, the neural network is the best algorithm to use. We can never assume that one algorithm type is superior to another until we try multiple algorithm types and observe which one performs best with a given data set.
- (4) The most important predictor variable in the model using all predictors ([Table 21.7](#)) is Credit_Num-2, which is the number of credits (deposits) two weeks prior to the churn event. A list of customers scored with the model provides a bank relationship manager 2 weeks to contact the customer to offer a reason to stay with the bank.

Postscript

It may appear odd that the term “ecosystem” was applied to CRM in business, but there is a strong analogy between the fundamental structure and function of natural ecosystems and business processes; only the names and the items are different. The three main processes that characterize a natural ecosystem are: (1) energy flow, (2) nutrient cycling, and (3) food relationships expressed in terms of who eats who. In a natural ecosystem, energy enters in the form of sunlight that is transformed and flows through the ecosystem compartments in the form of chemical energy. The chemical energy makes everything else in the ecosystem happen. Necessary chemical nutrients (e.g., carbon and nitrogen) are used to synthesize numerous building blocks (e.g., sugars and proteins) for the growth of organism biomass, which are broken down by decomposition of dead organisms to produce the original components (carbon and nitrogen) and recycled to build biomass in new organisms. This recycling system is organized into “trophic” structures called food chains and food webs, which are composed of a series of organisms related by links in a chain or a web of who eats who ([Fath, 2008](#)).

In business, input energy is not sunlight; it is *money*. Money causes everything to happen. The food chains and food webs are represented by the employee-management human resources (HR) structures, which function a lot like the natural trophic structures of who eats who. Sometimes, lower-level employees feel that upper-level employees are “eating them up.” This feeling is particularly strong and sharp during employee union strikes. It should be no surprise, therefore, that the

same approaches and analysis methods used to model natural ecosystems also work in these business “ecosystems.” It is for this reason that the lag variables developed originally to model animal behavior also work to model human response behavior when it comes to buying goods and services.

Keywords and phrases

Response, CRM, Temporal, Patterns, Ecosystem, Time-series, Abstraction, Lag variables, Customer behavior, Static variables, Predictive accuracy, Business ecosystem, Human nature.

References

- W. Farrel, 1998. How hits happen. Harper-Business: New York.
- Fath, B. (2008). Ecosystem ecology In E. Jorgensen & B. Fath (Eds.), *Encyclopedia of ecology* (pp. 1125–1131). San Diego, CA: Elsevier Science imprint.
- Haimowitz, I. J., & Kohane, I. S. (1996). Managing temporal worlds for medical trend diagnosis. *Artificial Intelligence in Medicine*, 8(3), 299–321.
- Inmon, W. H., Imhoff, C., & Sousa, R. (1998). Corporate information factory. Hoboken, NJ: Wiley Computer Publishing.
- Jensen, C. S., Snodgrass, R. T., & Soo, M. D. (1996). Extending existing dependency theory to temporal databases. *IEEE Transactions on Knowledge and Data Engineering*, 8(4), 563–581.
- Kahn, G. M., Fagan, L. M., & Sheiner, L. B. (1991). Combining physiologic models with symbolic methods to interpret time-varying patient data. *Methods of Information in Medicine*, 30, 167–178.
- Kattan, M. W., Oshida, H., Scardino, P. T., & Beck, J. R. (1997). Applying a neural network to prostate cancer survival data In N. Lavrac, E. Keravnou, & B. Zupan (Eds.), *Intelligent data analysis in medicine and pharmacology* (pp. 295–306). Boston: Kluwer Academic Publishers.
- N. Lavrac, E. Keravnou, B. Zupan, 2000. Intelligent data analysis in medicine. Faculty of Computer and Information Sciences, University of Ljubljana, Slovenia: White Paper.
- Moore, J. F. (1996). *The death of competition: Leadership and strategy in the age of business ecosystems*. New York: Harper-Business.
- Petzinger, T., Jr (1999). *The new pioneers: the men and women who are transforming the workplace and marketplace*. New York: Simon & Schuster.
- Rothschild, M. (1990). *Bionomics—economy as ecosystem*. New York: Henry Holt.
- Rowe, C. (2012). *Plato: Republic*. London: Penguin.
- Wijsen, J. (2001). Trends in databases: Reasoning and mining. *IEEE Transactions on Knowledge and Data Engineering*, 13(3), 426–438.
- J. Wijsen, J. Vanderbulcke, H. Olivie, 1993. Functional dependencies generalized for temporal databases that include object-identity. Proc. Int'l Conf. Entity-Relationship Approach 100–114. Arlington, TX.

Academic Analytics: a case study in student retention

Preamble

Predictive analytics with Big Data in education will improve educational programs for students and fundraising campaigns for donors (Siegel, 2013). Research in both educational data mining (EDM) and data analytics (DA) continues to increase (Siemens, 2013; Baker and Siemens, 2014). The key elements of recruitment, learning, and retention can be tracked and increased over time in both cases for administrators and donor development. A few early examples of Big Data programs in these key areas of formal education will be reviewed. The paradigms of educational psychology can inform the construction of systems that maximize “engagement.” Contemporary educational technology enables the personalization of education, whether face to face or at a distance. More responsive evaluation systems will allow the school to demonstrate the effectiveness of its services. There are more opportunities for education as a sustainable enterprise with the implementation of Big Data, both locally and globally. Student retention is the “lifeblood” of the academic institution, without which the institution can die – quickly.

Introduction

A new and growing element of educational technology is the application of predictive analytics to model performance, retention, and the overall learning experience for students in schools and colleges (Parmar et al., 2014). Many articles in education journals use the key phrase “Big Data” in reference to these analytical operations, rather than the actual size of their data sets. Many school administrators depend on patterns in their data discovered by this technology to develop personalized plans for student service and remedial actions and to facilitate increased student success rates. Progress in instruction based on these patterns can be monitored and amplified with the right interventions and at the

right time. Big Data analyses have great implications for education at all levels, ages, and venues, but it is important to understand the potential opportunities and risks in these operations (Nisbet et al. 2017).

Academic Analytics

The discipline of Academic Analytics includes: (1) student retention studies, (2) student recruitment, and (3) donor development. This chapter will focus on student retention studies only. This focus was selected because student retention analysis offers a relatively quick harvesting of “low-hanging fruit” in the form of retaining students who have a high probability of leaving before graduation (“churning”). Students are the “lifeblood” of a school. The primary goal of any school is to “lose” students; that is, to see them graduate. One of the primary strategies to accomplish this goal is to retain the students they have until graduation. The next two most important strategies in administration are to gain new students (student recruitment) and enhance donor development. Apart from the order in which these activities occur in an institution, we will discuss them in the logical order of occurrence, beginning with student recruitment, followed by student retention and donor development (beginning with alumni).

The basic process followed in all of these studies is shown in Fig. 22.1.

The general goals of Academic Analytics are: (1) adaptation of old processes to new conditions, (2) personalization of processes to fit each student, and (3) intervention to improve the structure of the process or prevent an undesirable outcome (e.g., student attrition).

Student retention

After students begin an educational program, a primary goal for both business and educational purposes is to retain the student in the program until

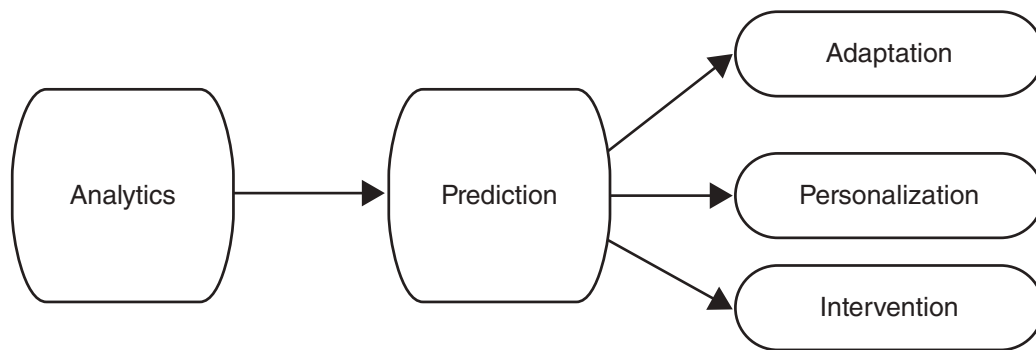


FIG. 22.1 The general process followed by the deployment of student recruitment and retention analyses. <https://www.google.com/search?q=Educational+analytics+images&biw=1536&bih=735&tbm=isch&tbo=u&source=univ&sa=X&ved=0ahUKewjdwoex0ZXSAhWE5CYKHdfuA54QsAQIGQ&dpr=1.25#imgsrc=ofMEsBNy4d5W7M>

graduation. Graduation is certainly the ultimate goal, but a secondary goal is persistence in the program. Some students leave for a term or two (stop-outs), while others *persist* in their enrollment from term to term. A better learning experience is fostered by greater persistence to graduation and it provides a more consistent financial environment for the institution. If a well-designed Big Data program is in place, then the patterns of successful students can be identified and “shaped” by future students. Beginning with hunches by the educators, data can be acquired and arranged for machine learning algorithms to find significant combinations of variables for testing and review. It is important to note that these steps constitute much more than just a record of student withdrawal from the program. There is emotional baggage with “churn” (attrition) in education and business. The primary focus of the administration on the data from successful students does not include these factors, but Big Data analyses can help to analyze them in a much more rigorous way than previously. A case study of student retention will be presented below.

This student retention study was supported by a grant from In Trust Foundation. The study represents both an update of a previous study done in 2015 and also an opportunity to extend the scope of text mining operations begun in 2015. The purpose of studying student retention is to guide the design of appropriate student interdiction programs to increase the probability that students will be retained in the seminary until graduation. The 2015 study focused on the complement of student retention, or student “churn.” The term “churn” was used first in modeling customer attrition in telecommunications in the mid-1990s, when phone service customers moved around among several of the “Baby Bell” phone companies (following the breakup of AT&T in 1986) and other service providers of the time, following the current best “bang for the buck.” These customers

were viewed as if they were flakes of butter moving around in a butter churn; hence the term “churn” was associated with customers who stopped buying goods and services from a company or organization. We chose to focus on the student retention side of the binary classification (instead of the churn side) in this study because some machine learning algorithms select important predictors (sometimes from among several candidates), which more directly explains the data pattern in the positive class, rather than the negative class. Hence we defined the positive class to be predicted in this study as students who are retained to the grant of a degree, rather than direct the machine to focus on the churn students. This model design produced results that could guide the Western Seminary administrators to design policies and programs focused on increasing student retention.

Retention history of Western Seminary

The previous churn analysis at the seminary in 2015 found that the churn rate was very high, at about 80%. The administration instituted several policies and programs aimed to increase student retention for many reasons, including:

1. The financial health of the seminary will be enhanced by stabilizing retention, which in turn will stabilize revenue.
2. The seminary will be able to serve its students better and generate further benefits for society.
3. The seminary must keep up with other seminaries, which will be doing analytical retention analyses soon.
4. The seminary will support many of its operations with advanced predictive analytics in the future; therefore it is prudent to build an analytical platform that will serve its future needs.

Student retention: a case study

Summary

This case study was performed for the analysis of student retention at a theological seminary in the United States in 2019, funded by the In Trust Foundation. It was the second analysis of student retention at the seminary. The first analysis of student retention at the seminary was conducted in 2015 and it focused on predicting which students did not continue in the seminary until receiving a degree (“churn” students). However, the 2019 study focused on the complement of student churn, student retention. This change appears to be rather subtle, but it shifted the focus from students who “cost” the seminary money to those who benefited the seminary by graduating with a degree. Completing a seminary degree also brings great personal and professional benefit to the student and the seminary. Gaining a new student for the school is 10 times the cost of retaining a present student. The total financial value of a student is between \$33,000 (MA) and \$50,000 (MDIV). This value is fully realized only by retention of students through the 60 credits (MA) or 90 credits (MDIV).

Data were extracted from the student information system and prepared in the open-source KNIME Analytics Platform and integrated into a single student-centric data set covering students who were enrolled at the seminary during the past 5 years. Unlike the 2015 study, GPA data elements were not used as candidate predictor variables because the seminary wanted to focus on the action of other variables affecting student retention. Also, the present study derived many new variables from combinations of existing variables. Separate models were built in KNIME and in the IBM SPSS Modeler (hereafter Modeler) analytics tool bench. Separate models were built for the online students and for students attending each on-campus location. Among these models trained, different subsets of predictor variables had the highest importance value in the models.

Methodology

The overall modeling process is shown in Fig. 22.2 in the form of a KNIME modeling workflow.

Data preparation operations

1. IBM SPSS Modeler was used to derive some time-related variables because it has a richer body of temporal functions available for data preparation than those available in KNIME. Also, a number of variables were derived from free-form text fields in student records. These operations are not shown in this case study.
2. KNIME was used to integrate data elements from a number of source data tables provided by the seminary.
3. KNIME was used to derive numerous variables including:
 - a. Mean(LikertScore_Online
 - b. Mean(LikertScore_Oncampus
 - c. Unique_Count(Comment)
 - d. StopOuts number
 - e. Total_Grade_Points
 - f. Sum(Hours)
 - g. Sum(Dropped)
 - h. Sum(Audit)
 - i. Drop_Rate_Yr
 - j. Grants_number
 - k. Loans_number
 - l. Aid_Sum
 - m. StopOut_Rate_Yr
 - n. Mean_Term_GPA
 - o. %GPA_improvement
 - p. Count(ActivityType.forum
 - q. Count(Description)
 - r. Sum(Service_Response_Months);
 - s. Sum(Hold_Yes);
 - t. UniqueCount(Actions)
 - u. Highest_Ed
 - v. Audit_Only
 - w. Total_Debt
 - x. Target variable

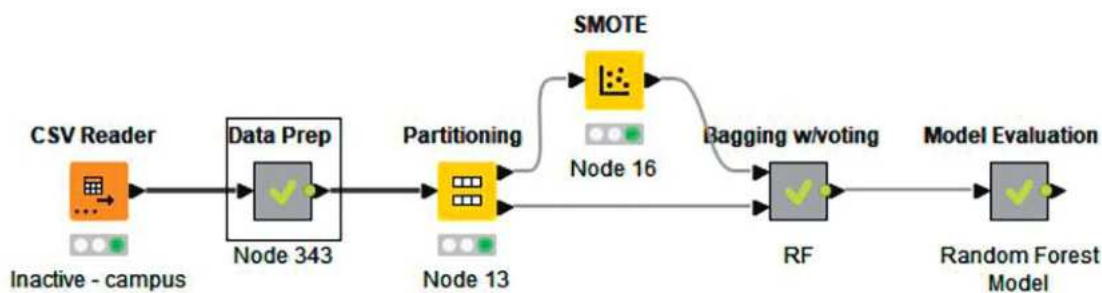


FIG. 22.2 High-level view of the KNIME modeling workflow to model student retention.

(Note: This part of the workflow was very complicated and therefore is not shown in this case study).

Data conditioning operations

The Data Prep metanode was used to perform several necessary data conditioning operations on the data set before it could be submitted to the modeling algorithm. Before the modeling data set can be input to the modeling algorithm, it must be conditioned properly by performing several operations in the Data Prep metanode shown in Fig. 22.3.

Filtering operations

Variables that contain only one value are useless for modeling and are removed by the Constant Value Column Filter node in Fig. 22.3. Variables with a lower than threshold variance value would be of little use in the classification of the target class; therefore they are removed by the Low Variance Filter node. Miscellaneous variables are removed by the Column Filter node (e.g., student name).

Missing value handling

Machine learning algorithms will delete (by default) an entire row if there is even one missing value in one of the variables in the row. Therefore appropriate values were imputed for missing values with the Missing Value node shown in Fig. 22.3.

One-hot variable processing

While it is not strictly necessary to convert categories of string variables into numbers for modeling with machine learning algorithms, the algorithms often work

better with numbers. Also, conversion of string variable classes to numbers permits the use of parametric statistical algorithms, although none were used in the final modeling operations.

Partitioning operation

The partitioning operation is performed by the Partitioning node in Fig. 22.2. The input data set was partitioned by the Partitioning node into a modeling data set containing 70% of the records flowing from the top output of the node and a hold-out validation data set with 30% of the records flowing from the bottom output of the node. A third data set (the Testing data set) will be discussed below.

The balancing operation

The SMOTE node in Fig. 22.2 performs the Synthetic Minority Oversampling Technique (SMOTE) using a nearest-neighbor analysis to synthesize enough new records of the target minority class to equal the target majority class. This balancing operation is necessary prior to modeling with machine learning algorithms to provide an equal number of records in the two target classes for the algorithm to classify the modeling records into one of the two target classes without any bias.

Bagged modeling with a Random Forest algorithm

The modeling operation is performed by a series of KNIME nodes collected into the Bagging w/voting metanode of Fig. 22.2. Fig. 22.4 shows the use of a Random Forest algorithm within a bagging loop.

The Shuffle node shown in Fig. 22.4 performs a randomization of the input rows before they are submitted

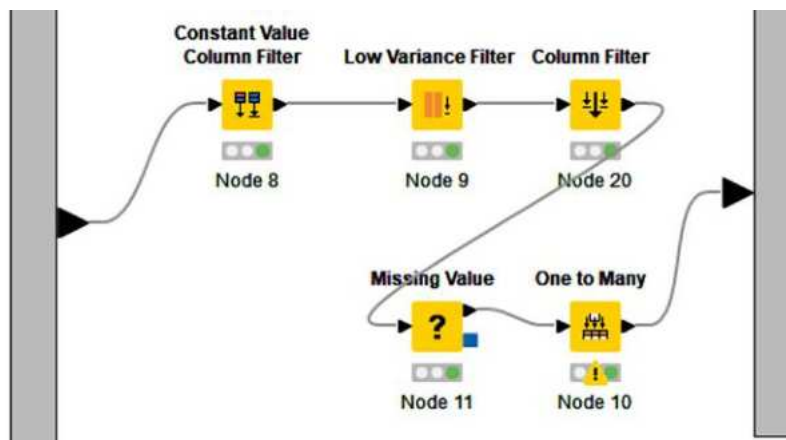


FIG. 22.3 Detailed view of the Data Prep metanode in the KNIME student retention modeling workflow.

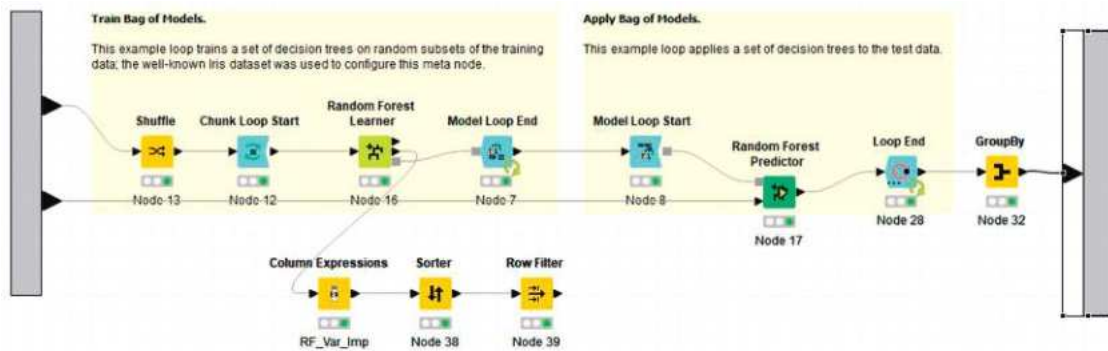


FIG. 22.4 The bagging loop used to train a Random Forest model of student retention.

TABLE 22.1 Top 10 predictor variables in importance in the model.

S Variable	S Variable Importance Value
Count(Stop Out)	10
Yrs_Since_Entered	8
Sum(Late_Pmt_Days)	5
Mean(LikertScore_Campus)	4
Count(ActivityType.forum)	4
Sum(Service_Response_Months)	4
EFC	4
DegreeLevel	3
Aid_Sum	3
Moodle_Signons_Per_Month	3

to the Random Forest Learner node. This randomization is necessary because the input data set is partitioned internally to produce a training set and a testing set in a 50:50 ratio. Randomization is necessary to make sure that the balance between the target classes is maintained for modeling. The initial partitioning operation shown in Fig. 22.2 and the internal partitioning operation performed in the Random Forest Learner node combine to produce three data partitions for modeling:

- Training set used to build the decision tree in each iteration of the algorithm
- Testing set used to calculate prediction error for a given iteration, which is used to adjust model hyperparameters for the next iteration of the algorithm
- Validation set used to evaluate prediction accuracy metrics after the last iteration of the algorithm (set in the configuration to 100 iterations)

The bagging loop formed by the Chunk Loop Start and Loop End nodes shown in Fig. 22.4 builds five models with five different random samples of the shuffled input data. The bagging operation removes much of the effect of sampling error incurred in formation of the training data set.

The GroupBy node shown in Fig. 22.2 aggregates results of the five bagging models to produce a final prediction of the target class for each student.

Results

Predictor variable importance

The importance value of each variable included in the model is calculated by the Column Expressions node labeled as “RF_Var_Imp” in the data stream shown in Fig. 22.4. The top 10 predictor variable importance values are listed in Table 22.1.

Key word list

students
 seminary
 learning analytics
 retention
 donors
 academic analytics
 education
 donor development
 big data
 graduation
 patterns
 predictive analytics
 churn
 educational data mining
 recruitment
 machine learning algorithms
 educational analytics
 big data analyses

Prediction accuracy assessment

The prediction accuracy values for the bagged Random Forest model are shown in Table 22.2.

Table 22.2 shows that the overall accuracy of the bagged Random Forest model was 0.896, or almost 90%. The model was almost equally powerful in predicting Target values = 1 (Sensitivity) and Target values = 0 (Specificity). The area under the ROC curve (AUC) value in Table 22.2 was 0.974, where 1.0 is a perfect mode. The ROC curve is shown in Fig. 22.5.

TABLE 22.2 Student retention model prediction accuracy metrics.

S Metric	S Value
Sensitivity	0.916
Specificity	0.864
Accuracy	0.896
AUC	0.974

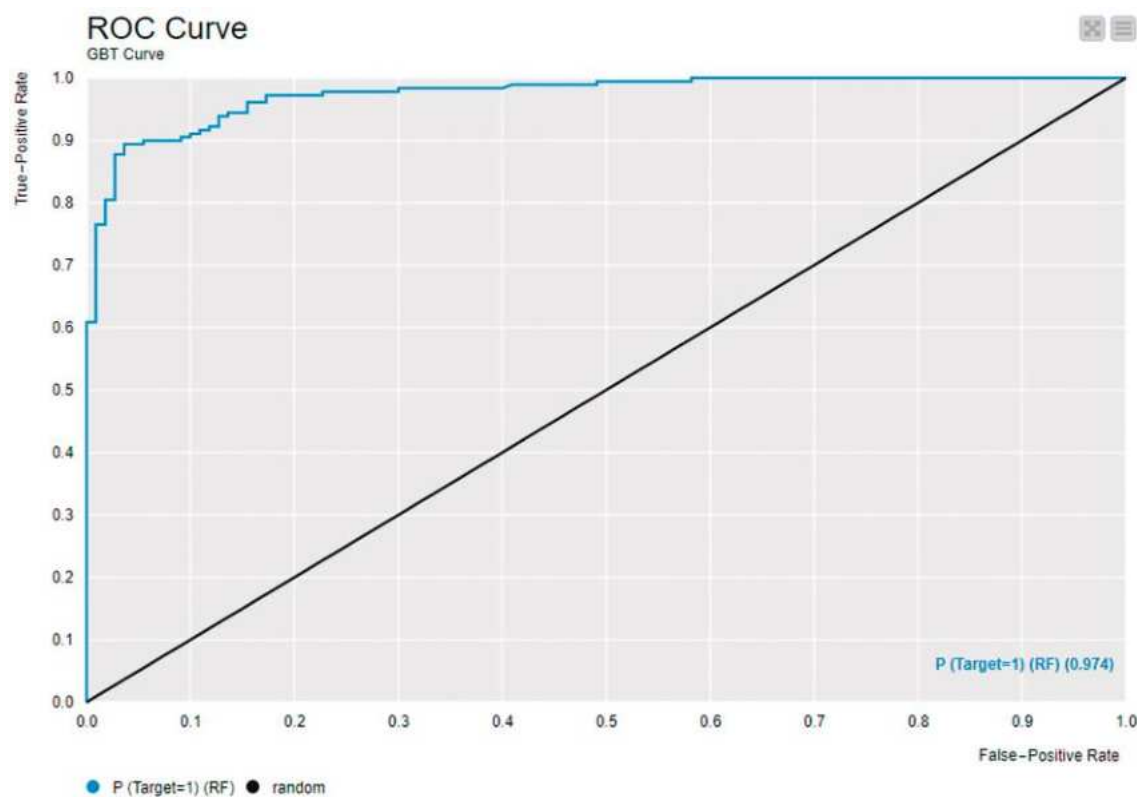


FIG. 22.5 The receiver operating characteristic (ROC) curve for the student churn analysis. Note the area under the curve (AUC) value of 0.974 shown in the lower right-hand corner of the plot.

Discussion

Results show that models for all seminary campuses (including the online campus) predict past student retention rates at above the 90% rate. Based on these results, the overall student retention rate for all campuses was calculated as 65% in the future, compared with the 50% rate in 2019 and only 20% in 2015. This increase reflects the effects of policies and programs put into place subsequent to the 2015 study of seminary student retention and illustrates the power of analytical modeling of student retention data.

Postscript

The case study in this chapter may beg the question, “Where do we begin?” One approach to answer that question is expressed in terms of the present 12 steps for implementing projects in Academic Analytics.

Step 1. Review strategy with your interdisciplinary team:

Evaluate your legacy business questions (sustaining operations), new questions for the current business (sustaining innovation), and new business opportunities (disruptive innovation)

Step 2. Audit your data:

List the database silos with customer data, product data, and interaction in the ecosystem

Step 3. Survey your customers and graduates.

Step 4. Analyze your *predicted variables*:

Use an accepted method to rank predicted variables to show what is important thus far (e.g., GPA, job placement?). Add new predicted variables, including aggregations (e.g., 5-year evaluations) and various abstractions (e.g., time-since some action), and combinations of individual variables to derive new variables such as `Learner_engagement`.

Step 5. Use some Feature Selection tool to select those variables with the most potential to be powerful predictors and generate a “short list” for submission to modeling algorithms.

Step 6. Plan dashboard metrics for stakeholders and decision makers.

Step 7. Select software application:

Select analytic methods, algorithms, and machine learning.

Step 8. Construct hardware platform to hold the data mart to support Big Data analytics.

Step 9. Design the logical and physical structure of the data mart.

Step 10. Capture student responses and analyze input data.

Step 11. Do staff training for delegation of maintenance of the system.

Step 12. Build Adaptive Assessment by using inter-related content, self-pacing of questions, and immediate feedback to student and teacher.

Keywords and phrases

Students, Retention, attrition, interdiction strategies, Academic Analytics, Big data.

References

- Baker, R., Siemens, G. (2014) Educational data mining and learning analytics, in: Sawyer, K. (Ed.) *Cambridge handbook of the learning sciences*. 2nd Edition. Cambridge, UK: Cambridge U. Press.
- Nisbet, R., Miner, G., & Yule, K. (2017). *Big Data in Education: Handbook of statistical analysis and data mining applications* (2nd Edition). New York: Academic Press.
- Parmar, Rashnik; Mackenzie, Ian; Cohn, David, and Gann, David. (2014). The new patterns of innovation: How to use data to drive growth. *Harvard Business Review*, January–February.
- Siegel, Eric (2013). *Predictive analytics: The power to predict who will click, buy, lie, or buy*. Hoboken, NJ: John Wiley & Sons.
- Siemens, G. (2013). Learning analytics: The emergence of a discipline. *American Behavioral Scientist*, 57(10), 1380–1400.

Medical informatics case study: predicting type 1 precancerous colon polyps

Prelude

Chapter 10 presented basic feature engineering operations to transform raw input data into useful modeling features. Chapter 14 described how to build a machine learning model to predict a target variable with predictor variables prepared with techniques described in Chapter 10. Chapters 15 and 16 described various model evaluation and enhancement techniques, including ensemble models. This chapter will dig deeper and focus only on those processing steps required to use EMR data for predictive modeling. This complex group of processing steps constitutes a form of engineering analogous to transforming a pile of raw materials and manufactured components into a functioning automobile.

Introduction

Colon cancer incidence in the United States

The colonoscopy procedure became popular in the United States after then-President Ronald Reagan underwent a life-saving colonoscopy in 1985. In the mid-1990s, screening guidelines were proposed, recommending regular colonoscopy procedures be performed after the age of 50. Between 2000 and 2016, there was a 3.3% decline annually in the incidence of colon cancer among individuals aged 65 years and older (Siegel et al., 2020). This trend, however, was reversed by showing an increase in incidence of 1% annually in those people aged 50 to 64 years, and an increase of 2% among individuals younger than 50 years. The sharp decline in colon cancer rate in individuals 65 years and older may be caused by the ameliorative effects of colon cancer screening by colonoscopy, but the

increase in incidence for younger people is of concern to physicians. Based on these trends, the 2018 American Cancer Society guidelines for colon cancer treatment recommended that the age to begin regular colonoscopy exams be reduced from age 50 to 45 (Wolf et al., 2018). Despite the reduction in the rate of colon cancer incidence above age 65, colon cancer remains the fourth leading cause of death by cancer in the United States (CDC, 2022). Therefore more aggressive measures of colon cancer detection are needed.

Colon cancer screening tests

The advent of the use of colon cancer test kits like Fecal Immunochemical Test (FIT), the Fecal Occult Blood Test (FOBT), and the FIT-DNA test (combining a DNA test with the FIT) provides alternative screening tools for colon cancer detection, in addition to colonoscopies. Other, more basic tests are needed, such as a screening model that analyzes common electronic medical record (EMR) data, such as BMI and blood chemistry data. Such a screening tool could be used for patients of all ages, would be noninvasive rather than require a colonoscopy as part of the test, and would be much more convenient than any of the screening tools currently in use. This chapter will describe the complex data engineering methodology necessary to build such a screening model based on EMR data.

The University of California, Irvine Colon Polyp Risk Modeling Project

The current 2% per year rise in the rate of colon cancer in people younger than 50 years and the inadequacy of present screening tests prompted a team of professionals

at the University of California, Irvine Medical Center to form a modeling project to address these issues. Staff for this project included:

- William Karnes, MD, Director of High-Risk Program and Colonoscopy Quality, H.H. Chao Comprehensive Digestive Disease Center
- Erica Duh, MD, Chief Resident, H.H. Chao Comprehensive Digestive Disease Center
- Robert Nisbet, PhD., Data Scientist, UC-Irvine
- Kai Zheng, PhD., UCI Institute for Clinical and Translational Science Center for Biomedical Informatics

This team developed a novel risk model predictive of one or more type 1 colonic neoplastic lesions at the time of colonoscopy. Modern machine learning techniques were used to combine tabular data from our UCI Colonoscopy database (presence or absence of colonic neoplasia) and the Epic Data Warehouse containing thousands of data points related to patient demographics, diagnoses, medications stored in five Operational Medical Outcomes Partnership (OMOP) data dimensions in the UCI Health Services database. When this model is deployed, it will help guide which patients should receive earlier (or later) colonoscopy screening.

The list of steps shown in Table 23.1 constitutes an overall process flow of operations, but the steps are not strictly sequential. For example, some new variables must be derived (Step 7) to calculate expressions like the mean systolic blood pressure values over all records for a given patient before grouping of records to the patient level (Step 5) can be performed. Other new variable derivations at the patient level are performed in Step 7. This flow of operations is shown in Fig. 23.1.

This medical application example can serve as a primer for data preparation engineering that can be applied to many modeling problems in many different data domains.

TABLE 23.1 Steps in the data preparation operation for the analysis of EMR data.

Data engineering steps for data preparation for modeling

The set of data engineering processes required to build the modeling data sets consisted of 10 steps:

1. Data source analysis
2. Addition of demographic variables
3. Data integration
4. Data description and quality analysis
5. Data reformatting
6. Grouping to the patient level
7. Data Exclusions or specific patients (rows) or variables (columns) for medical reasons
8. New variable derivation
9. Feature selection
10. Data conditioning

Step 1: Data source analysis

Many medical organizations store data in several different locations in several different formats (e.g., spreadsheets, computer databases, and billing systems). Seldom are all of the data elements available in a single data warehouse. The analyst must identify where necessary data are stored, determine how to access them, and gain permission to do so. This may be a lengthy process that must be accounted for in the analytical project plan.

The UCI Health Services medical database follows the OMOP data structure, with five concept categories:

- Conditions: 7,460,000 records
- Procedures: 9,500,000 records
- Drugs: 4,390,000 records
- Measurements (e.g., blood chemistry: 17,412,000 records
- Observations (e.g., smoking history): 4,390,000 records

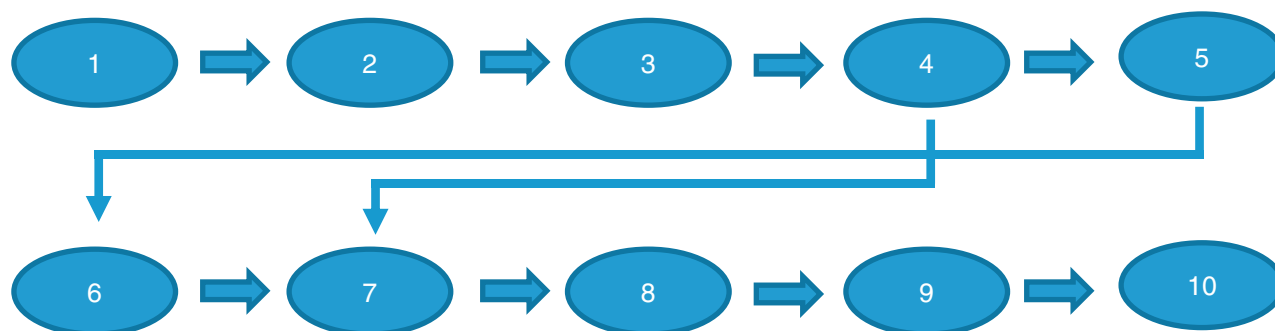


FIG. 23.1 Order of processing of the 10 steps of Data Engineering listed in Table 21.1.

The UCI Colonoscopy Quality Database (UCICQD) contained 22,041 records for 11,992 patients with data elements including:

- Age
- Gender
- Basal Metabolic Index (BMI)
- Colonoscopy results

Information from both of these UCI data sources was examined, and extracted into separate comma-separated (CSV) files.

Analytical project planning

A project plan should be composed in a project planning tool like Microsoft Project to keep track of finish-to-start relationships among tasks, task durations, and staff resources required to complete them. This plan can also be used to manage task slippages and provide progress reports to management. Failure to keep all of the many tasks properly coordinated may lead to inefficiencies and even to failure of the project.

Step 2: Inclusion of demographic and lifestyle variables

In many analytical models, one or more of the most important predictors are demographic, not related to the organization (the clinic in our case). For our data set, some demographic data were available in the UCICQD database, which were provided by patients. These variables may not have been consistently filled in by patients, but it was decided to use them anyway.

- Gender
- Ethnicity
- Smoking behavior
- Alcohol consumption behavior
- Marital status
- Sexual orientation
- Homelessness

In other organizations, demographic data can be provided by various data vendors, such as Acxiom, Equifax, and Experian.

Step 3: Data integration

The biggest problem with EMR patient data is that many records may be available for the same patient. Some potential predictor variables must be derived before these records are collapsed into a single record per patient, as per Fig. 23.1. Such variables include totals and average (mean) values for data elements such as blood measurements judged to be “High,”

“Low,” or “Normal” across all records for a given patient.

Step 4: Data description and quality analysis

Each data element must be examined to determine if they are of the proper data type (e.g., integer or real) required for its intended use in the analysis. For example, classification algorithms require a string data type input (a categorical variable) for use as a target variable. Data extraction operations and transmission of data from its source may cause the intended target variable to be cast as an integer, which must be changed to a string data type for use in the classification algorithm. Other analyses that should be performed include distribution plots for each data source to identify possible outlier values. This data type change was performed on the colon cancer input data source because the target value was available in the form of a CSV file which, when loaded into the modeling tool, was autotyped (cast) as an integer.

Data quality analysis should include research on the reliability of all data elements, including the number of missing values in each element. This information will guide later operations such as missing value imputation. Often, invalid dates are stored and must be corrected (e.g., 12-32-2021 changed to 12-31-2021) or removed.

Step 5: Data reformatting

Many medical operations store data in several different formats in several different places; few have data warehouses. This situation requires identification of the storage format and location of every data source required for analytics operations (Step 1). Subsequently, information stored in different formats must be reformatted and/or transformed into a common format to serve analytics.

These operations were not necessary for the colon cancer data set because all medical data in patient records were expressed according to the OMOP data dictionary.

Step 6: Grouping to the patient level

All data sources provided many rows of data for each patient. For example, the UCICQD provided over 22,000 records (rows) of data for 11,217 colonoscopy patients. These patients were matched with the millions of records in the UCI Health Services OMOP database in each of the concept categories for Conditions, Procedures, Drugs, Measurements, and Observations.

Step 7: Data exclusions

Row deletions. After grouping to the patients level (one row per patient), some rows must be deleted from the data set for various reasons, including:

- Patients with previous colonoscopies (this study focuses on first-time colonoscopy patients)
- Patients with invalid procedure dates

Column deletions. Some of the OMOP codes are not suitable for modeling and must be removed from the variable list.

- If the colon benign neoplasm (or colon malignant neoplasm) codes reflect the findings on the current colonoscopy, they should not be included (e.g., if entered on the day of or after the colonoscopy). If they predate the colonoscopy, they are clear risk factors for the presence of polyps in the current colonoscopy and cannot be ignored. Basically, any data entered on the procedure day or after should be ignored.
- Colonoscopy-related variables excluded:
 - Procedures related to the colonoscopy procedure
 - Drugs used during and prescribed after a colonoscopy
- Age-related variables were excluded, including:
 - Age
 - Age is a known factor related to the instance of colon cancer.
 - This screening tool is to be built on factors not related to age and used for recommendation of a colonoscopy, regardless of age.
 - Pneumonia and Zoster (Shingles) vaccination codes for Procedures and Drugs.
 - The instances of Pneumonia and Shingles are strongly related to age and must also be removed from the analysis.
 - This exclusion is a good example of the importance of domain knowledge in the data engineering process.
- Validation of each code to make sure that it does not inject any information inappropriate for this analysis. For example, several procedure codes and condition codes were found that could only be learned from the results of a colonoscopy (e.g., presence of malignant neoplasm in the colon). Therefore a screening model for patients with precancerous polyps before a colonoscopy is performed cannot use colonoscopy results data as predictor variables.

Step 8: New variable derivation

Based on the frequency analysis of OMOP codes, 2025 candidate binary predictor variables were derived for the Procedure, Condition, and Drugs concept categories. A value of 1 was assigned for presence of a given code

and 0 for its absence in each patient record. Additional variables were derived for:

- Other specific binary variables based on presence/absence of known conditions related to colon cancer, such as kidney disease.
- A binary target variable was defined as a string according to a series of medical rules applied to patients with first-time colonoscopies. This heuristic was expressed as an embedded-if-then-else logical statement composed in a Column Expressions node in KNIME.
 - If polyps are found that are described as “Hyperplastic” and are greater than or equal to 10cm in size, then set the Target variable to 1;
 - else if a polyp is described as “Serrated,” then set the Target variable to 1;
 - else if a polyp is described as a “Tubular adenoma,” then set the Target variable to 1;
 - else if a polyp is described as “Tubulovillous,” then set the Target variable to 1;
 - else if a polyp is described as “CA” – cancerous, then set the Target variable to 1;
 - else if a polyp is described as “Villous,” then set the Target variable to 1;
 - else if a polyp is described as a “Flat adenoma,” then set the Target variable to 1;
 - else set the Target variable to 0.
- Other variables were derived from the concept codes for Observations and Measurements:
 - Calculation of blood pressure mean values and ranges, heart rate mean value
 - Conversion of all “High” and “Low” specifications in Measurements concept codes to derive high and low outcome variables for each code
 - Counting the number of high and low measurement codes for each patient
 - Text mining techniques were employed to derive variables from the text information in the Observations concept category (e.g., variables related to smoking and alcohol use).

Discretization of textual variables

All nonnumerical variables contain textual information representing classes of the variable, like Ethnicity, which contains a list of codes for different ethnic groups. Machine learning modeling algorithms can work directly with these categorical variables, but parametric statistical modeling algorithms cannot (e.g., Logistic Regression). For general use, categorical values must be transformed into a group of discrete binary variables, one for each class in the categorical variable. For example, if the Ethnicity classes included codes 1 for White, 2 for Hispanic, and 3 for African-American, three new variables would be derived. For example, if a record contained an ethnic

code = 1, the binary variable for White would be assigned a value of 1, otherwise 0 would be assigned. Separate binary variables would be defined for Hispanic and African-American codes. Such a binary variable is referred to by some practitioners as a “dummy variable.”

Step 9: Feature selection

OMOP code frequency analysis

Codes in the Observations concept categories are handled separately in Step 7, but the problem remains for identifying which of the 50,000 different OMOP codes are in the Conditions, Procedures, Drugs, and Measurements concept categories. It was decided that only the most frequent among these codes in the UCI Health Services database would be derived as candidate predictor variables. Therefore a frequency analysis, consisting of the following text mining operations, was performed on all patient records.

1. Create a bag of words from the code strings to produce a list of common terms.
2. Perform a frequency analysis on the terms.
3. Sort the term frequencies from high to low for each of the four concept category codes in descending order to express the frequency of a given code.
4. Select the 675 codes for each of the Condition, Procedure, and Drug concept domains with the highest frequencies.
5. Select 200 codes for each of the Normal, High, and Low measurement results classes.
6. Eliminate all codes with dates on or after the colonoscopy date to prevent those data from being triggered by the colonoscopy procedure or following it.

A total of 2421 variables were derived for modeling, after which the list of variables was reduced to eliminate those variables judged by a set of heuristics to be of little or no use to the modeling algorithms. This operation is referred to as *feature selection*. Feature selection is a very important aspect of predictive modeling that is often overlooked or minimized. [Chapter 11](#) of this book is dedicated to feature selection; therefore only the feature selection steps used in the analysis of colon polyp data will be discussed here. The set of heuristics chosen for feature selection included:

- Constant value filtering
- Low variance filtering
- Correlation filtering
- Embedded modeling feature elimination

Constant value filtering

The initial feature selection filter removes those variables with a single value which have no value as potential predictor variables in any modeling algorithm.

Low variance filtering

The second filtering operation removes those variables that have a less than threshold variance because they have little potential to contribute to distinguishing between the data patterns of the two target classes.

Correlation filtering

The last initial filtering operation calculates the Pearson product-moment correlation coefficients and deletes those variables with a coefficient of 0.8 and above. The reason for this operation is that for statistical analysis modeling algorithms (e.g., Logistic Regression), highly correlated variables may generate a covariance matrix that cannot be inverted and solved. Machine learning algorithms are less sensitive to effects of highly correlated variable pairs, but the combined effect of many of them (termed *multicollinearity*) can increase the likelihood that the solution selected will not have the lowest minimum error (the solution search becomes “trapped” in a local minimum error location on the decision surface).

Embedded modeling elimination

[Chapter 11](#) describes a number of multivariable modeling methods for either eliminating unimportant modeling variables or calculating feature importance values, and then eliminating variables below a specified threshold importance value.

A multivariate method of feature selection using the XGBoosted Trees algorithm generated a list of important variables based on the information gain associated with each variable. The top 276 variables were selected (those with an information gain value greater than 1.0). Models with different information gain thresholds were tested, and the threshold = 1.0 was judged to produce the best model. The rule followed was to select the smallest number of predictor variables that produces nearly the most accurate model.

This rule flows out of the concept of Occam’s razor, which prescribes that the simplest explanation of a phenomenon is the best explanation (see [Chapter 11](#) for a more comprehensive discussion of Occam’s razor). Following this rule minimizes the number of variables in a model that is very near the most accurate model. The best combination of small number of variables and highest accuracy is selected. This list of variables is called the *short list*.

The short list is a valuable intermediate product of a predictive analytical modeling operation. [Chapter 11](#) describes the outcome of the *US District Court of Appeals case of AT&T vs. DTM Research*, in which the short list generated by DTM Research was deemed to be intellectual property acquired illegally by AT&T, leading to a judgment of \$17 million against AT&T.

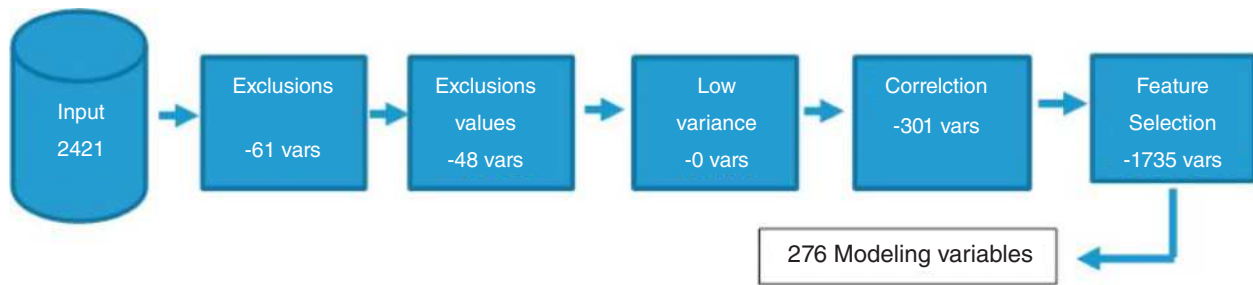


FIG. 23.2 The variable exclusion processing to generate a modeling data set of 276 variables.

Fig. 23.2 shows the development of the final short list of 276 modeling variables with the process through 5 variable exclusion processes.

The final modeling data set consisted of 276 candidate predictor variables, plus three descriptive variables and the target variable:

- Person_id – an anonymous sequential identifier for each patient;
- #Type1 polyps – number of type 1 polyps removed during the colonoscopy procedure;
- AgeProc – patient age at the time of the colonoscopy procedure;
- Target_binary – the modeling target variable (defined above).

Step 10: Data conditioning

The final set of operations in the data preparation process consists of a final modification of variable values for each patient (standardization) and balancing of rare target categories to fit the requirements of various modeling algorithms to be used in the modeling process.

1. *Standardization.* Candidate predictor variables may have very different ranges of values. The estimation of regression coefficients is affected by the relative importance of the variable to the solution and the effect of range. The resulting coefficient expresses the importance of the variable and a factor to correct for the effect of the variable's range. Standardizing input data to a regression operation removes the effect of differing ranges among the predictor variables. Standardizing data inputs is important for some machine learning algorithms that predict numbers (not classes) because a regression process is a component of the processing of the algorithm. Input colon polyp data was standardized to test the efficacy of the Logistic Regression algorithm in the modeling process.
2. *Balancing data sets with rare target classes.* If there are significant differences among the numbers in the Target classes, the assignment of predictions by the modeling algorithm will be biased toward the most

frequent class. The bias may reduce the overall accuracy of the models built on the data set. Balancing of any and all relatively rare target classes can be accomplished by one of two general operations:

- a. Undersampling the relatively common classes until the number of rows of each common target class equals the number of rows of the rare class. This operation can be performed in KNIME with the Equal Size Sampling node.
- b. Oversampling the rows in the relatively rare target classes until the number of rows equals the number of rows in the most common class. This operation can be performed manually in KNIME with the Sampling node or by using the Synthetic Minority Oversampling Technique (SMOTE) node.

The KNIME Statistics node was used to determine that Target classes of the final data set are almost perfectly balanced (1 = 49.7%; 0 = 50.3%), therefore balancing was unnecessary.

Modeling

The modeling phase of the project consisted of:

- Algorithm selection
- Architecture of the modeling loop design
- Model building
- Composition of results from three selected models into an ensemble model
- Model evaluation

Algorithm selection

The following algorithms were tested for prediction accuracy on the final data set:

- Multilayer perceptron (three-layer neural network)
- Deep learning neural network with two middle layers
- Logistic Regression
- Support vector machine

- Random Forest
- Gradient Boosted Trees
- XGBoosted Trees

Three of the modeling algorithms performed best:

1. Gradient Boosted Trees
 - a. The base learner is a classification tree that serves as a weak learner.
 - b. Weights of the initial tree are adjusted up for important predictor variables and down for unimportant predictor variables to provide input for the next tree that is built.
 - c. 100 boosted trees are trained in a champion-challenger process and the best predictor tree is saved.
2. Random Forest
 - a. The base learner is a decision tree that serves as a weak learner.
 - b. The optimum cut-point along the scale of each predictor variable is selected for the first iteration.
 - c. Subsequent iterations optimize the position of the cut-point using a random selection process along the range of the variable.
 - d. 100 trees are built following a champion-challenger process, with the best (most accurate overall) model retained after each iteration.

3. XGBoosted Trees

- a. Stands for Extreme Gradient Boosting algorithm.
- b. Much more scalable than the ordinary Gradient Boosting algorithm because of its use of named threads to facilitate parallel operation even with a sequential CPU. This architecture can allocate internal buffers for each thread to store gradient statistics.
- c. This algorithm iterates over all the possible splits for all the features. The process works best for symbolic variables rather than continuous variables.

A fused ensemble prediction was calculated as the most common prediction among the three models for each row in the validation data set.

Model architecture

Fig. 23.3 shows the complete KNIME modeling workflow after new variable derivation is completed.

The XGB Short-list Builder node is a *metanode* containing a number of processing nodes that perform the embedded feature selection operations described above. The Prep and Filtering metanode contains a series of nodes that perform the filtering and exclusion operations as described above. The Fused Ensemble Model

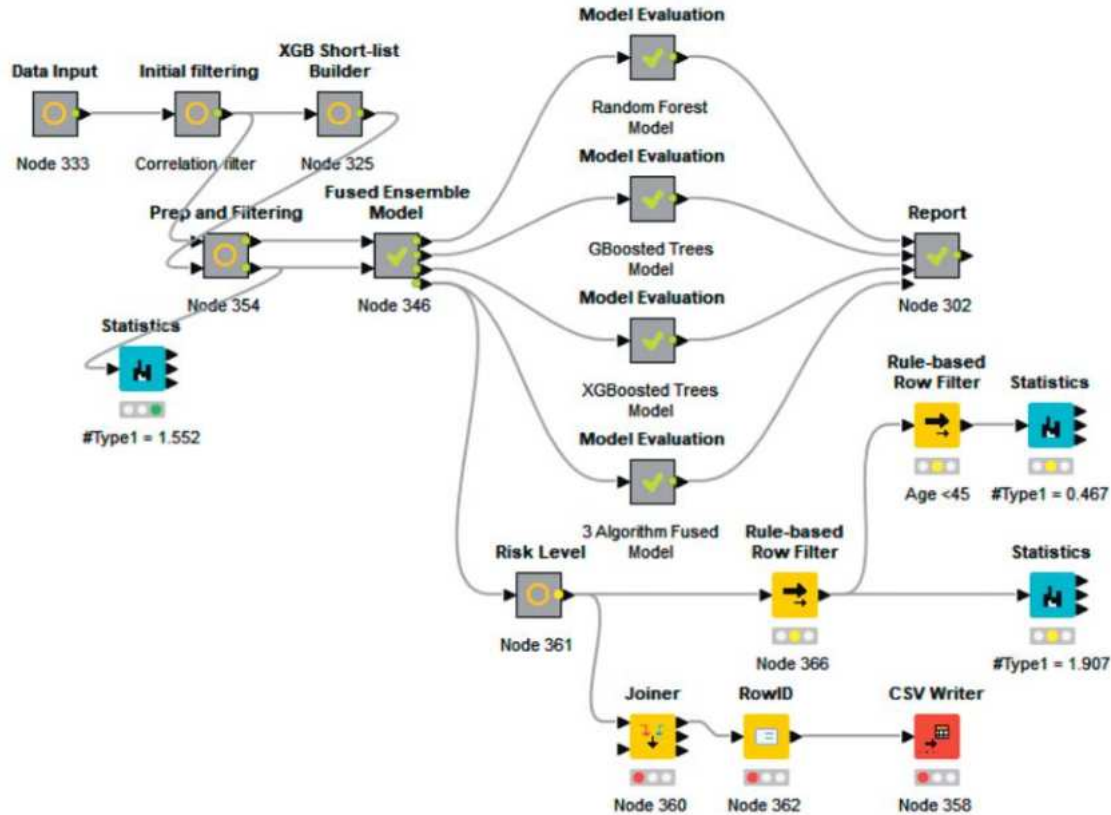


FIG. 23.3 The overall KNIME modeling workflow architecture.

metanode contains the three bagged modeling processes, plus the fusion of their results into a fused prediction for each patient. The Report metanode generates the report of the model prediction metrics. The Risk Level metanode of processing nodes produces precancerous colon polyp risk categories of Low, Medium, and High.

Machine learning modeling requires that the model be trained on a training data set and evaluated on a validation data set not used in any way during the model training process. The process of creating the training and validation data subsets is called *partitioning*. This function is performed in the Prep and Filtering metanode in Fig. 23.3.

The configuration of the Partitioning node sets the percentage of the whole data set (70%) to use to create the random sample for training and the validation percentage (30%) to be used to evaluate the predictive accuracy of the model. The problem associated with the partitioning process shown in Fig. 23.3 is that the model is trained on only part of the whole data set. If you reset the Partitioning node and run it again, it will train another model on another random sample of data to generate the Training and Validation data sets, and the results may be significantly different. The solution to this sampling problem is provided by arranging the modeling process to train a number of models on different data subsamples and combining all results using some heuristic (e.g., average, maximum, minimum, or median) to produce the final prediction for each data row. This processing architecture is called *bagging*.

Fig. 23.4 shows the Random Forest bagging loop, which is contained in the Fused Ensemble Model metanode. This workflow trains 10 models on different subsets ("chunks") of data and assigns the mode (most frequent) prediction value among the 10 models as the final prediction (the voting operation).

The voting operation in the Random Forest bagging loop shown in Fig. 23.5 is performed by the GroupBy node after the end of the bagging loop. This bagging process is similar to the cross-validation process that performs a sampling-based optimization of model

prediction accuracy in the model evaluation operation, instead of the model-building operation. Bagging was chosen as a sampling strategy (instead of cross-validation) to permit the workflow to be used in model deployment.

Results

Model accuracy

The results of this ensemble model were reported initially in Chapter 16 (Ensembles and Complexity). This chapter shows how those results were generated. The separate evaluation nodes in Fig. 23.3 calculate the accuracy metrics for each modeling algorithm. The Report metanode reformats the outputs of the evaluation nodes to produce the final report as shown in Table 23.2.

Sensitivity

The Sensitivity metric ranges from 0.0 to 1.0 and expresses how well the model predicts the Target = 1 values across all records in the validation data set. The formula for Sensitivity is:

$$\text{Sensitivity} = \frac{TP}{(TP + FN)}$$

where *TP* is the number of true-positive predictions and *FN* is the number of false-negative predictions.

Specificity

The Specificity metric expresses how well the model predicts the Target = 0 value across all records in the validation data set. The formula for Specificity is:

$$\text{Specificity} = \frac{TN}{(TN + FP)}$$

where *TN* is the number of true-negative predictions and *FP* is the number of false-positive predictions.

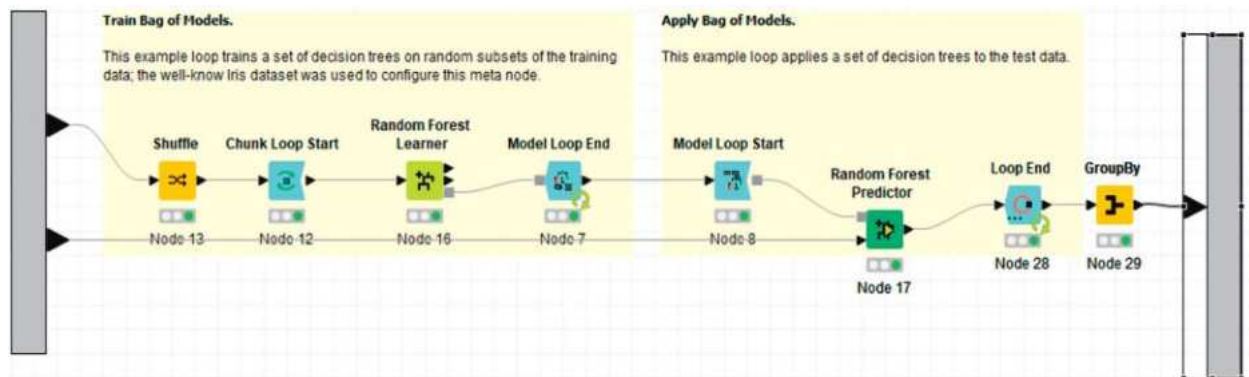


FIG. 23.4 The bagging loop for the Random Forest algorithm.

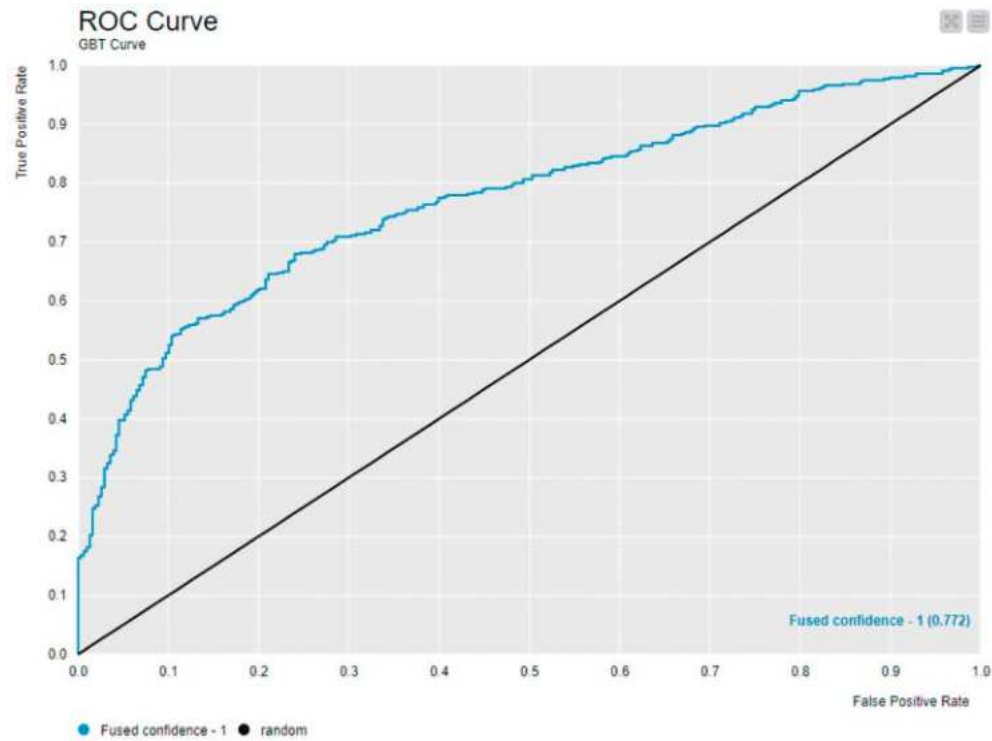


FIG. 23.5 The receiver operating characteristic (ROC) curve for the fused ensemble model.

TABLE 23.2 Prediction accuracy metrics for the fused model and the constituent models.

S Metric	S Value
Bagged Models with voting	
The Random Forest Model	
Sensitivity	0.789
Specificity	0.529
Accuracy	0.682
AUC	0.733
The Gradient Boosted Trees Model	
Sensitivity	0.77
Specificity	0.461
Accuracy	0.643
AUC	0.631
The XG Boosted Trees Model	
Sensitivity	0.748
Specificity	0.494
Accuracy	0.643
AUC	0.677
The 3 Algorithm Fused Model	
Sensitivity	0.807
Specificity	0.506
Accuracy	0.683
AUC	0.772

Global accuracy

Overall accuracy is equal to the weighted mean of Sensitivity and Specificity. Another formula for accuracy is:

$$\text{Accuracy} = \frac{TP + TN}{(TP + TN + FP + FN)}$$

Area under the curve

The area under the curve (AUC) is the total area under the receiver operating characteristic (ROC) curve. Fig. 23.5 shows the ROC curve for the Random Forest model.

The height of the *blue line* above the *diagonal black line* represents the predictive power of the model through the 10 deciles of response. The area below the blue line and above the black diagonal line divided by the total area above the diagonal black line represents the overall predictive power of the model. This number is called the AUC and is shown in parentheses at the lower right corner of the graph as 0.772, which corresponds to the AUC value in Table 23.1 for the Fused model.

Based on the results for models built with all three algorithms and the fused predictions shown in Table 23.1, it is judged that the fused model prediction will be an acceptable predictor of the presence of type 1 polyps in new patients.

Type 1 detection rate

The Statistics node on the left side of the workflow shown in Fig. 23.3 calculates that the mean number of type 1 polyps found for all data rows before the model was 1.552 (the type 1 polyp detection rate). The Statistics node on the right side of the workflow calculates that the type 1 polyp detection rate for all patients predicted to have type 1 polyps as 1.907, a 22% improvement in type 1 polyp detection rate produced by the model.

Discussion

The current recommended age for beginning regular colonoscopy exams is 45. This fused ensemble model is intended for use with people below age 45 to screen for the likelihood that they have type 1 colon polyps. The overall accuracy of the fused model is only moderate (0.68, or 68% accuracy); however, the Sensitivity accuracy (reflecting the true-positive accuracy) is 0.807, or 80.7% accurate. The lower overall accuracy is due to the combination of the Specificity accuracy (reflecting the true-negative rate) of only 0.506 or 50.6% accurate. The relatively poor ability of the model to predict negative examples correctly is due to the depressant effects of

the colonoscopy miss rate, which is estimated to be between 20% and 25% at the UCI Medical Center. Techniques of one-class classification are being explored to address this issue.

Another view of the predictive power of the model is the type 1 polyp detection rate compared before and after the model, which shows a 22% improvement in detection rate. This moderately large increase in type 1 detection rate lends credence to the conclusion that this model could be used as a preliminary screening method to provide the rationale for recommending that certain patients with a relatively high predicted risk for type 1 polyps have a colonoscopy, regardless of age.

Postscript

Colonoscopies are not fun. A common quip is that something is “about as much fun as a colonoscopy.” It is no wonder that people younger than the threshold age (45) for beginning colonoscopies avoid them as long as they can. Colon cancer is one of the most treatable cancers among humans if it is caught early enough. Predictive models like the colon polyp model can be used to justify to the patient that a colonoscopy is necessary, regardless of the patient’s age. Models like this can predict colon cancer purely by analyzing common clinical data available in most electronic medical records. These models have the potential to drastically reduce the death rate of colon cancer. This is a very practical outcome from a very technically complex procedure.

Keywords and phrases

Coloscopy, Polyp, Bagging, Colon cancer, Target variable, Feature selection, Detection rate, Ensemble model, Prediction accuracy, Binary variables.

References

- CDC. 2022. Colorectal cancer statistics. (<https://www.cdc.gov/cancer/colorectal/statistics/>)
- Siegel, R., Miller, K., Fedewa, A., S., Butterly, L., Anderson, J., Cercek, A., Smith, R., & Jemal, A. (2020). Colorectal cancer statistics. *CA: A Cancer Journal for Clinicians*, 70, 145–164.
- Wolf, A., Fonham, E., Church, T., Flowers, C., Guerra, C., LaMonte, S., Etzioni, R., McKenna, M., Oeffinger, K., Shih, Y. T., Walter, L., Andrews, K., Brawley, O., Brooks, D., Fedewa, S., Manassaram-Baptiste, D., Siegel, R., Wender, R., & Smith, R. (2018). Colorectal cancer screening for average-risk adults: 2018 Guideline update from the American Cancer Society. *CA: A Cancer Journal for Clinicians*, 68, 250–281.

Credit risk case study

Prelude

Credit ratings are one of the most central parts of our economy, pioneered by the Fair Isaac & Company (now known by the acronym FICO). The Retail Credit Company (RCC) began collecting data on millions of Americans. However, in addition to credit information, along with data on the private lives of consumers. When RCC planned to computerize this data, the government wouldn't allow it to happen.

In 1970, the Fair Credit Reporting Act (FCRA) was passed, requiring credit reporting bureaus (like RCC) to open their files to the public and delete data on race, sexuality, and negative information after a specified period of time. In 1975, RCC changed its name to Equifax. Over time, Experian and TransUnion were created, and these companies are now the top three credit reporting agencies. These three companies, however, had difficulty in comparing their scores. FICO, founded in 1956, subsequently developed an algorithm to generate a credit rating metric, now known as the FICO score.

FICO scores are used today along with other demographic data collected by the three major credit bureaus to build credit models to predict that the likelihood of default of a credit application is either "good" or "bad."

Introduction

This chapter will briefly describe three forms of credit models: (1) credit scoring, (2) prediction of credit risk, and (3) prediction of bankruptcy. The following case study will present the primary tasks necessary to build a credit risk model. This case study also will show how to perform several explainable AI (XAI). XAI techniques are discussed in [Chapter 18](#) also to induce business rules from the output of a "black box" MLP neural network algorithm.

Credit scoring

FICO and Vantage credit scores

A credit scoring model evaluates a number of variables to predict the worthiness of a person to receive credit in lieu of full payment for goods and services. There are many applications for which credit ratings are used to make financial decisions. [Fay \(2023\)](#) defines a credit score as a three-digit number that expresses the likelihood that a customer will pay their debts. There are two primary vendors of credit scores in America, the FICO score and the Vantage score (generated by a consortium of the three major credit bureaus, TransUnion, Equifax, and Experian). Both of these score vendors use the same credit score scale: 300–850. However, the vast majority of companies concerned about customer credit use the FICO score.

The FICO Score

The FICO credit score has become the primary source of information that financial institutions use to judge the creditworthiness of a potential borrower. The average credit score for buying a home mortgage is about 620 for most loan-granting institutions. The average credit score for buying a car is slightly lower; however, it is generally set at about 600. Other vendors of goods vary in their credit score requirements for buying on credit (usually financed through a credit union institution). [Fay \(2023\)](#) lists five primary factors used by FICO for calculating the credit score:

1. Payment history
2. Credit utilization
3. Credit history
4. Types of credit
5. New credit

Each of the factors is associated with a weight to reflect its business importance. The credit scoring model combines the values in the factors with their weights to calculate the credit score.

The Vantage Score

Factors in the Vantage model are:

High Weight: Payment History (40%) – On-time payment history is the top predictor of risk. Values for on-time payments are positive, while values for late payments are a negative and remain on a credit report for seven years.

Extreme Weight: Age and Type of Credit (21%) – The combination of credit payments is an important factor. For example, the history of on-time payments on a 30-year mortgage, a 5-year car loan, and monthly credit card bills results in a big positive value.

Extreme Weight: Credit Utilization (20%) – Positive values result when the ratio of balances to available credit is under 30% (0.3).

Medium Weight: Total Balances (11%) – Higher total credit balances result in lower credit scores and vice versa.

Low Weight: Recent Behavior (5%) – For newly opened accounts, a high number of hard inquiries lowers the credit score.

Extremely Low Weight: Available Credit (3%) – This factor is the amount of credit you have available to use. This factor alone is a predictive factor, but it has a relatively low weight in its application.

Other credit scores

Fay (2023) lists some other credit scoring models in use other than FICO and Vantage.

TransRisk – Produced by TransUnion, it determines credit risk on new accounts, rather than existing accounts. This credit score is not used by many vendors, and Fay (2023) claims that TransRisk scores for some individuals are drastically lower than their FICO score.

Experian's National Equivalency Score – This score ranges from 0 to 1000, calculated from factors of payment history, credit length, credit mix, credit utilization, total balances, and the number of inquiries. The Experian scoring system is not in parallel with the FICO score. A score of 100 means a 10% chance that at least one account will become delinquent in the next 24 months, while a score of 900 means a 90% chance of delinquency happening.

Credit Xpert Credit Score – This score inspects credit reports for approval of new account candidates. It looks for false information to discount and other ways to raise the credit score, and its use may lead to more loan approvals for some customers.

CE Credit Score – Provided by CE Analytics, it provides a free service available at Quizzle to counter the cost of credit union scores and to make the CE scoring

methodology very transparent. Quicken Loans uses this score for its customers.

Insurance Score – This score is calculated variously by different insurance companies, depending on the nature of their business. These scores may not correlate exactly with the FICO score, but customer premiums are based directly on the Insurance score, not the FICO score. For example, many fire insurance companies used to consider the frequency and severity of loss due to fire by modeling them separately and combining models to calculate fire insurance risk. Today, most fire insurance companies use a Tweedie analysis to express the probabilities of frequency of loss (which follows a Poisson data distribution) and severity of loss (which follows a log-normal data distribution). Tweedie-transformed probabilities are additive, while the raw probabilities of the two input distributions are not! The Python scikit-learn library provides a Tweedie function for this purpose.

Credit risk models

Credit risk is the probability of financial loss caused by the failure of repayment (default) on a loan from a financial institution. Credit risk modeling algorithms predict this probability of default to guide the decision to approve a loan to a given customer.

A high credit risk score can make it difficult to buy a car or a house or obtain a student loan. The problem is that it may be difficult for the credit risk modeler to map the score to the variables that predicted it.

Case study: credit risk score for a loan application

Recently, many banks and financial institutions that issue loans have adopted machine learning methods to predict credit risk scores to help decide whether or not to grant a loan. The General Data Protection Regulation (GDPR), in force in the UK since 2020, mandates that such credit score results be explainable in terms of predictor variables. Such legislation has not been passed yet in the United States, but it is coming; several such regulations are under current review. At a general level, Chapter 18 discussed several methods for explaining the results of machine learning models, termed XAI (explainable AI). This case study will present more details for several methods for mapping credit score results back to the variables used to predict them and estimating the variable (or factor) importance values.

Workflow: GC_data_RAN (available on the book web page)

Steps in model building with the GC_data_RAN workflow:

1. Input data: German Credit Scoring.txt from the UCI Machine Learning Repository.
2. New var derive metanode
 - a. The KNIME Statistics node was used to determine the balance status of the target variable (Creditability).
 - i Good = 70%
 - ii Bad = 30%
 - b. The KNIME Column Expressions node was used to split Marital Status/Sex variable into gender and marital_status variables.
3. The KNIME Partitioning node was used to split data into training set (70%) and testing set (30%).
4. The KNIME SMOTE node was used to oversample Creditability = "bad" category records to be equal to the Creditability = "good" record number.
5. Three algorithms were used to train models with the balanced training data set. Outputs of Sensitivity, Specificity, Overall Accuracy, and AUC were recorded for each algorithm. The AUC value refers to predictions or Creditability = good records, using the training set of records. The algorithms used were:
 - a. The KNIME Random Forest algorithm
 - b. The KNIME XGBoosted Trees algorithm
 - c. The KNIME RProp MLP (Multilayer Perceptron) algorithm
6. The KNIME Scorer node used to evaluate the prediction accuracy metrics for each algorithm

Model results

Prediction accuracies

Prediction results by the Random Forest model are shown in Table 24.1.

The Receiver Operation Characteristic curve

Fig. 24.1 shows the Receiver Operation Characteristic (ROC) curve for the Random Forest model.

The ROC curves for the XGBoosted Trees and MLP models are comparable to Fig. 24.1 and will not be shown.

Prediction results by the XGBoosted Trees model are shown in Table 24.2.

Prediction results for the MLP neural network model are shown in Table 24.3.

TABLE 24.1 Prediction metrics results for the Random Forests model.

S Metric	S Value
Sensitivity	0.773
Specificity	0.645
Accuracy	0.733
AUC	0.786

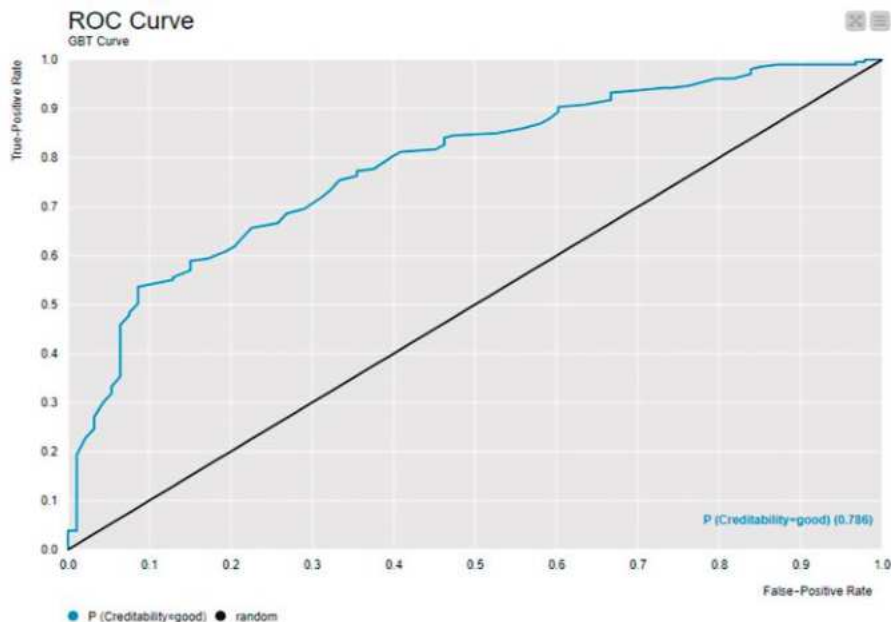


FIG. 24.1 The ROC curve for the Random Forests model. The AUC value for this curve (0.786) is displayed in the lower right corner of the plot.

TABLE 24.2 Prediction accuracy metrics values for the XGBoosted Trees model.

S Metric	S Value
Sensitivity	0.763
Specificity	0.624
Accuracy	0.72
AUC	0.768

TABLE 24.3 Prediction accuracy metrics for the MLP neural network model.

S Metric	S Value
Sensitivity	0.798
Specificity	0.488
Accuracy	0.713
AUC	0.705

Feature importance methods

The workflow contains three methods to evaluate feature importance: (1) number of times a feature is used in the first split (#split0) of building a tree in a Random Forest model; (2) calculation of the average information gain across all splits in building a Gradient Boosted Trees model; and (3) formation of a rule set from a decision tree algorithm used to predict the predictions of the MLP neural network model.

The normalized importance values for the Random Forest model are shown in Table 24.4.

Information gain – top 18 features across all splits in a Gradient Boosted Trees model

The normalized information gain values for the XGBoosted Trees model are listed in Table 24.5.

Building a ruleset from the decision tree algorithm

Some analytical toolsets like KNIME provide a facility to induce business rules from decision trees. A Multiple Layer Perceptron (neural network) model can be built to predict a target variable. Prediction accuracy metrics can be calculated from the confusion matrix output, but there is (usually) no way to determine predictor variable importance from the output. This situation would violate GDPR for use in the UK. Business rules can be induced from a follow-on modeling operation with a decision tree algorithm. The variables have been recoded

TABLE 24.4 Normalized importance values for the Random Forests model.

S Feature	D Importance
Duration in months	1
Balance of current account=	0.833
Amount of credit=	0.778
Foreign worker	0.667
Payment of previous credits=	0.444
Age in years	0.333
Purpose of credit	0.278
Value of savings or stocks=	0.278
Installment in % of disposable income	0.222
Most valuable asset=	0.222
Further debtors/Guarantors=	0.167
Type of apartment	0.111
Number of previous credits	0.111
Has been employed by current employer	0.056
Further running credits=	0.056

TABLE 24.5 Normalized information gain values for the XGBoosted Trees model.

Balance of current account=>=200 DM	1
Living in current household for=>great...	0.21
Payment of previous credits=problem...	0.148
Further running credits=no further ru...	0.148
Duration in months	0.128
Further debtors/Guarantors=none	0.125
Payment of previous credits=paid ba...	0.124
Purpose of credit=household appliances	0.088
Occupation=executive/self-employed...	0.088
Has been employed by current emplo...	0.085
Value of savings or stocks=between ...	0.084
Value of savings or stocks=no savings	0.081
Purpose of credit=new car	0.079
Value of savings or stocks=between ...	0.078
Living in current household for=betw...	0.075
Balance of current account=<= 200 DM	0.074
Purpose of credit=retraining	0.071
marital_status=married	0.071

for use by the MLP neural network algorithm to obviate the necessity of normalizing the input data. The recoding operation permits the induction of business rules that are properly interpretable. The top four business rules that follow were induced from the decision tree model.

Rule #1 Credit risk = good if:

\$ownership of house or land_Most valuable available assets\$ = No AND
 at department store or mail order house_Further running credits\$ = Yes AND
 \$>=200 DM_Balance of current account\$ = Yes

Rule #2 Credit risk = good if:

\$less than 1 year_Has been employed by current employer for\$ = No AND
 \$Amt_credit\$ < 3,000 DM AND \$no assets_Most valuable available assets\$ = No AND
 \$Duration\$ <= 24 months AND \$>=200 DM_Balance of current account\$ = Yes

Rule #3 Credit risk = good if:

\$paid back previous credits at this bank_Payment of previous credits\$ = No AND
 \$unskilled with permanant residence_Occupation\$ = No AND
 \$retraining_Purpose of credit\$ = Yes AND \$Age\$ < 40 AND
 \$no savings_Value of savings or stocks\$ = Yes AND
 \$Duration\$ < 24 AND
 \$>=200 DM_Balance of current account\$ Yes

Rule #4 Credit risk = good if:

\$Has been employed by current employers for between 1 and 4 years = Yes AND
 \$savings contract with a building society/Life insurance_Most valuable available assets\$ = No AND
 \$no savings_Value of savings or stocks\$ = Yes AND
 \$Duration\$ < 24 months AND \$>=200 DM_Balance of current account\$ Yes

Postscript

The FICO credit score is not only the most commonly used metric of creditworthiness but also a good example of how a data analytics company (FICO) can work hand in hand with government lawmakers by facilitating the passage of credit industry regulations that specify compliance defined in terms of the credit score. This integral cooperation permitted FICO to get a head start on other credit score algorithms by providing the only credit score methodology accepted by government regulators for many years. It is a powerful business enabler when the government requires analysis outputs that only your company can provide.

This business dynamic might be a good model for EU data analytics companies to follow in specifying regulations mandated by the proposed AI Act in the EU instead of fighting against them after the act is passed.

Keywords and phrases

Algorithm, Prediction, Factors, MLP, Credit score, FICO score, Vantage score, Business rules, Credit risk, Random Forest, Variable importance.

Reference

Fay, B. (2023). What is a credit score? <https://www.debt.org/credit/report/scoring-models/>

The intelligence revolution

Preamble

When ChatGPT was released in December 2022, many industry pundits announced that a new era in Internet browsing had begun. Since then, the hype has only increased. Microsoft announced that ChatGPT-3 was integrated with the new version of their Bing search engine. Subsequently, a virtual horde of apps like ChatGPT (kinds of large language models) have been built, many of them in the public domain. The next-generation Internet search engines will incorporate LLMs seamlessly to constitute Internet encyclopedias. Instead of return postings that contain subject matter related to the query, the return will be a summary of the subject matter itself!

Introduction

Humanity progresses over “ages” or “revolutions” over the decades and centuries. In recent centuries, we have experienced the “Early Industrial Age (Revolution)” starting in England in 1760 and in the United States in about 1860. The Late Age Industrial Revolution began in the United States after World War II in about 1950, followed by the “Information Age (Revolution)” of the past few decades. In early 2024, as this book is being published, we are now at the beginning of the swelling of the wave of the “Intelligence Age (Revolution).”

This Intelligence Age is characterized by terms and concepts such as:

- a. Large language models (LLMs) – such as ChatGPT-3 (and versions 3.5 and 4 subsequently) and other similar projects
- b. TinyML
- c. (HUMANE) – Personal AI (devices) which will replace the “Smart Cell Phone”
- d. FAB LABS – MIT professors, colleagues, and students run FAB LABS that can “build almost anything” using robots

Large language models (ChatGPT, BARD, etc.)

ChatGPT was originally developed by OpenAI; later, Microsoft became involved. It is one of the latest rages in AI. It holds the possibility of fundamentally changing peoples’ professional work and personal lives. It could (probably will ...) make existing web search engines like Google, Bing, Yahoo!, and DuckDuckGo obsolete in their current forms. There are early indications that these providers will need to have traditional search and chat coexist, side by side, to compete. For instance, Bing introduced Copilot in 2023. When ChatGPT was released in November 2022, its announcement created a large buzz in the AI field – it literally went viral (for more details on this, please see Section 7.3.1 I [Chapter 7](#) of [Burk and Miner, 2023](#), especially for how ChatGPT could create well-constructed code in programming languages like R and Python with no outside interference).

ChatGPT is in a family of LLMs; it is fine-tuned with both supervised and AI reinforcement learning techniques. An LLM is a type of neural network. This type of network architecture is called a *transformer*, which processes and generates text in sequence (see [Chapter 1](#) of this book for the section on LLM for additional information). Various groups, from OpenAI to Google to many others, have been developing their own versions of these LLMs during the past months, with the first one available to the public announced in November 2022 (ChatGPT – <https://openai.com/blog/chatgpt>).

Listing of some of the LLMs ([Islam, 2023](#)):

1. **ChatGPT** by Open-AI and Microsoft (*Note: GPT = Generative Pre-trained Transformer*)
 - **ChatGPT-3 has 175B** parameters
2. **LaMDA by Google**
 - has up to 137B parameters
3. **BARD** by Google
 - has up to 137B parameters (as based on LaMDA) but has the potential to provide up-to-date information unlike ChatGPT, which is based on data collected only up to 2021

4. **PaLM by Google**
 - with 540B parameters
5. **mT5 by Google**
 - 13B parameters but covers 101 languages and thus can perform well on many cross-lingual NLP tasks
6. **Gopher** from Deepmind
 - has 280B parameters
7. **Chinchilla** from Deepmind
 - with only 70 billion parameters but four times more data than Gopher
8. **Sparrow** from Deepmind
9. **Claude** from Anthropic
10. **Ernie 3.0 Titan** from Baidu
 - has 260B parameters
11. **Ernie Bot** from Baidu
12. **PanGu-Alpha** from Huawei
 - contains over 200 billion parameters, but model is based on 1.1 TB of Chinese-language sources, including books, news, social media, and web pages
13. **OPT-IML** from Meta AI
 - has 175 billion parameters
14. **BlenderBot-3 Meta AI**
15. **Jurassic-1** from AL21 Labs
16. **Exaone** from LG AI Research
17. **Megatron-Turing NLG** from NVIDIA
 - with 530 billion parameters, making it the largest and most powerful of its kind
 - NVIDIA's Megatron-Turing Natural Language Generation (MT-NLG) model is a transformer-based language model with 530 billion parameters, making it the largest and most powerful of its kind. (Islam, 2023)

Working with the publicly available versions of ChatGPT and BARD, the authors of this book did several things; among them, the following two are noteworthy:

We asked ChatGPT-3 to explain LLMs and include references in APA format. It did so beautifully! So did BARD. Following are the details on the ChatGPT response:

OpenAI ChatGPT

<https://openai.com/blog/chatgpt>

OpenAI's ChatGPT output to the query "Explain large language models with references in APA format"

Response:

"Large language models are a class of artificial intelligence models that use deep learning techniques to generate human-like text. These models are designed to process vast amounts of data and learn patterns from it, allowing them to generate text that is contextually relevant and coherent.

One of the most prominent large language models is GPT-3 (Generative Pre-trained Transformer 3), which was released in 2020. GPT-3 is a transformer-based language model with 175 billion parameters, making it one of the largest language models in existence. It was pre-trained on a massive corpus of text and can generate a wide range of outputs, including text completion, translation, summarization, and conversation.

According to a paper by Brown et al. (2020), GPT-3's performance on various natural language processing tasks is state-of-the-art, surpassing that of other language models on multiple benchmarks. The authors also note that GPT-3's impressive performance can be attributed to its large size and pre-training on diverse sources of text data.

Another large language model that has gained attention is BERT (Bidirectional Encoder Representations from Transformers). Developed by Devlin et al. (2019), BERT is a transformer-based language model that was pre-trained on a massive amount of text data and can be fine-tuned for a wide range of natural language processing tasks.

According to a paper by Liu et al. (2019), BERT's performance on various natural language understanding tasks is

also state-of-the-art, surpassing that of other language models on multiple benchmarks. The authors note that BERT's success can be attributed to its ability to learn bidirectional contextual representations of text, which allows it to capture a wide range of linguistic phenomena.

In conclusion, large language models are a class of artificial intelligence models that use deep learning techniques to generate human-like text. GPT-3 and BERT are two examples of large language models that have achieved state-of-the-art performance on various natural language processing tasks, thanks to their size and pre-training on massive amounts of text data."

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., & Amodei, D. (2020). Language models are few-shot learners. <https://doi.org/10.48550/arXiv.2005.14165>.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 4171-4186).

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. <https://doi.org/10.48550/arXiv.1907.11692>.

We also asked ChatGPT-3 to write a Python program to train a Random Forests algorithm with cross-validation. The result was a Python program that did *not* work in Python. We then asked BARD to do the same thing and the program worked! BARD accessed the Iris data set (*a commonly used small data set used in example programs of major statistical programs*) online, trained the model with cross-validation, showed the final model accuracy. We were impressed! Here is the Python code that BARD produced, interspersed with our explanations of what this code is doing:

Python code produced by asking BARD to train a Random Forest algorithm with cross-validation using the Iris data set:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import cross_val_score, KFold
from sklearn.ensemble import RandomForestClassifier

iris = load_iris()
X = iris.data
y = iris.target

# Create a Random Forest classifier
clf = RandomForestClassifier(n_estimators=100)

# Perform 5-fold cross-validation
kf = KFold(n_splits=5)
scores = cross_val_score(clf, X, y, cv=kf)

# Print the average cross-validation score
print("Average cross-validation score:",
      scores.mean())
```

This code will first load the Iris data set into memory. Then it will create a Random Forest classifier with 100 trees. Finally, it will perform five-fold cross-validation on the data set and print the average cross-validation score.

The output of this code will be something like the following:

```
Code snippet
Average cross-validation score:
0.9666666666666667
```

This means that the Random Forest classifier was able to correctly classify 96.66% of the data points in the cross-validation set. This is a good indication that the model is well trained and will generalize well to new data.

This program worked without modification. The actual average cross-validation score is:
0.9066666666666666

We later asked ChatGPT-version 4 to write a Python program to train a Random Forests algorithm with cross-validation, and it performed flawlessly.

This code generation capacity of LLMs is touted in several articles as transformation in computer science coding. Initial programs can be generated by an LLM (instead of by entry-level software engineers), then be modified and combined with other program elements to produce the final, end-product program for use in the real world.

Other examples of how LLMs are gaining information and answers in other avenues are pouring in daily. Recently, it was reported that an AI project translated 5000-year-old cuneiform writings (on tablets) at the push of a button ([Lidman, 2023](#)). Previously, it had taken years and years to translate just small portions of such writing because there are only a few hundred experts around the world that can decode the clay tablets. A group of Israeli computer scientists created an AI language program that could translate the ancient Akkadian (a language that has not been written or spoken for 2000 years) cuneiform. Because tens of thousands of such clay documents had already been digitized, they could be translated instantly when fed into this AI program. Libraries and museums have more than a half-million of these tablets, but only a fraction have been translated previously. Now these can be translated rapidly, as needed ([Lidman, 2023](#); [Guthertz et al., 2023](#)).

Students catch on quite fast when any new technology surfaces. After the November 22, 2022, announcement that OpenAI's ChatGPT was available for free use, students immediately started using it to write term papers and get answers to other problems posed by their teachers in classes. Recently, a group of California professors have looked at this trend and how it affects their teaching and testing/grading procedures ([Walker, 2023](#)). Here are some of the "take aways" from these professors' thinking as of early summer 2023:

1. While some professors focus on cracking down on "cheaters" (using ChatGPT for their homework), many more see that this technology is here, and it is better to assign work that allows students to use LLM/ChatGPT so they can see it as a "research tool," while at the same time gaining knowledge of its limitations and production of frequent errors (incorrect information, at least in its current version, i.e., ChatGPT-3).
2. Some professors emphasize to their students that this AI technology can be a "consultant" for students.
3. Some professors allow the use of AI-generated text in assignments as long as the students quote/cite the AI/LLM source, the same way that one would cite a "conversation with an expert."

4. Other professors forbid the use of AI in their classes. To check for this, these professors run every assignment turned in by students through three different AI checkers to ensure that the student actually put their words/formulas/results in their “own words.”
5. Many professors acknowledge that “Generative AI” is out there and will be present in the future and that it is best NOT to ignore it or ban it, as that would be detrimental to everyone, but instead, embrace it and see how best to add this new technology to our arsenal of problem-solving methods.

TinyML

TinyML is a field of machine learning that focuses on the development and deployment of ML models on low-power, low-footprint, microcontroller devices. The devices that TinyML models are designed for are small devices that are power efficient and capable of running on battery power for years before the battery needs to be replaced. TinyML is really a subset of what is known as edge AI, but while TinyML is similar to edge AI, it takes edge AI one step further, making it possible to run machine-learning models on the smallest microcontrollers. As a result, TinyML is one of the hottest trends right now, with 2.5 billion TinyML-enabled devices estimated to reach the market in the next decade and a projected market value exceeding \$70 billion in just 5 years. TinyML uses many machine-learning algorithms, but the trend is moving to making use of deep learning methods (for more details, please see Section 7.3.2 of Chapter 7 in the book by [Burk and Miner, 2023](#)).

HUMANE

HUMANE is one of the “hottest” tech things in the world today and maybe rightly so because it has the potential to really usher in the “Intelligence age – revolution” with gusto, because the main device of HUMANE is a small wearable item, probably about the size of a small “campaign button” that can be attached to a lapel or above a shirt pocket. This small device, the AI assistant, called the Ai Pin, will be able to do things like:

1. Take phone calls
2. Translate languages using AI
3. Show up emails and messages
4. Tell you what foods to buy next to keep your pantry and refrigerator supplied with items needed for your “food restrictions or diet”
5. And many other things

The HUMANE device does not need to be paired with any other device. It interacts with the world the way you, a human, interact, e.g., the device hears and sees what you see. The HUMANE device does not need a touch-screen, but we simply talk to it. Tap on the device and ask it to give us an update on messages and our upcoming calendar events. We can ask it questions and it will provide answers similar to how OpenAI’s ChatGPT performs (but verbally speaking these answers back to us as we walk down the street and go about our business).

What is “HUMANE” (HUMANE) and how did it come about?

- a. Simply put, Hu.ma.ne (<https://hu.ma.ne/>) is a proposed wearable AI assistant developed by former executives of Apple. HUMANE could usher in a world without the need for smartphones ([Jose, 2023](#)). HUMANE, a company founded by Apple veterans Imran Chaudhri and his wife, Bethany Bongiorno, in 2017, has been rapidly forging ahead over the past few years, adding over 60 former top Apple employees to their team. Plus, as we write this in mid-2023, HUMANE has close to 170 employees, with job openings waiting to be filled. One of the main missions of HUMANE is to turn around the current phenomena of everyone walking down the street with their eyes glued to their cell phones and bring “back to humanity” the phenomenon of looking and seeing the people and places they are passing as we walk, and thus “bringing humanity back into our lives.”
- b. Practically, the device of HUMANE has the potential of changing our lives again, as prior revolutions have done, thus making our lives more efficient with more time to devote to things of each person’s greatest interests. As Author Gary Miner proofs this in June, 2024, he has just received his Ai Pin device from Human and is learning how to use it. This revolution has already begun.

HUMANE’s mission, as stated on their home web page (<https://hu.ma.ne/mission>), is:

Technology that improves the human experience and is born from good intentions.

Products that put us back in touch with ourselves, each other, and the world around us.

Experiences that are built on trust with interactions that feel magical, and bring joy.

HUMANE was founded on the principle that we all deserve more from technology.

And HUMANE’s “Story,” also on their home web page, (<https://hu.ma.ne/story>) states the following:

Imran Chaudhri is a designer, inventor and innovator. He spent over 20 years at Apple imagining and creating some of the world’s most beloved consumer products, like the Macintosh,

iPod, iPad, Apple Watch and iPhone. Imran is best known for his work on inventing the groundbreaking user interface and interactions on the iPhone, and is named as an inventor on thousands of patents. His work defines how the world interacts with technology, and is driven by his insistence on putting the human experience front and center in the design process.

Bethany Bongiorno led the teams behind some of Apple's most transformational products. As a Director of Software Engineering, she was responsible for all software project management for iOS and macOS and also played a key leadership role in the execution of critical projects such as the launch of the original iPad. Prior to her time at Apple, Bethany studied physics, worked in astrophysics research, and was a data management and software development consultant. Bethany is passionate about building and leading high performing teams to solve complex problems.

Together, Imran and Bethany envision a future that is even more intelligent and even more personal, and have committed HUMANE to building not for the world as it exists today but as it could be tomorrow. Rethinking, reconsidering, and remembering honest human connection in the context of computing, they seek to reshape the role of technology in our lives.

The HUMANE home web page states that "Good AI is HUMANE," and works on the following 10 principles of designing "Good AI":

1. Good AI is Trustworthy.
2. Good AI is Ethical.
3. Good AI is Transparent
4. Good AI is Innovative.
5. Good AI is Useful.
6. Good AI is Efficient.
7. Good AI is Human-centered.
8. Good AI is Accessible.
9. Good AI is Inclusive.
10. Good AI is Timeless.

The company had been operating in somewhat of a "secretive mode" for some time, but made headlines the last week of April 2023 when they presented a TED Talk, finally announcing to the world what they were up to. In this TED Talk, Chaudhri demonstrated its projector-based wearable AI assistant device for the first time. The projection is that this tiny device (about the size of a political campaign button of years past that one could wear on a lapel or above a shirt pocket) will be used as an AI-backed personal assistant, replacing all smartphones in the near future.

HUMANE had previously raised over \$100 million in 2021 and, in March 2023, raised \$250 million, becoming the only AI startup to do so without any products announced.

The nature of HUMANE and the exact way that its technology works has not yet been revealed and probably won't be until the company releases its first product. However, the company has filed numerous patents related to HUMANE, which provide some hints of its operation. The patent documents mention the phrases,

"wearable multimedia device," "cloud computing," and a "laser projection system."

Tiny to Large AI-FAB LABS

Neil Gershenfeld is the director of the MIT Center for Bits and Atoms. Neil grew up wanting to do "practical things," not being academically drawn, but his parents wanted him to go to college and become a professionally educated person. So he honored his parents and pursued a degree in physics, but always kept his mind focused on "how can we do something practical" with our knowledge. Being "practical" was "what he was made of." As a professor at MIT, he developed the "Fab Labs" concept, which has opened such labs around the world today. But recently, he and his students have been involved in developing robots, even miniaturized robots that in fact can reproduce themselves and even build larger, more complicated robots, so that an "end product" can be developed entirely by the "robot work force" (Fridman, 2023; Abdel-Rahman et al., 2022).

The physics and math of modular robotics systems is rather complicated. A short summary of the mechanical-algorithmic processes involved is presented in Abdel-Rahman et al., 2022:

Modular robotic systems built of reconfigurable components offer an efficient and versatile alternative to traditional monolithic robots. However, as modular systems scale up, construction efficiency is compromised due to an increase in travel time and path planning complexity. Here we introduce a discrete modular material-robot system that is capable of serial, recursive (making more robots), and hierarchical (making larger robots) assembly. This is accomplished by discretizing the construction into a feedstock of simple primitive building blocks which can be re-configured to create a wide range of functionality. The discretization significantly simplifies the swarm's navigation, error correction, and coordination. The component composition is supported by an algorithm to compile the building blocks into swarms and plan the optimal construction path. Our approach challenges the convention that larger constructions need larger machines to build them and could be applied in areas that today either require substantial capital investments for fixed infrastructure or are altogether unfeasible. Abdel-Rahman and colleagues introduce a discrete modular material-robot system that is capable of serial, recursive (making more robots), and hierarchical (making larger robots) assembly. This is accomplished by discretizing the construction into a feedstock of simple primitive building blocks combined with an algorithm to plan the optimal construction path and assemble the building blocks into functional units and swarms. (Abstract from Abdel-Rahman, 2022)

For the physical and mathematical details, please see Abdel-Rahman et al. (2022); this reference contains several graphic illustrations that can make grasping the

understanding of “how robots can build almost anything” easier.

For those readers who would like to explore further how “tiny robots can make larger robots and the robots can even reproduce themselves,” the following links on the “Fab Lab developments” may be very useful:

Neil’s Website: <http://ng.cba.mit.edu/>
 MIT Center for Bits and Atoms: <https://cba.mit.edu/>
 Fab Foundation: <https://fabfoundation.org/>
 Fab Lab community: <https://fablabs.io/>
 Fab Academy: <https://fabacademy.org/>
 Fab City: <https://fab.city/>

Neil Gerhenfeld’s work has even extended to the concept that this “robotic system” can literally “make almost anything” (Chandler, 2022; Clifford, 2017; Fridman, 2023). If this is true, then we should be able to extend this to performing any kind of data acquisition, data cleaning, and data analytics “automatically” just by asking our “robotic system” to do so, right? We will explore this a bit more in the next section of this chapter.

Data and Predictive Analytics as a “Personal AI” or “Company AI” that functions automatically with AI and AGI – the future is only 3–7 years away!!!

We are probably only 3 to 7 years away from having a “Personal AI” that will take over the need to carry a “Smart Phone,” as this new Personal AI will do everything the smartphone did, plus much more. (*Actually, as this book goes to press, and author Gary Miner is editing his chapter, he has already received his Ai Pin Device from HUMANE as of May 2024 - this age is already here, we authors did NOT predict accurately !!!!!*).

By combining the technologies of HUMANE with TinyML and things like the Fab Labs, “robots can build almost anything,” it seems pretty clear that within a very short time, humankind has the ability to have their “Smart AI Assistant” (the HUMANE lapel button device) do almost anything for wearer just by asking it, verbally, to perform what is needed or wanted. This includes development of a needed data file and analyzing that data by the methods that the person asks to be performed.

We have already demonstrated this ability ourselves by asking ChatGPT and BARD to perform specific data analytic methods. We asked relatively simple (for Predictive Analytics) things, but the LLM went out and found appropriate data sets to use and found the needed predictive analytic algorithms. Which performed essentially identical to what we authors have done ourselves using conventional software on the same data sets (e.g., KNIME, STATISTICA-TIBCO, R, and Clementine).

It may not be an overstatement to expect that this ability to perform Predictive Analytics automatically may happen in 3–7 years, using an a personal AI assistant in the form of a lapel button. Elements of this technology may become available even before this book is published!

The only things that might delay the onset of the public’s use of a personal AI assistant are things like copyright infringement, but foregoing this, the future is almost here, and what some call the Intelligence Age has already begun.

Issues related to LLMs that may delay the deployment to all in the society of “Personal AI Assistant” include:

1. Questionable accuracy of information
2. Copyright infringement
3. Privacy
4. Plagiarism
5. Bias in coverage

Among these five issues, the biggest seems to be copyright infringement of the documents accessed online to build the LLM responses. The new AI legislation being drafted in the EU (Artificial Intelligence Act) addresses concerns of data quality, transparency, human oversight, and accountability, but it does not (yet) address copyright issues – it may address this problem before it is passed. If not, we expect it to be amended soon to do so. See <https://www.weforum.org/agenda/2023/03/the-european-union-s-ai-act-explained/>

It is quite likely that Google has an agreement with document sources like Wikipedia (where content is provided and can be deleted by any citizen of the world) to permit the LLMs to use text in results that reference the source. We say this because we’ve seen in our ChatGPT and BARD use that a lot of the references returned are from Wikipedia.

Pro and Con concerns about the use of personal AI assistants:

Any (or at least almost any ...) advance of humankind has both the proponents and the naysayers. The proponents proclaim that our “lives will be revolutionized” all for the better, and the naysayers proclaim that “the world is about to end” or something of that nature.

- Pros:
 - MIT “Fab Lab” professors and students are attempting to create a “good environment” that is “open” and not controlled by government or corporate power but instead, becomes a real servant of humankind – of individual humans.
 - HUMANE’s mission is to bring back “human-ness” to society ... it sees the AI-Personal Assistant as “Good AI in Action!!!”

- Cons
 - Many YouTube/Podcasts/Twitter posts in 2023 are bringing forth all kinds of “end of the world” scenarios with the advancement of this AI revolution, but most of this seems to be FEAR and lack of knowledge of how “Smart AI” really works.
 - These YouTube/Podcasts/Twitter posts are also expressing how it will be used by a few, primarily for controlling the world.

But most of the Con concerns seem to be the result of FEAR and lack of knowledge on how “Smart AI” really works. Most technical advancements of mankind have the potential to be used for good or evil (e.g., nuclear energy), but there are all kinds of others, even including the Internet, smartphone, and current forms of Internet communication, including all social media).

Postscript

One prediction appears clear at this time: AI-based personal assistants like the HUMANE model will become commonplace in 3–7 years. Beyond that, not long afterward, we may see that we will have the ability to perform Predictive Analytics by simply posing a query to an Internet search engine. Many other wonderful things may happen that we can’t even imagine now. “The Intelligence Age” has already begun!

Keywords and phrases

AI, ChatGPT, BARD, Parameters, LLM, Python, Intelligence Age, Natural language, Personal AI, Deep Learning.

References

- Islam, Arham; February 22, 2023; Top Large Language Models (LLMs) in 2023 from OpenAI, Google AI, Deepmind, Anthropic, Baidu, Huawei, Meta AI, AI21 Labs, LG AI Research and NVIDIA; <https://www.marktechpost.com/2023/02/22/top-large-language-models-llms-in-2023-from-openai-google-ai-deepmind-anthropic-baidu-huawei-meta-ai-ai21-labs-lg-ai-research-and-nvidia/>.
- Burk, S., & Miner, G. (2023). *It's all analytics, Part 3, The applications of AI, analytics, and data science*. New York: Routledge Press and CRC Press.
- Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.

- M., Wu, J., Winter, C., & Amodei, D. (2020). Language models are few-shot learners. <https://doi.org/10.48550/arXiv.2005.14165>.
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics* (Volume 1, pp. 4171–4186). Human Language Technologies. (Long and Short Papers).
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. <https://doi.org/10.48550/arXiv.1907.11692>.
- Abdel-Rahman, A.; Cameron, C.; Jenett, B.; Smith, M.; and Gershenfeld, N. November, 2022; Self-replicating hierarchical modular robotic swarms; DOI:10.1038/s44172-022-00034-3.
- Chandler, D.; November 22, 2022; Flocks of assembler robots show potential for making larger structures: Researchers make progress toward groups of robots that could build almost anything, including buildings, vehicles, and even bigger robots. MIT News. <https://news.mit.edu/2022/assembler-robots-structures-voxels-1122>.
- Clifford, C., July 17, 2017; Elon Musk: “Robots will be able to do everything better than us”; CNBC-Makelt (<https://www.cnbc.com/2017/07/17/elon-musk-robots-will-be-able-to-do-everything-better-than-us.html>).

HUMANE

- Jose, B. April 26, 2023; updated May 1, 2023. Humane, the wearable AI assistant from former Apple executives, could usher in a world without smartphones. The Indian Express. <https://indianexpress.com/article/technology/artificial-intelligence/humane-ex-apple-executives-wearable-ai-assistant-8577660/>

Translating clay tablets/Akkadian to English

- Guthertz, G., Gordin, S., Saenz, L., Levy, O., & Berant, J. (2023). Translating Akkadian to English with neural machine translation. *PNAS Nexus*, Volume 2(Issue 5) May 2023. <https://doi.org/10.1093/pnasnexus.s/pgad096>.
- Lidman, M. June 17, 2023. Groundbreaking AI project translates 5,000-year-old cuneiform at push of a button. *The Times of Israel* (<https://www.msn.com/en-us/news/technology/groundbreaking-ai-project-translates-5-000-year-old-cuneiform-at-push-of-a-button/ar-AA1cFIEX?>).
- Walker, R. June 19, 2023. California Professors Test Out AI in the Classroom, Even as Cheating Debate Continues. Noozhawk: The Fresh-est News in Santa Barbara County. (https://www.noozhawk.com/california-professors-test-out-ai-in-the-classroom-even-as-cheating-debate-continues/?utm_medium=email).

TINY ROBOTS – Can Make Anything

- Fridman, L.; May 28, 2023; Neil Gershenfeld: Self-replicating robots and the future of fabrication; Lex Fridman podcast (<https://lexfridman.com/neil-gershenfeld/>).

Right models – luck & ethics of analytics

Chapter 26 to 28 discuss several important considerations that must be kept in mind when deploying analytical models. The goal of every data analytics project should be to develop a model that *can* be deployed. Many models just sit on the shelf because they are too expensive or even impossible to deploy. The modeling considerations presented in this part of the book should be included in preliminary planning for the development of the intended model. Failure to do so may doom the modeling project from the start.

Chapter 26 describes the concept of “good enough” models, even if slightly more predictive models can be built. According to this concept, the “best” model may be the one built with only 20% of the effort required to build the most predictive model, following the Pareto Principle.

Chapter 27 describes several ethical considerations that can be associated with analytical models. Managers should consider whether or not the intended model can be deployed from an ethical standpoint, not just from a technical standpoint. Failure to consider ethical aspects of certain decisions have led to multimillion-dollar legal settlements against some companies, such as the lawsuit against AT&T described in Chapter 11.

Chapter 28 chronicles one of the most pernicious errors in the history of analytics, which positioned the parametric statistical p -value as the criterion to accept or reject a hypothesis. In many cases, the initial hypothesis was not and indeed could not be identified. The p -value was used, even in these cases, to “decide” whether or not a given predictor variable was “important” or not. Based on its relative success in some data domains like agronomy during the 1930s and 1940s, the p -value became a shibboleth in other sciences as a measure of “significance.” The phrase “it has been scientifically proved,” touted by many reporting media, was almost always based on p -value results of analyses in data domains in which the data distributions violated egregiously many of the mathematical assumptions on which the p -value was based. Several review studies of many medical articles presented in the chapter show that there is about as much chance of their conclusions being correct as there is in predicting a heads or tails outcome of a coin toss. The chapter ends with a call to consider application of machine learning algorithms for analyses of highly nonlinear data sets composed of many interactive predictor variables.

The “Right Model” for the “Right Purpose” – when less is good enough

Preamble

The problem with seeking solutions to problems in the world is that it is not just more complicated than we think, it is more complicated than we *can* think. Consequently, our solutions are bound to be less “complicated” than the problems demand. Most often, our response is to oversimplify, and one of our strategies to simplify our task is to follow what we think are common perceptions. For example, many critics of early efforts by humans to fly in airplanes argued that if God had meant for us to fly, he would have given us wings. Today, we understand that our technological capabilities are every bit as much an enabler to do things as are our bodily appendages. The error of those long-ago critics of human flight is not in their understanding that we had never flown in recorded history but in their presumption that we *could not* fly. Not only is nature more complicated than we think it is, but our ability to apply human technology to understand it and harness it is greater than many can imagine.

The way in which we deal with most problems in the world follows that pathway of thinking in one way or another. Often, we are victims of our own narrow perceptions, and we substitute those narrow perceptions for the much more complicated reality of the situations. Rather than take the time to study and understand (at least in principle) all aspects of a situation, we jump to a solution to a problem, following a very narrow path through a decision landscape constrained by our assumptions and presuppositions. Even if the assumption is valid, we ignore the effects of many other influences for the sake of generating a solution in the required time frame. Generating acceptable solutions in a given time frame is commendable, but our usual way of doing it is not!

One of the common perceptions in predictive analytics is that *more is better*. This is expressed in the belief that

1. More is better
2. Efficiency or Sufficiency must be selected (this is a false dichotomy, discussed below)

More is not necessarily better: Lessons from nature and engineering

The Efficiency Paradigm

Efficiency is usually defined in terms that involve maximizing output and minimizing input. The best solution is often defined in terms of the most efficient solution; this idea is often referred to as the “Efficiency Paradigm.” This paradigm assumes that the goal of efficiency is to maximize output while minimizing input.

In statistical analysis, the Efficiency Paradigm is expressed to define an *efficient solution*, one that has a relatively small variance. One solution (e.g., an estimator) can be considered more efficient than another if the covariance matrix of the second minus the covariance matrix of the first is composed largely of positive numbers. When all the elements of the resultant matrix are positive, it is called a *positive semi definite matrix*. The most efficient solution is the one closest to a positive semi definite matrix.

This mathematical definition can be useful if the Efficiency Paradigm is correct in the context of the solution. But what if it isn’t? Many examples in the real world appear to violate this paradigm. For example, ecological succession occurs in a previously forested area when a highly efficient grass community (defined in terms of productivity per gram of biomass) is replaced by a less-efficient shrub community, which in turn is replaced by even less efficient forest community. The Efficiency Paradigm might still apply to the mature forest if efficiency is defined in terms of accumulation of biomass over time, rather than in terms of productivity rate. In that case, sufficient productivity occurs to permit crown closure and shading out of competing species, including highly productive grasses.

This approach to defining efficiency in terms of sufficiency is at the core of the current debate on the definition of sustainable agriculture (Falvey, 2004). Voices supporting sustainable agriculture maintain that we

should avoid thinking in terms of the false dichotomy of efficiency and sufficiency and embrace both. Rather than make sufficiency dependent on efficiency, we should turn the concept around and let efficiency be defined in terms of sufficient solutions, rather than solutions of maximum productivity. In this natural context, we might call this the Sufficiency Paradigm.

The Sufficiency Paradigm

We might even replace the Efficiency Paradigm with the Sufficiency Paradigm in business also. This paradigm shift in business should be reflected not only in the business goals of a company but in the business processes followed to achieve them. In the context of the theme of this book, we can prescribe this paradigm shift in the IT departments as a technical “specific” to enable and promote the development of the business organism.

If we follow the Sufficiency Paradigm in analytical data mart design and analytical solution development, it will change the way we create analytical models. It will lead us to accept solutions that are good enough (sufficient) to build the synergies and products of the business organism, which will maximize productivity *within* the constraints of our goals to promote long-term stability and growth.

Under the Sufficiency Paradigm, the best analytical solutions will not be defined solely in terms of maximizing financial productivity. Rather, they will be defined in terms of how well they work together with other business processes to enhance the cohesive action throughout the entire profit chain. This cohesive action permits the company to be proactive and adaptive to change, rather than reactive and hampered by it. This set of features is intrinsically *organic* rather than *mechanical*.

Embrace change, rather than flee from it

In his book *Bionomics: Economy as Ecosystem*, Michael Rothschild (1991) maintains that mechanistic organizations fear change because it happens faster than their rather rigid business processes can respond to it. Tom Peters (1988) recommends that companies design business processes to take advantage of change to evolve new market niches. Peters also recommends that information should flow freely to encourage (and spread the “contagion” of) innovation. This free flow will permit even bad news to travel fast and even encourage it to do so. Mistakes become like pain that the whole body feels, not just the receptor site. Sharing of this information by a “foot” in the business organism prevents the other “foot” from making the same mistake. The business

organism learns from mistakes and can be driven by these mistakes to evolve into a more successful state.

Decision making breeds “true” in the business organism

Often, analytical results may drive decision-making activities to design actions in remote parts of the organization. However, these decisions may be difficult or impossible to implement. For example, it may be very difficult (or impossible under constraints of time and budget) to create the Customer Analytic Record (CAR) in the production database. One reason for this is that it may be very difficult to access and integrate all detailed data sources used to create the CAR. For example, the CAR may contain data at the monthly time-grain, but detailed switch data may be necessary to build a customer cross-sell model in a telecommunications company like AT&T for a new home office service product. Another reason could be that no business processes exist in production operations to do the necessary data preparation to create the CAR. This “disconnect” between modeling and production operations represents a gap in the information flow pathway. This is why many analytical models just sit on the shelf in IT and are not implemented in production.

The business organism must have a decision flow pathway between analysis and production (sites of action) which is properly designed to transmit information quickly and efficiently. This is the *Digital Nervous System* (Gates, 1999).

Muscles in the business organism

Another very important system in the pathway leading to business action is represented by the business processes (analogous to muscles), which are properly trained to turn the decision information into action. These business processes at the site of action must be developed *before* they can receive the decision information and act on it. *Therefore, the analytical modeling solution must be reverse engineered from the point of action.* That means you must model the model development process from the back end in the business unit, rather than from the front end in IT. Implementation requirements must be designed for each step in the decision information pathway, starting with the action site and proceeding toward model development and data access. *Inputs* for each step must be coupled with necessary information *transforms* to generate the precise nature of the *outputs* of that step required as inputs of the next step in the process. This input-transforms-output engineering paradigm must permeate all analytical modeling steps from data access to deployment processes.

Top-down versus bottom-up solution architecture

The overall design of the IT network systems and business process for each step along the way is an expression of building the solution model from the **top down** (following **Plato**). The detailed design of the business and analytical processes at each step is an expression of building the solution model from the **bottom up** (following **Aristotle**). Along the way, compromises and assumptions must be made to “jump” over problems that would otherwise prevent information flow. If we try to build the solution model solely from the top down or from the bottom up, we will reach a point where we discover that we don’t know enough or understand enough to “link” the steps in the process. System engineers quantify these links in simple ways and call them “transfer functions.” This is the way complex systems are modeled in the real world.

What is a complex system?

A *complex system* is an organization of interconnected and interacting parts that as a whole exhibit one or more properties that cannot be described solely in terms of properties of the individual parts. We introduced these systems-level properties in [Chapter 1](#) as *emergent properties*. These emergent properties are not obvious and may not even exist in the set of properties of the individual parts. The example of the rainforest given in [Chapter 1](#) was used to illustrate how these emergent properties can be among the primary aspects of the complex system that permit it to exist and function as a system.

The Corporate Information Factory

The concept of the business organism can be viewed in the context of a complex system. Like the forest system which changes over time, the business organism can adapt to changing business conditions. In order to facilitate the building of analytical solutions in such an adaptive business organism, the entire decision pathway must be designed with such solutions in mind. This can be done very effectively in an Exploration Data Mart, dependent on an Enterprise Data Warehouse. The concept of the business ecosystem as a part of the Corporate Information Factory was introduced by [Inman et al. \(1998\)](#) to refer to the concept of the business ecosystem driven by a “brain,” composed partly of memory (the relational data warehouse), and partly of an analytical system served by information stored in denormalized form – the analytical data mart ([Fig. 26.1](#)).

This data mart, along with others designed for other reporting purposes, could be expressed either in logical format or physical format. A physical data mart is hosted on a separate system with database schemas designed to serve specialized purposes. The logical data mart is hosted on the data warehouse system and is implemented in the form of database “views” or just composed of a group of denormalized tables containing aggregate data suitable for creating the CAR.

In the process of creating a CAR in a bank, for example, the easiest place to begin is to build a logical data mart by designing tables, such as householding tables, with summary data aggregated at the account, individual, and household levels. This householding process is necessary for a business like a bank because the same customer can own multiple bank accounts, and each bank account can

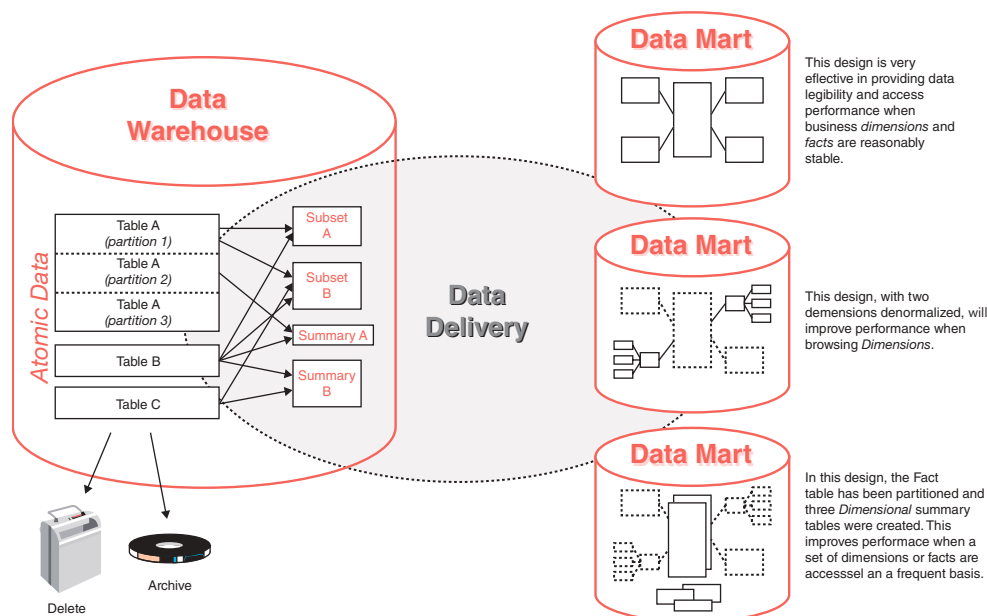


FIG. 26.1 The Corporate Information Factory.

have multiple owners. To focus modeling efforts at the customer level in a bank, data must be householded. Then the customer becomes the household and is labeled, possibly, with the primary owner’s ID. Other tables containing demographic data (for example) can be added to the logical data mart by joining other kinds of data with keys in common with the householding tables. This logical structure is well suited as a data source for many reporting and analytical operations.

But a properly designed data source is only one element in the decision chain. The other elements include:

- The IT network structure in a company is the digital nervous system, through which analytical solution results are communicated to the site of business action.
- The business processes (muscles), properly conditioned, move the service and product function of the company (the “bones”) to generate profit.
- The right actions are produced.
- Results of the actions (e.g., customer responses) generate more or less value for the company.
- Results are fed back to the business brain and provide a basis for learning how to build better models (the “Virtuous Cycle” of Berry and Linoff, 1997).

This learned response is characteristic of an adaptive organism, rather than a static machine. In the system composing the business organism, the decisions designed in IT breed “true” as they pass through the time steps in the process from one part of the system to another. That is, decisions remain unchanged in nature as they pass through various functions in the business organism through time. The reason they can remain unchanged is because they are designed correctly up front so they fit the decision-response pathway through each process from perception in the “brain” to the site of action in the “muscles” of the business organism.

The 80:20 rule in action

In 1906, an Italian economist, Vilfredo Pareto, observed that about 20% of the people owned about 80% of the wealth in the world. This principle was picked up in the late 1940s by quality management pioneer Joseph Juran, wherein he cast his concept of the *vital few and the trivial many* (Juran, 1951). Juran generalized this principle in quality management to postulate that 20% of something is always responsible for 80% of the results. He adopted Pareto’s principle to explain this and named it *Pareto’s Principle* or *Pareto’s Law* (aka the 80:20 rule).

The creation of sufficient analytical solutions may also follow the 80:20 rule. Certainly, the structure of the modeling process follows this rule in that about 80% of the

modeling project time is spent in data preparation and only about 20% in training and testing the model. Also, based on the 80:20 rule, we might expect to achieve a “sufficient” modeling solution in many cases, with only 20% of the effort we could spend modeling to create the solution with maximum predictability. There is some support for believing this in the concept of *Agile Modeling*.

Agile Modeling: An example of how to craft sufficient solutions

One of the most insightful approaches to modeling comes from the environment of Extreme Programming software development (XP). The premise of XP is to deliver the software the customer needs when it is needed. Niceties, enhancements, and other “bells and whistles” have to take a back seat to utility and timeliness. The approach of XP was extended by Scott Ambler to cover the “modeling” of the entire software development process, referred to as Agile Modeling (Ambler, 2002). One of the most important utility functions in Agile Modeling is the feedback loop to the stakeholders. Stakeholders are brought into the development process at key points in the project to validate the current state of the potential utility *in their perception*.

Ambler (2002) defines the concept of “just barely good enough” (JBGE) as the peak on the net benefit curve, beyond which continued effort results in diminishing returns. Fig. 26.2 shows the JBGE point on a net benefit curve. Ambler cites six propositions of Agile Modeling regarding the JBGE point, which pertain very closely to the development of analytical models.

1. JBGE is actually the most effective policy

a. JBGE is analogous to the inflection point on a curved response graph:

The JBGE point in Fig. 26.2 is the most “reasonable” position for effort to end. Assuming that additional effort could be spent on creating other JBGE models for other purposes, the optimum benefit across the entire modeling operations would restrict modeling efforts to the JBGE levels of effort.

2. JBGE does not imply poor quality:

The JBGE level of model production may not produce the highest accuracy, but it is sufficient to get the job done for which the model was commissioned. The stakeholders of the model, not the modeler, are the best judges of the utility of the model.

3. JBGE depends on the situation:

What is good enough for one situation may not be good enough for another situation. The classic example is in fraud, in which a 90% accurate model is good enough for most situations, but it is certainly *not* good enough for a fraud model.

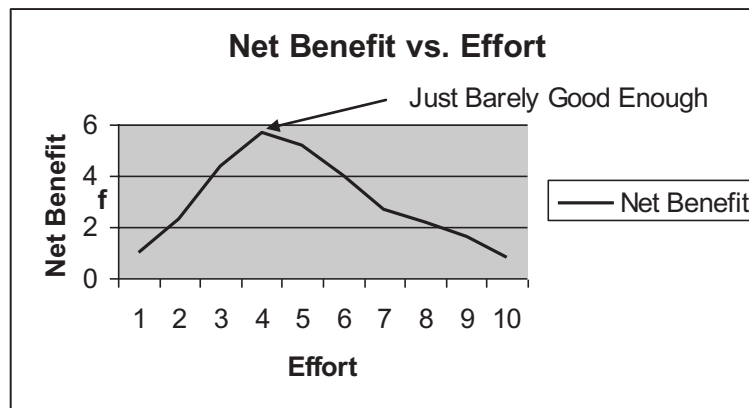


FIG. 26.2 Relationship between net benefit and total effort.

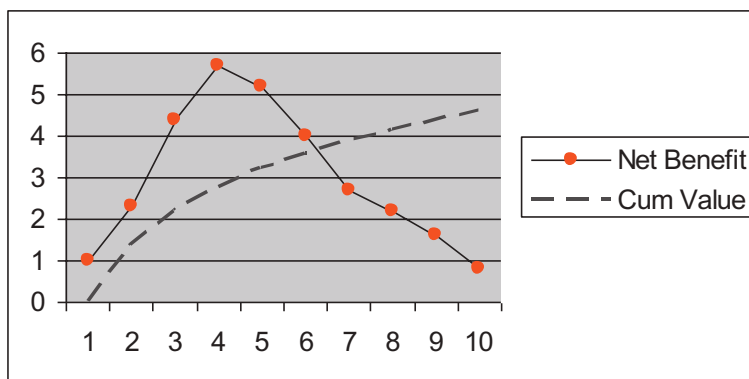


FIG. 26.3 The relationship between realized value (net benefit) and cumulative traditional value (Ambler (2002)).

4. The JBGE model evolves over time:

The initial model can be refined and/or updated over time. As needs and conditions change, the model can change. The characteristics of the model can adapt to new conditions. In this way, a model is like a biological species, which can respond to changing environmental conditions by changes in its very nature that enable it to adapt to new conditions.

5. The point of maximal benefit comes before you think it will:

In Fig. 26.2, the point of maximum net benefit occurs at about the 4th level of effort. It may appear counter intuitive that additional effort is associated with a decline in benefit, but when additional costs are considered, the 4th level of effort is best.

6. The realized value of a model may exceed the perceived value:

This statement may appear counter intuitive at first, but further consideration in the context of the business environment can clarify it. The traditional concept of value rises with additional effort, but much potential value can be masked by delays. The realized value of a timely model (even though it is not the most accurate possible) can far exceed that of an untimely model.

For example, a medical diagnostic model of only moderate accuracy delivered in time to define a successful outcome may be more valuable than a more accurate model delivered late, particularly if the patient dies in the meantime. We can see this dynamic expressed in Fig. 26.3.

The cumulative value shown in Fig. 26.3 represents the traditional view of defining value according to the accuracy of the model and the features included in it. Naturally, as effort increases throughout the development project, higher accuracy is achieved and more features are added. But the true utility value of the model may follow the curve of net benefit, rather than the curve of the cumulative value.

Timeliness and sufficiency in Agile Modeling

In addition to timely medical diagnoses, time itself is a variable to optimize. A glimpse into the future is provided by James Taylor (2007), who stresses this point in a discussion of smart enough logistics. He considered a delivery truck of the future delivering packages with RFID chips using trucks with GPS systems. The model-driven dispatch system would know what packages are on the truck and where it is at any given time. The

system might provide the driver initially with a route to follow, based on optimum delivery time. Later in the day, however, the dispatch system model might conclude from the present state of the system that the driver will not have sufficient time to deliver all of the packages and make a scheduled pickup. Therefore the system plots a new route for the driver and schedules another driver for the pickup. Such an agile system defines sufficiency partly in terms of timeliness of each delivery and pickup.

Postscript

This chapter (and indeed, the whole book) is designed to present the case that in many cases, “less” is good enough. In the book as a whole, we present very few equations. Rather, we present intuitive explanations of the concepts presented which are “sufficient” to enable the reader to understand enough of the theory and mathematics underlying the practice of predictive analytics to create acceptable models. Naturally, one would like to have models that are as accurate as possible. The definition of “possible,” though, must be composed of elements of time requirements and benefits relative to present methods in a time domain. As in the medical diagnosis example discussed above, it may be better to shoot for a model of lower accuracy done sooner than to wait for a more predictive model later. In this context,

our greatest challenge in analytical modeling is not finding ways to analyze data but deciding when less performance is good enough.

Key words and phrases

Efficiency, Sufficiency, Paradigm, Business organism, Business ecosystem, Data mart, Agile modeling, Benefit curve, Data warehouse, Efficient solution, Complex system.

References

- Ambler, S. (2002). *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process* (p. 400). Hoboken: John Wiley & Sons.
- Berry, M., & Linoff, G. (1997). *Data mining techniques for marketing, sales and customer relationship management* (2nd ed.). San Francisco: John Wiley & Sons.
- Falvey, Lindsay (2004). *Sustainability: Elusive or illusory? Wise environmental intervention*. Institute for International Development.
- Gates, B. (1999). *Business at the speed of thought*. New York, NY: Grand Central Publishing.
- Inman, W., Imhoff, C., & Sousa, R. (1998). *The corporate information factory* (p. 274). New York, NY: Wiley Computer Publishing.
- Juran, J. (1951). *Quality control handbook*. New York: McGraw-Hill.
- Peters, T. (1988). *Thriving on chaos: Handbook for a management revolution*. New York: Harper-Collins.
- Rothschild, S. (1991). *Bionomics: Economy and ecosystem*. New York: Henry Holt and Company, Inc.
- Taylor, J. (2007). *Smart (enough) systems*. New Jersey: Prentice Hall.

Ethics and data analytics

Preamble

This book has presented principles, methodologies, and examples for integrating predictive analytics and data science to many current applications. In the implementation of each of these applications, there are ethical aspects that relate to the question of whether or not to proceed and how to do it. We should integrate these ethical aspects into our decision-making operations for both practical and theoretical reasons. In a discussion of the ethical aspects of any subject, it is helpful to begin with the presentation of a concrete situation from the real world. The next step is to work backward to derive a list of best practices with reference to basic principles that connect to one or more worldviews to help determine the right things to do (Sandel, 2010). We will briefly review the secular systems of ethics that serve as a relevant background to ethical aspects of decisions enabled by data science studies of today. The three secular academic perspectives for making ethical systems and decisions are: (1) the normative, (2) the situational, and (3) the existential. This chapter will join a needed discussion on ethics in the new discipline of data analytics (Fung, 2015).

The birthday party – a practical example for ethical action

Imagine that a grandfather is at a birthday party with his grandchildren. In the course of the festivities, he takes some photos and videos on his own cell phone, and then he uploads them to Facebook and YouTube for the extended family and all the world to see. Is that the right thing to do?

The grandfather's action prompts a wide range of ethical questions in personal, business, and governmental contexts:

- Should the parents be concerned for children's safety (personal ethics)?
- Should the online platform be allowed to sell the visuals with meta tags (business ethics)?

- What data sharing should be permitted if the customer has "opted in?"
- Should the federal government be permitted to load the information into a super data mart for possible police or military investigations (government ethics)?
- What is the right balance of privacy and security?
- Can unfair biases be part of the logic of an algorithm?
- Should simulated images of people (e.g., Hollywood stars) be treated as public or private information?
- Should inputs of information to generative models from published sources be considered as plagiarism?

In the excitement of the moment, has the grandfather done the right thing by uploading the media recordings from the birthday party? By what standard do we judge this action in personal, business, and governmental contexts? What is the relevance of the grandfather's motivations? And are the goals in this situation reached in acceptable ways as right conduct?

Philosophers have debated questions like these for millennia. But let us begin with the use of ordinary language for some common ethical dilemmas and deliberations as we approach a systematic and consistent ethics. In personal life, should the pictures be uploaded? What personal private ethics should we use in this age of data science? In business, what public ethics are best? And in government, what legal ethics are best?

Secular ethics

In the realm of secular ethics, moral reasoning can be viewed from at least three major perspectives:

1. Normative-deontological perspective (like a code of conduct)
2. Situational-teleological perspective (like a panel of experts)
3. Existential-motivational perspective (like a moral code)

Normative-deontological perspective

This perspective focuses on universal standards for behavior in ethics. These “rules of the game” in a given context are designed to serve the highest “good.” One philosophy that has flowed out of this perspective is utilitarianism, which aims to generate the greatest happiness for the greatest number of people. Companies can follow this perspective by posting rules for all to follow to generate the most benefit for everyone.

Situational-teleological perspective

Rather than take an overall view, this perspective focuses on goals to be reached in a particular circumstance. Freedom is defined in terms of individual rights of liberty via laissez-faire or liberal fairness standards. This perspective includes both free-market libertarians standing for voluntary choices and egalitarians standing for equal chance of success. Freedom of choice is a primary ethic. A libertarian concept of a free market enables personal choice combined with the rule of nonaggression in interpersonal relations.

Existential-motivational perspective

This perspective focuses on one’s personal attitude. What is the right attitude in a given situation? Moral virtues are viewed as residing personally to control behavior, and ideas of the good life reside within the individual actor. This tradition of ethics goes back to attempts by the Sophists of ancient Greece to stand on human reasoning without reference to religion. From this perspective, ethics are based on conscience and the inward moral and religious principles for a just society in everyday life. Nevertheless, virtue becomes part of self-realization and an ingredient to promote the common good also. Behavior arises from affective intentions in the heart.

Standard discussions about ethics point to norms, motives, and goals, with varying weights of concern in different situations. These factors combine to produce policies, laws, and sanctions. Recent work by Rawls (1999), Sandel (2010), and Frame (2008) describe these approaches and their combination.

Normative-deontological ethics for data science – guiding the norms of government

For the normative approach in a secular context, the system of thought is foundational and built on natural law. This foundation consists of the abstract beliefs that underlie reality and personal duties. These fundamental beliefs may include the nonaggression principle and self-defense, for example. Disclosure is an important element of the implementation of those principles. However, there is little agreement with professional ethicists about the constancy of the duties required. Rawls (1999)

includes fairness as a key plank in the platform of ethical duties. Moral dilemmas for formal ethics in this context were popularized by psychologist Lawrence Kohlberg (1981). Normative policies for data analytics include fairness for all classes and not just the elite. Norms for some aspects of data science have been enacted in the European Union (EU) in the form of the General Data Protection Act (GDPR), and additional restrictions are under review in the proposed Artificial Intelligence Act in the EU. No similar regulations are currently in force in the United States, but several acts are under review.

Situational-teleological ethics for data science – guiding the goals in business

In the utilitarianism of John Stuart Mill, welfare aims to maximize utility to promote the greatest happiness for the largest number of people. This approach, however, is based on calculation rather than principle, and value is flattened without the qualitative differences of goods for individuals. This sets up a moral dilemma for a “just person” in the marketplace: should they maximize their own benefit or that of the general welfare? McCloskey (2006) has observed that a new rhetoric has enabled the compounding of business success because successful people are now encouraged by their significant others.

In business advertising, for example, the messages should be clear and formed to assist decisions, rather than be manipulative. With opt-in and opt-out capabilities, the viewer is able to move through commercial web pages and choose what effects they want. The ethical element here is that the customer is helped to achieve their situational goals, which may benefit all other customers too. This consequentialist effort is justified by meeting the most immediate objectives of the actors. Disclosure and transparency are important parts of ethical operations in business.

Existential-motivational ethics for data science – guiding norms of morality in business

From an existential-motivational perspective, ethics in data sciences can be seen to operate in at least three areas of life: business, governmental, and religious. Coherence among these areas of life could lead to productivity for self and others. Additionally, the moral dilemmas for interpersonal ethics remind us of the need for the personal dimension in all three areas (Haan, 1977). The existential perspective can lead to an extreme self-focus, without sympathy for others.

Michael Sandel on “Doing the Right Thing” with data analytics

These three perspectives on ethics at Harvard University are reviewed by Dr. Sandel in his landmark book, *Doing the Right Thing* (Sandel, 2010). Sandel has

been teaching ethics at Harvard since 1980. Importantly, he was a student of Charles Taylor of Oxford and McGill Universities, who has done landmark studies on the meaning of living in a secular age (Taylor, 2018). Both Taylor and Sandel have a long-term interest in the study of secularism as a cultural phenomenon in the West. This interest is the reason that Sandel reviews three secular perspectives rather than including a religious approach. His approach is combinatorial but leads with the existential and teleological perspective.

As a student of Taylor, Sandel encourages a more robust approach to the issue of worldview that includes the “subtraction definition” of the Secular Age. This approach describes the secular view as merely a subtraction of the past religious traditions in the West. We do call ourselves secular but cannot totally disengage with centuries of religious work on ethical systems from the past. From this standpoint, Sandel counsels secular ethicists to dialog with people from various religious backgrounds.

“Trolley Story” is used by Sandel as a starting point for moral reasoning in his lectures (Sandel, 2010). He begins his ethical discussion with a concrete (although hypothetical) situation involving a trolley driver. Five workers are at risk at the base of a hill on the trolley line. In one case, the dilemma for the driver is the decision to save the five by derailing the car while killing another person on the sidelines. A second case involves the choice of an onlooker to push another person from a bridge over the tracks to stop the trolley car. How do the three ethical approaches help with the driver’s decision? Welfare priority would look to save the most people with the least injuries. The situational perspective would look at the goals of all the actors. And thirdly, the virtues view would consider the attitudes of drivers, onlookers, and workers at the scene.

Marcus (2016) moves the situation from trolleys to driverless cars and must return to these same ethical questions. How does one weigh the safety of your passengers compared to yourself, bystanders, and others in this new mode of transportation? How does this factor into choices at the original purchase of the vehicle? How do these three ethical perspectives inform and guide the ethical decision making? Do they work best as a whole or one by one?

Moral reasoning is conducted to figure out what we believe and why. Deep learning processes are used with public articulation (open discussion) and social collaboration (a multitude of counselors). We look for the principles in the course of struggling with the scenarios. How should we reason our way through such moral dilemmas? What are the reasons for a course of action? Contradictions in our principles may arise in our recommendations. Cognitive dissonance is not easy to endure

but must be expected at times in the course of thinking about ethical dilemmas and decision points.

In time, the reasons or the recommendations may be revised due to the cognitive tension. Moral reflection is moving back and forth between principles and practices. How to be more than prejudiced is the goal of moral reasoning. Sandel (2010) maintains that it should be a public endeavor versus an isolated and individualistic puzzle. A similar zeal for collaboration is seen in the work of discovering new worlds with practical phenomenology (Spinosa et al., 1997).

Sandel (2010) also comments on practical ethical implications. He deduces that citizenship and national service are important as virtues. In the freedom perspective watching situational goals, not only utility and consent are highlighted, but moral limits of markets are marked too. Any investigation of the trolley car “accident” will be judged by utilitarianism within a situational approach. Virtues in the actor include valuing key social practices such as equality and solidarity.

Sandel (2010) goes on to say that there is a need to support and integrate socioeconomic classes. The social context of a well-integrated society is necessary to do the right thing more often. He recommends a strong program of public schools to ensure this context. To be taught in the right context is to seek mutual respect amidst value disagreements, if possible. We should attend to moral and religious disagreements directly. Engagement is required, rather than avoidance of ethical issues at large and among different groups. However, our public schools tend to dampen individual differences.

Alignment of ethical perspectives: The ethical road map

Considering the three perspectives of ethics, Davis (2012) presented a strategy for the application of ethics to data science in the workplace. He suggests four steps to define a pathway from values through practices used to define an alignment of the three ethical perspectives to guide ethical responses. He suggests that an audit of these principles be conducted alongside actual practices, and their separate influences be calibrated to define a synthesis. A road map can be constructed to watch actual performance in ethical dilemmas. The challenge is to anticipate and shape the next ethical decision points for Data Science along this road map in a step wise manner.

Step One: Inquiry: What are the values for principles and practices in an organization and its mission statement?

Step Two: Analysis: What are the data practices of an organization?

Step Three: Articulation: What is the alignment of practices with values in the organization?

Step Four: Action: What is the performance of this alignment of ethical practices at milestones in projects done by the organization?

These four steps are a good framework for developing ethical policies for data analytics and making real-time decisions. With time and persistent application, the organization can develop a culture of ethical policies and practices.

A good example of the application these four steps can be found in the blessing and the bane of Generative AI.

What is generative artificial intelligence?

One of the most visible and powerful artificial intelligence (AI) techniques coming to market is generative AI – machine learning methods that learn about content or objects from their data and use it to generate brand-new documents with previous content combined with new content (e.g., graphics). Several of the popular generative AI programs were introduced in [Chapter 25](#) (ChatGPT and Bard).

Generative AI is currently used for producing a variety of novel content, such as images, video, music, speech, text, software code, and product designs ([Gartner, 2023](#)).

Common uses today

- *Draft text output:* Staff writers can produce AI-generated text documents on various subjects and editors can edit them to produce a final copy.
- *Question answering:* Generative AI tools like ChatGPT and Bard enable users to pose questions via online apps and copy the results to a document.
- *Summarization:* Summaries and abstracts can be generated from input content for articles, emails, and web pages.
- *Simplification:* Outlines and keyword lists can be generated.
- *Software coding:* Draft code generation (see an example in [Chapter 25](#))

Gartner sees “...generative AI becoming a general-purpose technology with an impact similar to that of the steam engine, electricity and the internet” ([Gartner, 2023](#)). Generative AI can be used to create software code, accelerating drug development, and customer relationship management (CRM) applications. In 2021, [Gartner \(2021\)](#) projected that by 2025, he expected generative AI to account for 10% of all data produced, up from less than 1% in 2021. In 2023, Gartner’s projection was refined to predict that by 2024, 60% of data from AI applications alone will be synthetic from generative AI applications designed to simulate reality ([Gartner, 2023](#)).

Resolving ethical issues with generative models

The four steps of the ethical road map discussed previously (Inquiry, Analysis, Articulation, Action) can be followed to resolve some of the ethical issues associated with generative models.

1. *Plagiarism in schools*

- a. ChatGPT can write essays, movie scripts, and blog posts. Teachers are worried that students will use ChatGPT as a Wikipedia replacement to complete homework and write essays and term papers. The underlying problem with using this technology for that purpose obviates the need to develop the writing and researching skills required to create the document themselves and assignments for them, endangering students’ willingness to develop skills like writing and researching ([Shrivastava, 2022](#)).
- b. Some teachers are permitting students to write essay drafts using ChatGPT and edit them to form a final copy. This form of incorporation of new technology into current teaching methods is reminiscent of the introduction of hand calculators to mathematics and physics classrooms.

2. *Bias in decision making.* All information management systems and even the thinking processes of all people have a bias in them. Documents output by Large Language Models for use in a company or organization are biased by the nature and the extent of documents used to train the models and by the form of the query posed to generate them. Policies or controls should be instituted to detect biased outputs and deal with them in a manner consistent with company policy and any relevant legal requirements.

3. *Copyright infringement*

- a. Use of images of people downloaded from the Internet poses possible copyright infringement issues.
- b. Software like Stable Diffusion can be trained on copyrighted images scraped from the web, including blogs, news sites, and stock photo sites like Getty Images.
- c. The problem is that there is no legal structure available for judging the copyright infringement of these images.
- d. One image vendor, Getty Images, has banned the upload and sale of illustrations generated using AI art tools like DALL-E, Midjourney, and Stable Diffusion ([Vincent, 2022](#)).

4. *Cybersecurity and fraud.* Generative AI systems can be used for cyber invasion and fraud attacks. Image “deep fakes” can be used for social engineering of

personnel. Controls should be designed and installed to provide some measure of protection and mitigation of damages. Legal departments should investigate the degree to which existing insurance policies cover AI-related breaches.

Postscript

Chapter 26 discussed the value of building models that are “good enough,” rather than spending a lot more time building models of slightly higher predictive accuracy. This concern can be viewed as a matter of practical necessity in data analytics that should bind what is done. Chapter 27 discusses issues of ethical and moral concerns that may also bind what is done. This book ends with Chapter 28, which discusses some mathematical constraints on what we think should be done and how we should interpret results. All of these constraints in data analytics are part of the system (or the data “ecosystem”) that must be managed in a dynamic equilibrium such that balance points must be negotiated within concerns of ethics, morality, and practicality. The system is dynamic because these balance points are different for different projects; there is no single balance point among them. This view of the concept of a business ecosystem is analogous to the way natural ecosystems work. There is no single “balance of nature,” only a dynamic equilibrium among many natural ecosystem components and influences (Botkin, 1992).

Key words and phrases

Ethics, Morality, Principles, Secular, Situational, Plagiarism, Generative AI, Moral reasoning, Ethical roadmap, Ethical perspectives, Ethical dilemmas, Copyright infringement.

References

- Botkin, D. (1992). *Discordant harmonies* (pp. 256). Oxford, UK: Oxford University Press. p.
- Davis, Kord (2012). *Ethics of big data*. Cambridge, UK: O'Reilly.
- Spinosa, C., Flores, F., & Dreyfus, H. L. (1997). *Disclosing new worlds: Entrepreneurship, democratic action, and the cultivation of solidarity*. Cambridge, MA: The MIT Press.
- Frame, John (2008). *The doctrine of the Christian life*. Phillipsburg, NJ: P & R.
- Fung, Kaiser (2015). The ethics conversation we're not having about data. *Harvard Business Review* November 12.
- Gartner, “Gartner Identifies the Top Strategic Technology Trends for 2022.” October 18, 2021, <https://www.gartner.com/en/newsroom/press-releases/2021-10-18-gartner-identifies-the-top-strategic-technology-trends-for-2022>. Accessed January 1, 2023.
- Gartner, Gartner Identifies Top Trends Shaping the Future of Data Science and Machine Learning. August 1, 2023. <https://www.gartner.com/en/newsroom/press-releases/2023-08-01-gartner-identifies-top-trends-shaping-future-of-data-science-and-machine-learning> Accessed August 9, 2023.
- Haan, Norma (1977). *Coping and defending: Processes of self-environment organization*. New York: Academic Press.
- Kohlberg, Lawrence (1981). *Essays on moral development, Vol. 1: The philosophy of moral development*. San Francisco, CA: Harper & Row.
- Marcus, Amy Dockser (2016). Driverless cars spur questions on ethics. *The Wall Street Journal* June 24.
- McCloskey, Dierdre, N. (2006). *The bourgeois virtues: Ethics for an age of commerce* (Volume 1). Chicago: IL: The University of Chicago.
- Rawls, John (1999). *A theory of justice*. Cambridge, MA: Belknap Press of Harvard University Press.
- Sandel, Michael (2010). *Justice: What's the right thing to do?*. New York: Farrar, Straus and Giroux.
- Shrivastava, R. 2022. Teachers fear that ChatGPT will encourage cheating. *Forbes/Innovation*, December 12, 2022. <https://www.forbes.com/sites/rashishrivastava/2022/12/12/teachers-fear-chatgpt-will-make-cheating-easier-than-ever/?sh=7197ba031eef> Accessed: 3/13/2023
- Taylor, Charles. (2018). *A secular age*. Harvard University Press.
- Vincent, J. 2022. Getty Images bans AI-generated content over fears of legal challenges. *The Verge*, September 21, 2022. <https://www.theverge.com/2022/9/21/23364696/getty-images-ai-ban-generated-artwork-illustration-copyright>.

The significance of luck

Preamble

One of the greatest challenges we face in predictive analytics studies is the common perception that the phrase “it has been proved” means something; many times, it does not. The reason is that the traditional statistical analysis methods are based on the meaning and significance of the p -value. The p -values reflect the estimated probability that you are wrong in your prediction when you think you are right. The smaller the p -value, the greater the probability that you are right in your conclusion. The problem is that the theory underlying the p -value does not fit reality in the real world in most cases, but it is used anyway. This chapter will explore the nature of “statistical proof” and “accuracy” of predictive analytic models.

Introduction

Gartner (2017) thinks of predictive analytics as “an approach to data mining” that has four attributes:

1. An emphasis on prediction (rather than description, classification, or clustering)
2. Rapid analysis measured in hours or days (rather than the stereotypical months of traditional data mining)
3. An emphasis on the business relevance of the resulting insights (no ivory tower analyses)
4. An increasing emphasis on ease of use, thus making the tools accessible to business users

These four attributes appear somewhat biased toward business users, which is understandable when predictive analytics is used today in all kinds of endeavors. Whatever the domain, another attribute is important. Thus, another, fifth element will be added to Gartner’s four:

5. Significance (the “Real Accuracy”) of the model

The discussion in this chapter will focus on this fifth attribute.

The problem of significance in traditional p -value statistical analysis

According to one study, as many as 70% of medical research is so flawed in either experimental design or algorithm choice that there is as much probability of the results being correct as in flipping a coin (Murray et al., 2008). Based on studies like this, the controversy over the proper use of p -values in studies utilizing traditional statistics has increased dramatically since 2009. Nowhere is this more apparent, important, and even critical than in medicine. On the one hand, the traditional statistical tool of p -value was adopted to assist with reproducibility and allow comparison of studies by different clinical investigators. On the other hand, the pursuit of “ p -value perfection” can have tragic consequences. Siegfried (2015a) states that “...the standard statistical methods for evaluating evidence are usually misused, almost always misinterpreted and are not very informative even when they are used and interpreted correctly ...” These problems persist because the quest for “statistical significance” is mindless. Determining significance has become a surrogate for good research. The author concludes that “scientific studies are not as reliable as they pretend to be ... no more reliable than public opinion polls,” but such flaws could be fatal when relied upon in medical research and treatment.

Gigerenzer and Marewski (2015) state that the concept of statistical significance has become an idol, “worshipped” as the path to truth. It is advocated as the only “game in town,” and it is practiced in a “compulsive, mechanical way – without judging whether it makes sense or not.”

This publication was followed in July of 2015 by a second part (Siegfried, 2015b), in which he claims that

Statistics is to science as steroids are to baseball – addictive poison. But at least baseball has attempted to remedy the problem. Science remains mostly in denial. True, not all uses of statistics in science are evil, just as steroids are sometimes appropriate medicines. But one particular use of statistics – testing null hypotheses – deserves the same fate with science as Pete Rose got with baseball – banishment.

Testing of null hypothesis was the hallmark of scientific methodology of the past century. But it has also been the prime reason for making many research results irreproducible, if not erroneous. This has been particularly rampant in the medical sciences, apparently because most of the users in the medical sciences do not know how to apply traditional p -value Fisherian statistics properly to their research designs.

A following article rapidly followed on August 27, 2015 titled: “Psychology results evaporate upon further review,” which reported that only 35 of 97 “statistically significant results” could be replicated by a group of 270 researchers (Bower, 2015), also reported in Nosek (2015). Bower, (2015) used a complicated “replication/ reproducibility” analysis which sought to answer the question of whether the level of expertise of the replicating scientific team or the significance level of the p -value was more important in determining reproducibility; the initial study’s p -value significance level appeared to have precedence. Some of the results are illustrated in Fig. 28.1, which clearly shows that a majority of the studies could not be replicated.

Finally, Siegfried (2016) brought this issue to a head by claiming that misunderstandings about common statistical test damage science and society. By this time, the problems associated with the rather blind faith in p -values had been reported sufficiently that the scientific community was beginning to listen. Thus a “watershed” announcement was made by the American Statistical Association (ASA) during March 2016, stating clearly that “While the p -value can be a useful statistical

measure, it is commonly misused and misinterpreted ... this has led to some scientific journals discouraging the use of p -values, and some scientists and statisticians recommending their abandonment.”

The response of the American Statistical Association

The ASA had been aware of this problem at least 2 years previously in 2014, when they formed an internal group to study this problem with a goal of producing a document on the proper use of p -values for the guidance of researchers, practitioners, and science writers who are not statisticians. The results of this study group were finally published in June 2016 (Wasserstein and Lazar, 2016), in which they proposed six principles to guide the use of p -values:

1. p -values can indicate how incompatible the data are with a specified statistical model.
2. p -values do not measure the probability that the studied hypothesis is true or the probability that the data were produced by random chance alone.
3. Scientific conclusions and business or policy decisions should not be based only on whether a p -value passes a specific threshold.
4. Proper inference requires full reporting and transparency.
5. A p -value, or statistical significance, does not measure the size of an effect or the importance of a result.

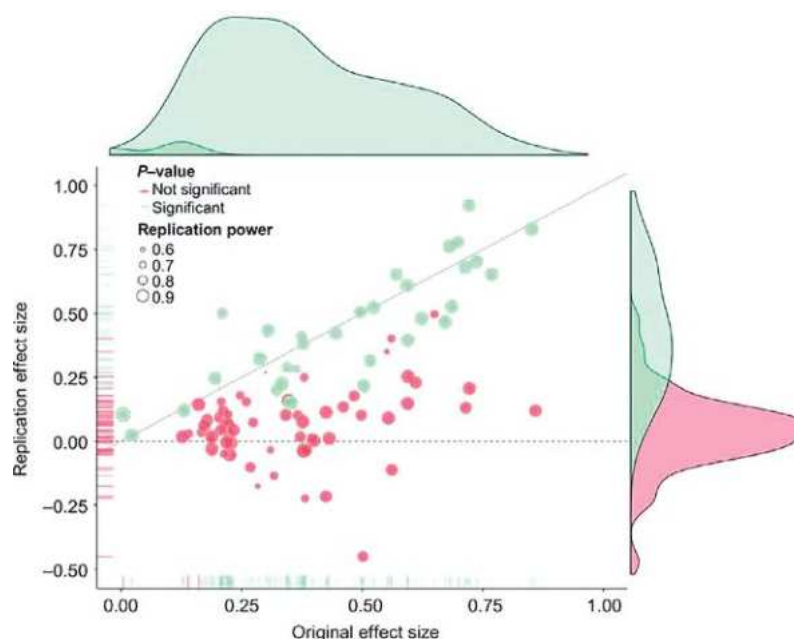


FIG. 28.1 Original study effect size versus replication effect size (correlation coefficients). Diagonal line represents replication effect size equal to original effect size. Dotted line represents replication effect size of 0. Points below the dotted line were effects in the opposite direction of the original. Density plots are separated by significant (green) and nonsignificant (red) effects (Nosek, 2015).

6. By itself, a p -value does not provide a good measure of evidence regarding a model or hypothesis.

The ASA study group's conclusion was that good statistical practice emphasizes principles of:

- Good study design and conduct
- Inclusion of a variety of numerical and graphical summaries of data
- Demonstration of an in-depth understanding of the phenomenon of the study
- Clear interpretation of results in the context of the problem
- Inclusion of a complete report with proper logical and quantitative understanding of what data summaries mean. No single index should substitute for scientific reasoning.

In the medical subdiscipline of radiology, Eklund et al. (2017) found that in using functional magnetic resonance imaging (fMRI), there was greater than a 70% error rate (false-positives) in identifying Alzheimer disease in over 40,000 scientific papers indexed by PubMed over the past 20 years. This does not mean that doctors cannot trust MRI results, but it does mean that one must question the validity of CONCLUSIONS of many studies using traditional parametric statistical techniques to diagnose Alzheimer disease. A good bottom-line consideration is that if the paper states "It has been scientifically proven," it is very likely to be false.

It is becoming increasingly clear that many research studies in medicine that are analyzed with traditional statistical methods cannot be replicated. This situation in medicine today is rather ironic, considering the fact that the need to replicate studies was the primary motivation for R.A. Fisher to develop his statistical analysis methods. Another interesting example of this comes from the lab of immunologist Dr. Tim Errington at the

University of Virginia's Center for Open Science (Feilden, 2017). Dr. Errington runs "The Reproducibility Project" which has attempted since 2011 to repeat the findings reported in cancer studies. He found that only two of the five "landmark cancer studies" in this project were reproducible – which means that conclusions of 60% of these so-called "LANDMARK" studies may be wrong! Fig. 28.2 shows some of the vials of test materials involved in one of these studies.

Replication is supposed to be a hallmark of scientific integrity. The concern over this lapse of scientific accuracy and integrity has been growing for some time, especially over the past 3 years, such that the University of Washington (Seattle) offered a new course in the spring of 2017 titled "Calling Bullshit in the Age of Big Data." The current version of the course is "Calling Bullshit: Data Reasoning in a Digital World." The goal of the class is to teach students and the public at large how to spot and refute BS wrapped in numbers, statistics, and algorithms. Among the learning/behavioral objectives for this course is the following: After taking this course, you will be able to provide a statistician or fellow scientist with a technical explanation of why a claim or conclusion is "in error." The syllabus for the original course is available from <https://callingbullshit.org/syllabus.html#Big%3B>.

This University of Washington course is being taught by Carl Bergstrom, a biologist, and Jevin West, a professor in UW's Information School. After the original course was announced, they woke up the next morning to chaos. They had 20,000 visitors to the course website, their mailboxes were full, and they were getting book offers. Bergstrom and West are longtime scientific collaborators who spent years grumbling about seeing inflated claims, manipulated algorithms, and twisted interpretations of scientific research, not only in the popular press but also in grant applications and scientific papers. Therefore they decided to put together this



FIG. 28.2 Some of the vials of testing biomaterials analyzed among the studies reported by Tom Feilden. Only two of five "landmark cancer studies" results were confirmed in replication studies carried out at the University of Virginia. (From Feilden, T. Most scientists "can't replicate studies by their peers." BBC News: Science & Environment, February 22, 2017. <http://www.bbc.com/news/science-environment-39054778>, <http://www.bbc.com/news/science-environment-39054778>.)

“fun class.” They apparently “hit a nerve” because the class, capped at 160 students, filled up within 1 minute of online registration. As a result, the materials are now available free online. The huge interest in and very large class size are indications of the concern people have about being able to recognize what is true and false. Many people in our society mistrust science as a provider of truth.

Postscript

Much of this mistrust of science is the result of misuse and misinterpretation of traditional p -value statistics, which were developed largely in the last (20th) century. Since about the year 2000, we have had at our disposal the availability of modern machine learning and other predictive analytic technologies that have the ability to analyze patterns in data more closely and make predictions that are more accurate. One of the central purposes of this book is to describe these new methods and show some case studies that used them. We hope that the result will lead analysts to embrace these methods and use them.

Key words and phrases

p -value, Scientific, Consequences, Replication, Statistical significance, Null hypothesis, Effect size, Predictive analytics.

References

- Bower, B. (2015). Psychology results evaporate upon further review. *Science News*. <https://www.sciencenews.org/article/psychology-results-evaporate-upon-further-review>.
- Eklund, A., Nichols, T., & Knutsson, H. (2017). Cluster failure: why fMRI inferences for spatial extent have inflated false-positive rates. *PNAS*, 113(33), 7900–7905.
- Feilden, T. (2017). Most scientists “can’t replicate studies by their peers.” BBC News: Science & Environment, February 22, 2017. <http://www.bbc.com/news/science-environment-39054778>.
- Gartner, 2017. <http://www.gartner.com/it-glossary/predictive-analytics/>.
- Gigerenzer, G., & Marewski, J. (2015). Surrogate science: the idol of a universal method for scientific inference. *J. Manage.*, 41(2), 421–440. <http://journals.sagepub.com/doi/full/10.1177/0149206314547522> <http://doi.org/10.1177/0149206314547522>.
- Murray, D., Pais, S., Blitstein, J., Alfano, C., & Lehman, J. (2008). Design and analysis of group-randomized trials in cancer. *J. Natl. Cancer Inst.*, 483–491.
- Nosek, B. (2015). Estimating the reproducibility of psychological science. *Science*, 349(6), 251. <http://doi.org/10.1126/science.aac4716>.
- Siegfried, T. (2015a). Science is heroic, with a tragic (statistical) flaw. *Science News*. <https://www.sciencenews.org/blog/context/science-heroic-tragic-statistical-flaw>.
- Siegfried, T. (2015b). Top 10 ways to save science from its statistical self: null hypothesis testing should be banished, estimating effect sizes should be emphasized. *Science News*. <https://www.sciencenews.org/blog/context/top-10-ways-save-science-its-statistical-self>.
- Siegfried, T. (2016). Experts issue warnings on problems with P values. *Science News*. <https://www.sciencenews.org/blog/context/experts-issue-warning-problems-p-values>.
- Wasserstein, R. L., & Lazar, N. A. (2016). The ASA’s statement on p -values: context, process, and purpose. *Am. Stat.*, 70(2), 129–133. <http://doi.org/10.1080/00031305.2016.1154108>.

A

Example of data mining recipes using Windows 10 and STATISTICA 13

Linda A. Miner

Professor Emeritus, Southern Nazarene University

This tutorial demonstrates the Data Mining Recipes module in version 13 of STATISTICA for Windows 10. The required hotfix has been applied to the application. The tutorial also demonstrates what might happen if there is a labeling error in the data, which could happen from time to time.

Open the data set

1. Click on the File tab at the top of the screen.
2. Select the Open option and browse to the storage location of "MidWest Company Data Copyright 2004 Right Brain Inc with text label error.sta".
3. See a portion of the data set in Fig. A.1.

Open the Data Miner Recipes interface

1. Click on the Data Mining tab at the top of the screen.
2. Select the Data Miner Recipes option, as shown in Fig. A.2.

The first screen of the Data Miner Recipes interface is shown in Fig. A.3. Click on the *New* option to start a new Data Miner recipe.

Screen #2 will display, as shown in Fig. A.4. Click on the Open/Connect data file box.

Opening the data set

1. Click on the Files box shown in Fig. A.5 and navigate to where you stored the tutorial data sets.

2. Select the MidWest Company Data Copyright 2004 Right Brain Inc with text label error.sta STATISTICA data set.

Click on Select variables, as shown in Fig. A.6.

Notice the highlighted box in Fig. A.6 to be clicked.

Fig. A.7 shows the variable selection screen.

Select the variables highlighted in the three columns.

Click OK, and an error screen pops up, as shown in Fig. A.8.

Evidently, there is one or more text labels in the Inferiority variable. Text labels are entries (strings of characters) that label specific numeric values within a variable in the spreadsheet. The Data Miner Recipes interface doesn't know what to do with the text label in the Inferiority variable.

For now, just exit the Data Miner Recipes by clicking on the X in the upper right corner of the screen. Don't save the recipe and return to the display of the spreadsheet, as shown in Fig. A.1. We will look at the specifications for the variable Inferiority and see if it has any text labels.

Fixing the text label problem

1. Right-click on the Inferiority variables and select the Variables Specs option to display the screen shown in Fig. A.9.
2. Click on the Text Labels box highlighted in Fig. A.9 to show the window in Fig. A.10. There is a text level for the value 101.
3. Click on the Delete Row box on the right side of the display.
4. Click OK to return to the data set spreadsheet shown in Fig. A.1.
5. Click the File tab on the top menu and select the *Save* option to save the spreadsheet.

FileHomeEditViewFormatStatisticsData MiningGraphsToolsData

CutSelect AllCopyClearFormatDeleteClipboard/Data

ReplaceRepeatFindFind/Replace

Random ValuesCopy DownCopy RightFillStandardize

VariablesCasesVariablesCasesOLE ObjectInsertLinks

DDEOLEObject

Feature FinderOptions

Data: MidWest Makers Company Personality Data (11v by 155c)

	1 Group	2 Inferiority	3 Social Competence	4 Preference for Variety	5 Creativity	6 Desire to Dominate	7 Exhibitionism	8 Affiliation	9 Autonomy	10 Gender	11 Years with
1	Product Design	32.5	40	33.3	17	25	30	40	35 Female	Over 10 y	
2	Product Design	15	36.7	46.7	19	34	25	37.5	32.5 Male	Over 10 y	
3	Sales and Service	17.5	36.6	33.3	20	27.5	20	37.5	25 Female	Over 10 y	
4	Night Shift Assembly Line 2	23	32	41	20	13	20	41	33 Male	Over 10 y	
5	Sales and Service	27	37	33	20	25	25	42	35 Female	Less than	
6	Day Shift Assembly Line 1	35	44	49	20	30	35	42	20 Male	Over 10 y	
7	Product Design	32.5	33.3	43.3	21	36.32	20	42.5	32.5 Male	Over 10 y	
8	Consumer Research	32	31	31	21	32	21	33	31 Male	Over 10 y	
9	Day Shift Assembly Line 3	35	30	33	22	20	22	40	27.5 Female	Over 10 y	
10	Product Design	15	47	50	23	29.5	40	42.5	42.5 Male	6 - 10 yez	
11	Product Design	21	50	50	23	35	50	40	40 Male	6 - 10 yez	
12	Night Shift Assembly Line 2	42.5	30	33.3	25	20	15	30	35 Male	Over 10 y	
13	Day Shift Assembly Line 3	30	43.3	40	25	20	20	47.5	32.5 Male	Over 10 y	
14	Night Shift Assembly Line 2	20	40	40	25	25	25	40	35 Male	Over 10 y	
15	Day Shift Assembly Line 3	30	40	30	25	35	30	35	40 Male	Over 10 y	
16	Product Design	20	40	46.7	25	32.25	35	37.5	37.5 Female	6 - 10 yez	
17	Consumer Research	35	26	30	25	35	40	37.5	27.5 Female	Over 10 y	
18	Product Design	32.5	30	30	26	29	20	37.5	27.5 Male	6 - 10 yez	
19	Product Design	17.5	33.3	36.6	26	29	30	37.5	32.5 Male	1 - 5 year	
20	Product Design	15	40	43	28	25	35	45	35 Male	6 - 10 yez	

Ready

MidWest Makers Company Personality Data C1.V1 Product Design / Set OFF / Weight OFF / GAP / NUM / REG

12/15/2016 1:33 PM

FIG. A.1 A portion of the data set Tutorial A_MidWest Company Data Copyright 2004 Right Brain Inc with text label error.sta.

Statistics - Tutorial A_MidWest Company Data Copyright 2004 Right Brain Inc with text label error

File Edit View Insert Format Statistics Data Mining Graphs Tools Data Window Help

Data Miner Recipes

Word - Add to Workspace

Vars - Cases

155c

	5 Creativity	6 Desire to Dominate	7 Exhibitionism	8 Affiliation	9 Autonomy	10 Gender
1	17	25	30	40	35 Female	Over 10 y
2	19	34	25	37.5	32.5 Male	Over 10 y
3	20	27.5	20	37.5	25 Female	Over 10 y
4	20	13	20	41	33 Male	Over 10 y
5	20	25	25	42	35 Female	Less than
6	20	30	35	42	20 Male	Over 10 y
7	21	36.32	20	42.5	32.5 Male	Over 10 y
8	21	32	21	33	31 Male	Over 10 y
9	22	20	22	40	27.5 Female	Over 10 y
10	23	29.5	40	42.5	42.5 Male	6 - 10 yez
11	23	35	50	40	40 Male	6 - 10 yez
12	25	20	15	30	35 Male	Over 10 y
13	25	20	20	47.5	32.5 Male	Over 10 y
14	25	25	25	40	35 Male	Over 10 y
15	25	35	30	35	40 Male	Over 10 y
16	25	32.25	35	37.5	37.5 Female	6 - 10 yez
17	25	35	40	37.5	27.5 Female	Over 10 y
18	26	29	20	37.5	27.5 Male	6 - 10 yez

FIG. A.2 Selection screen for the Data Miner Recipes interface.



FIG. A.3 Screen #1 in the Data Miner Recipes interface, showing the selection of the *New* option.

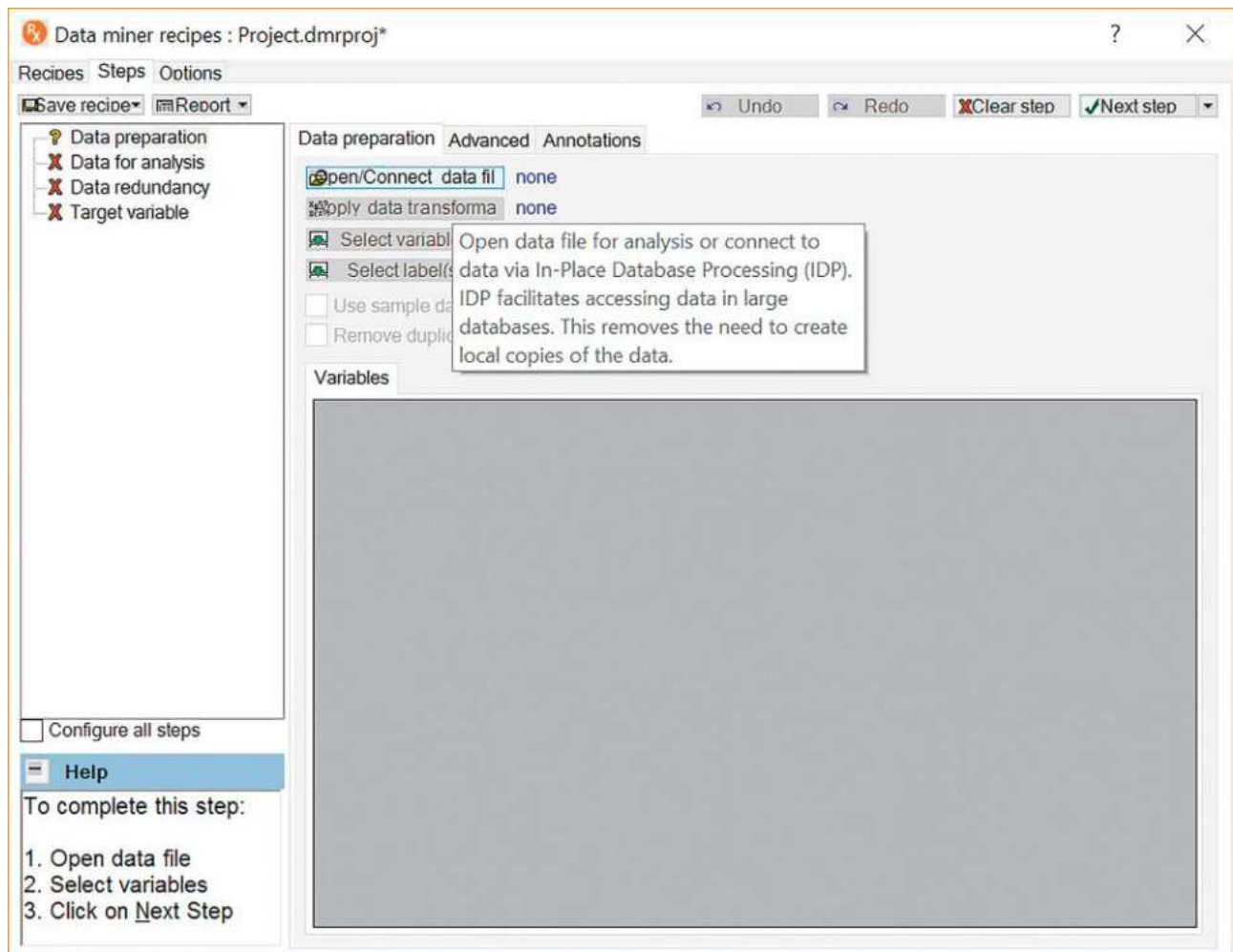


FIG. A.4 Selecting the option for loading the data set.

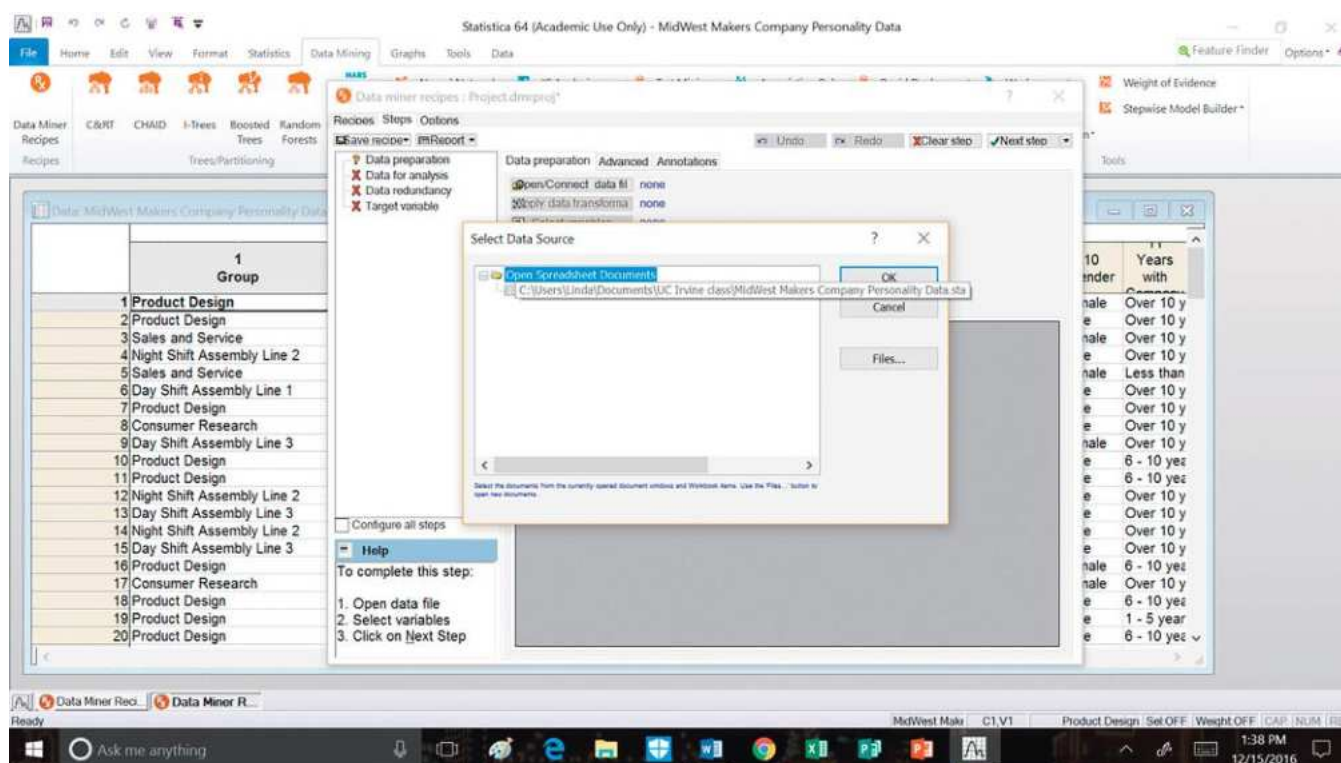
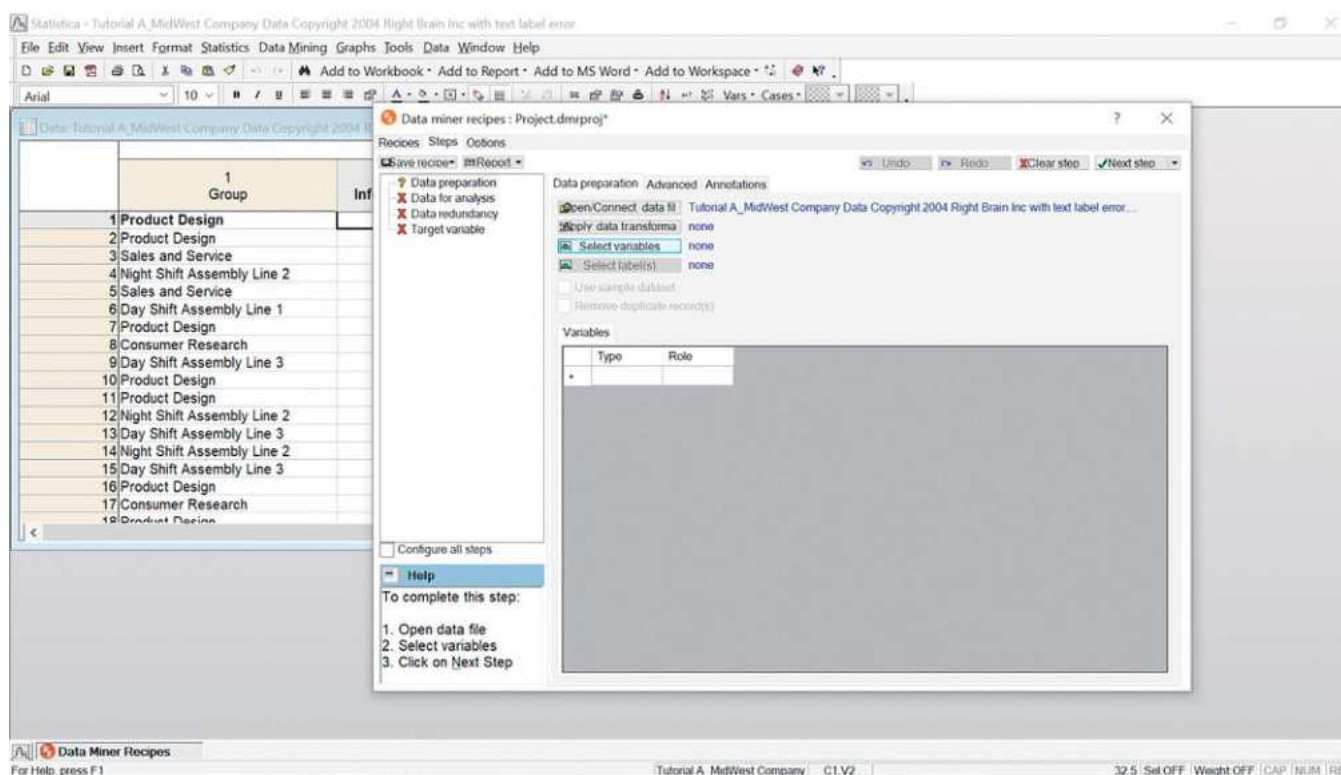


FIG. A.5 Selecting the location and the name of the data set to open.

FIG. A.6 Data Preparation option selection, with the *Select Variables* highlighted.

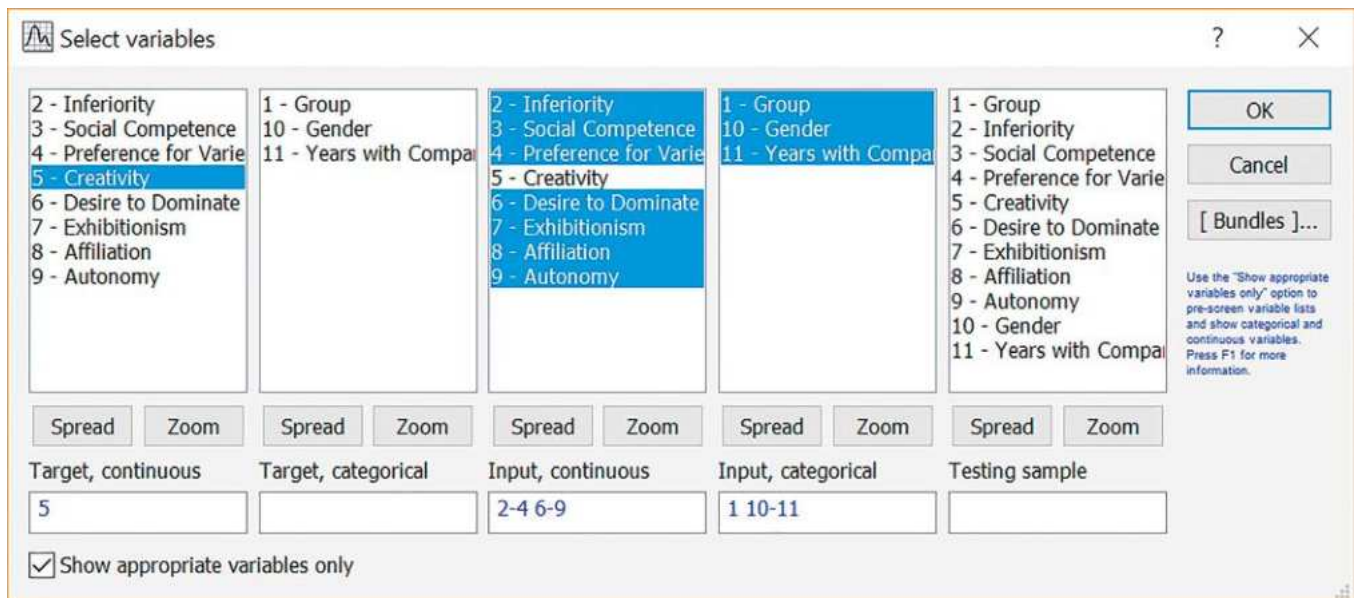


FIG. A.7 Variable selection screen, showing selected variables highlighted. (Note that “Show appropriate variables only” is checked, which prevents variables with an inappropriate data type from being displayed in a given column.)

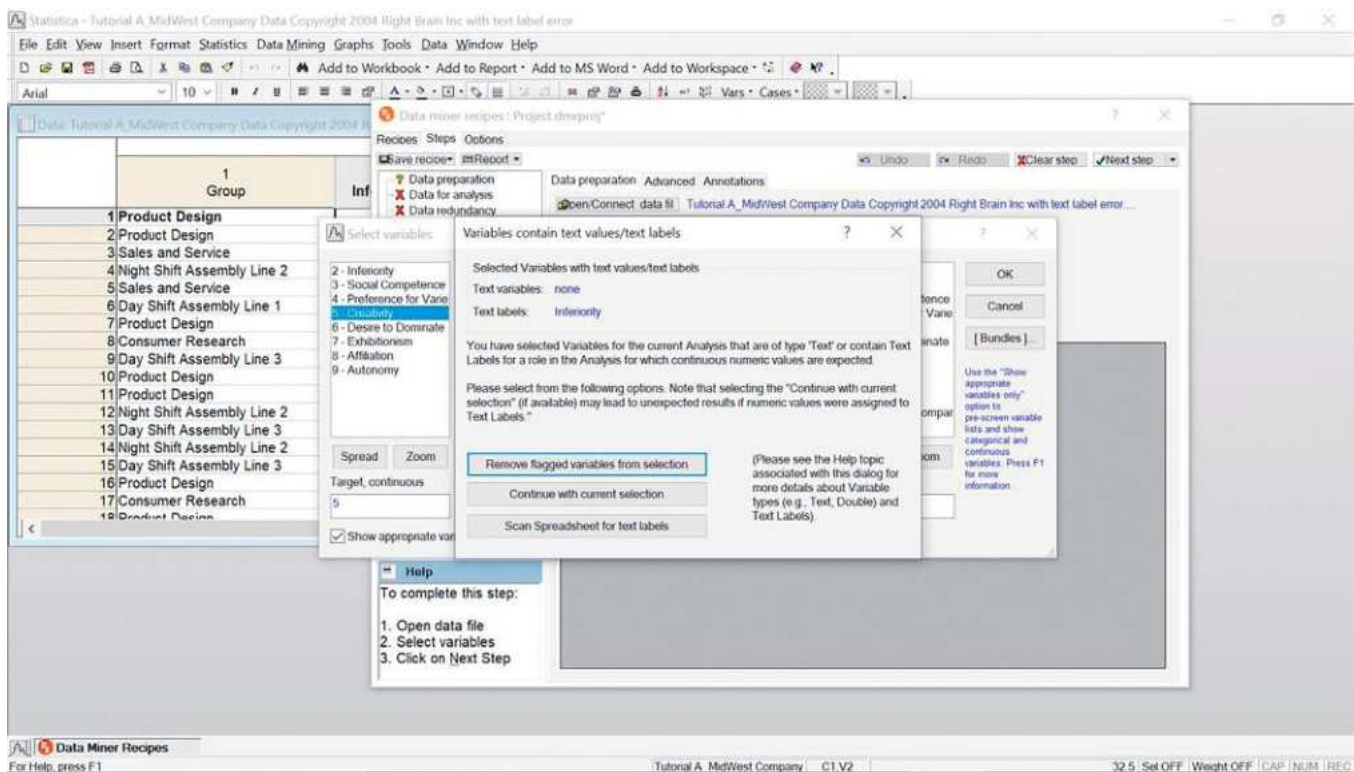


FIG. A.8 The error message popup.

Variable 2

Font: Arial, Size: 10

Name: **Inferiority** Type: Double

Measurement Type: Auto Length: 8

☐ Excluded ☐ Label ☐ Case State MD code: -9999

Display format

- General
- Number
- Date
- Time
- Scientific
- Currency
- Percentage
- Fraction
- Custom

Buttons: All Specs..., Text Labels..., Values/Stats..., Properties..., [Bundles]...

Long name (label or formula) **Functions**): ☒ Function guide

Labels: Use any text. Formulas: Must begin with an = sign. Use variable names or V1, V2, ..., V0 is the case number. Examples: (a) = mean(v1:v3, sqrt(v7), AGE) (b) = v1+v2; comment (after);. In case of conflict, variable names take precedence over variable text values. Specify text values by appending \$, as in "value\$".

FIG. A.9 Variable specifications for the Inferiority variable.

Data mining recipes (again)

1. Click on the Data Mining tab..
2. Select the Data Miner Recipes option
3. Click *New*.
4. Click the *Open/Connect data file* box.
5. Select the spreadsheet in the list (the only one listed is the one you saved).
6. Click the Select variables box and select the variables shown in Fig. A.7.
7. Notice that the error box shown in Fig. A.7 does not pop up this time because we deleted the text label for the Inferiority variable.
8. Fig. A.11 shows the updated analysis screen with a list of selected variables.
9. Click the Next Step box at the upper right of the screen, and a statistical data report will appear, as shown in Fig. A.12.
10. Click on the Next Step box to go to the data redundancy step. This step will eliminate one

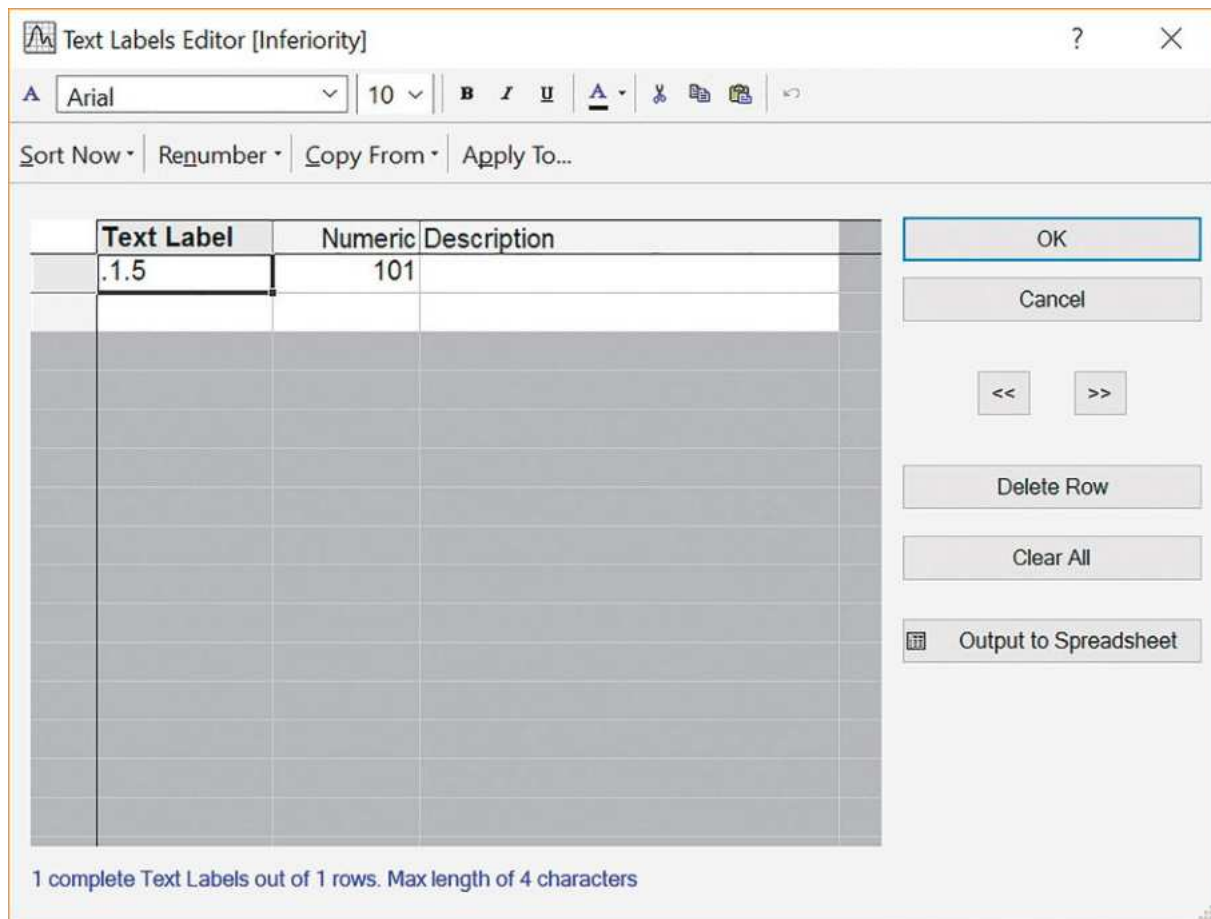


FIG. A.10 The erroneous text label entry.

- member of any pair of variables that have a correlation coefficient greater than 0.7.
- a. Click on the Correlation Coefficient option.
- b. Click the *Next Step* box.
- c. The program reports that there were no redundant variables.
- d. This check prevents any highly correlated pairs of variables from being used to predict the target variable. Such redundant variables can increase the probability of undertraining a machine learning model or preventing the mathematical processing (matrix inversion) of a parametric statistical analysis operation (e.g., linear regression).
- e. Click OK and the variable selection screen appears.
11. Click on the *Fast predictor screening* option to configure the variable selection screen, as shown in Fig. A.13.
12. Click on the *Next Step* box to see the Model building screen shown in Fig. A.14.
 - Notice in Fig. A.14 that the CR&T (Classification and Regression Trees) algorithm, the Boosted Trees algorithm, and the Neural Network algorithm are chosen by default to compose the ensemble model to be built by the Data Miner Recipes interface.
 - a. Click the *Next Step* box to build the ensemble model.
 - b. In less than a minute, the model will be trained, and some intermediate results will be displayed, as shown in Fig. A.15.
13. Without a deployment data set, the Data Miner Recipe interface cannot generate any objective prediction accuracies. The only accuracies that can be reported are correlation coefficient values shown in Fig. A.15.

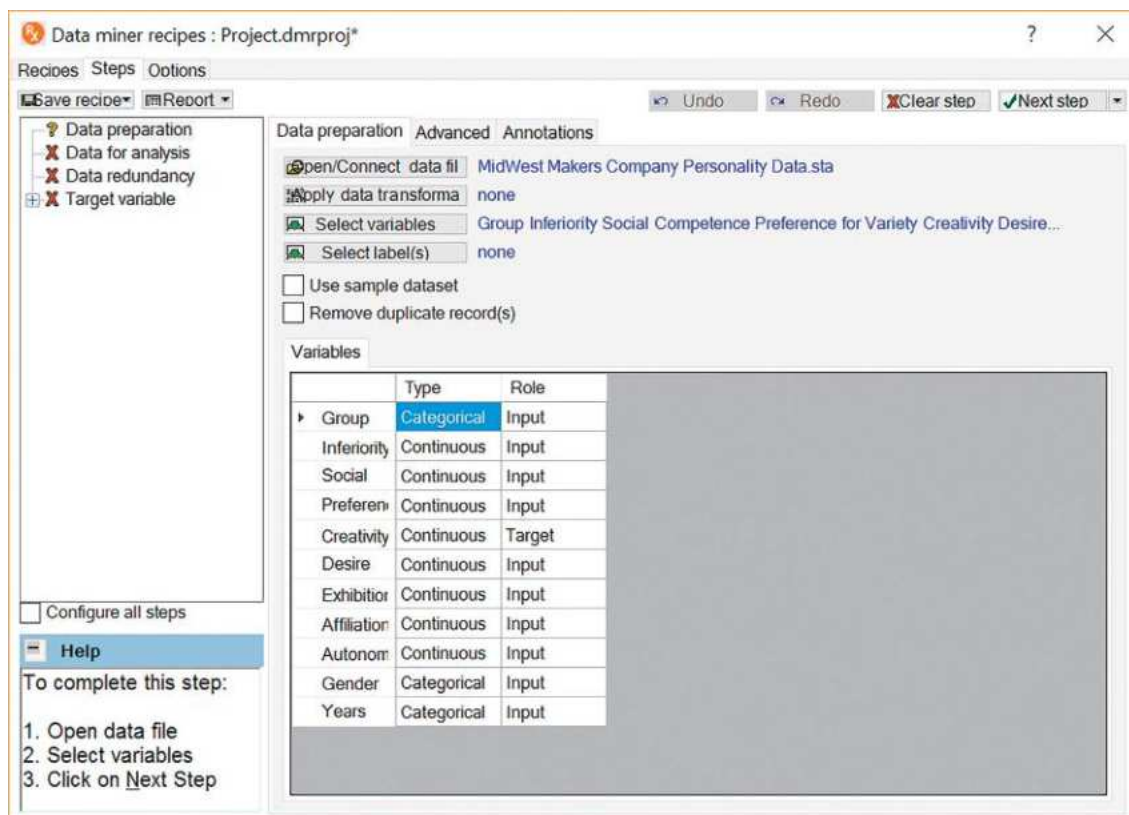


FIG. A.11 Data are loaded; variables are selected.

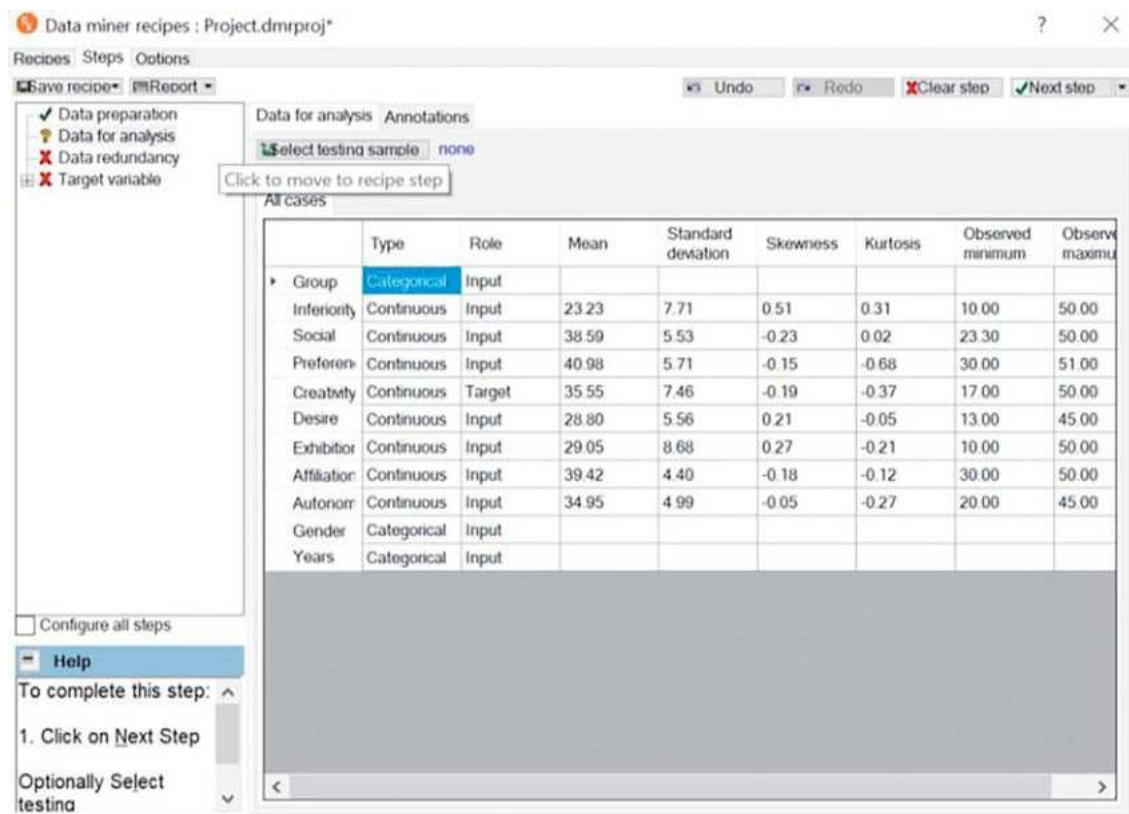


FIG. A.12 The descriptive statistical report for the data set.

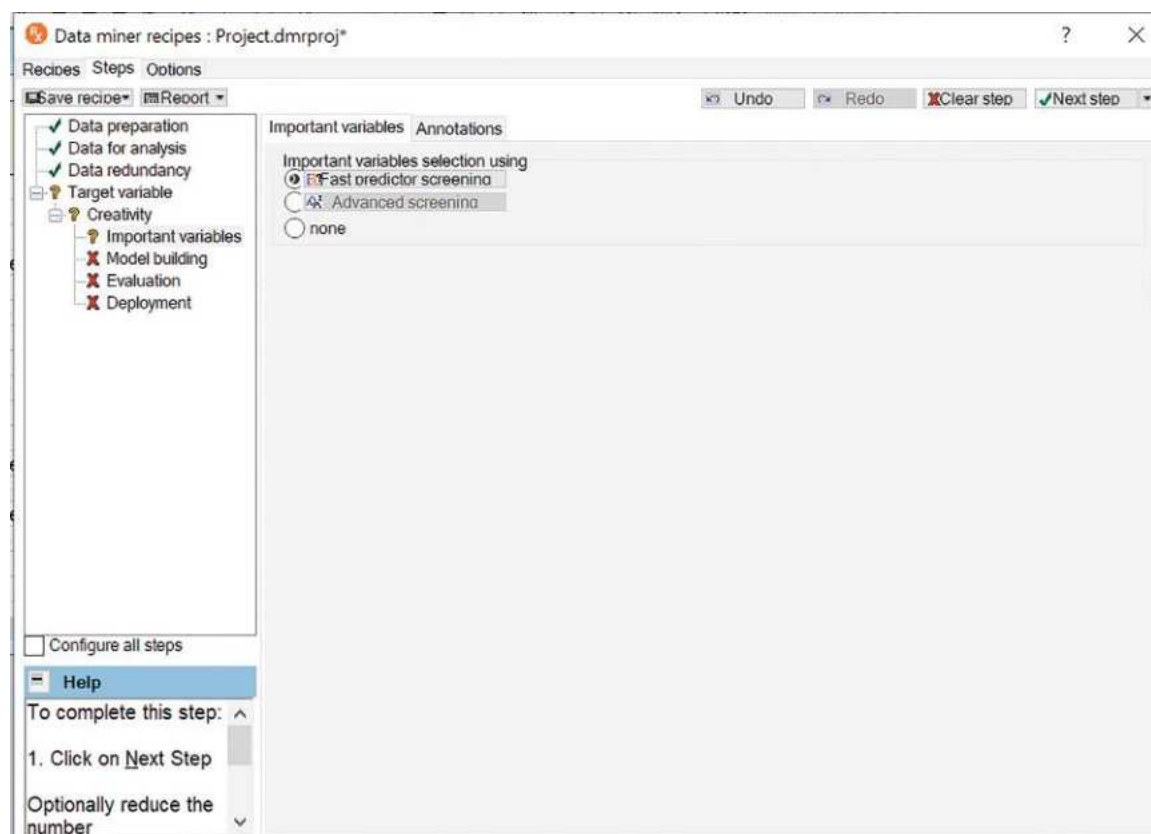


FIG. A.13 The variable selection screen, showing the method to be used.

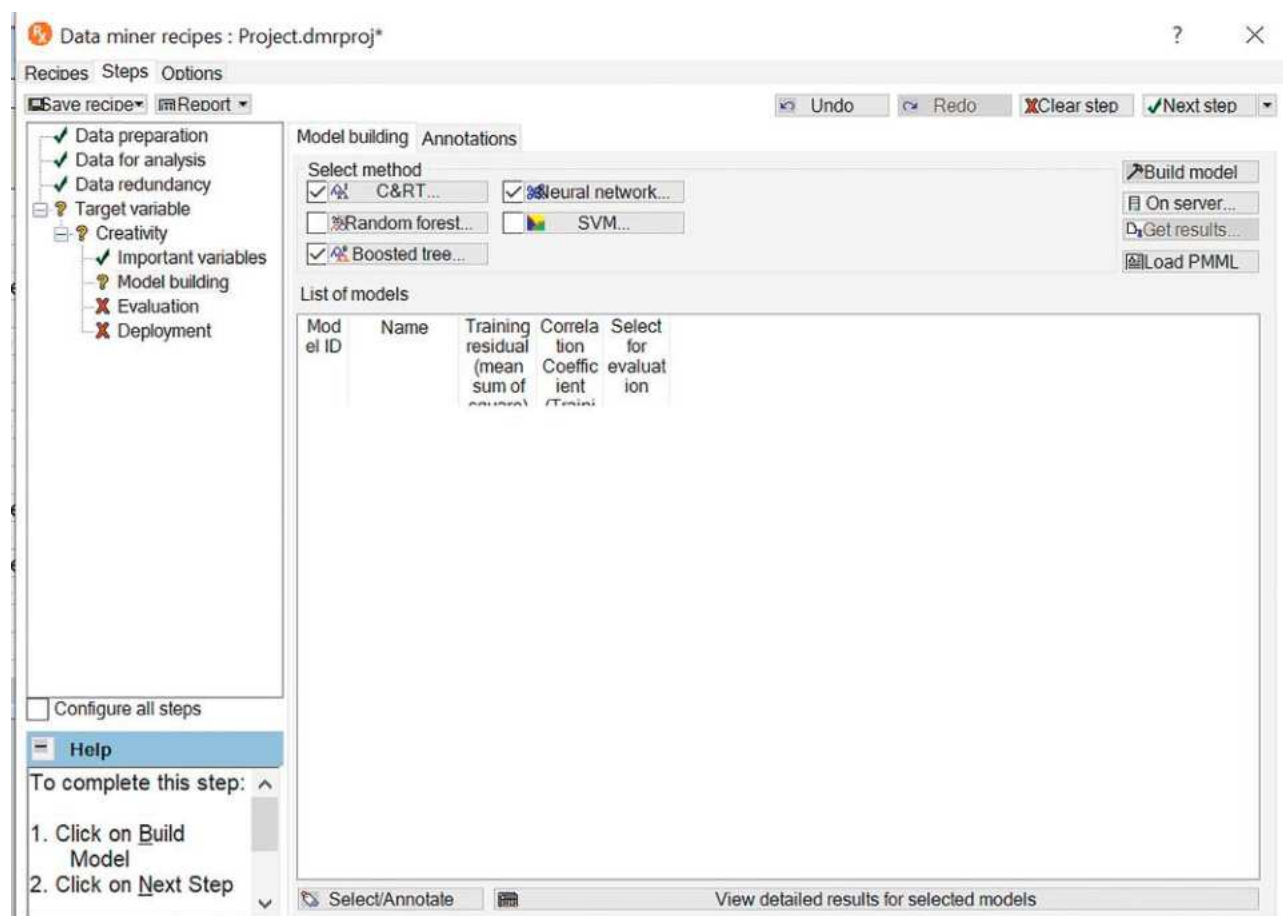


FIG. A.14 The model selection screen.

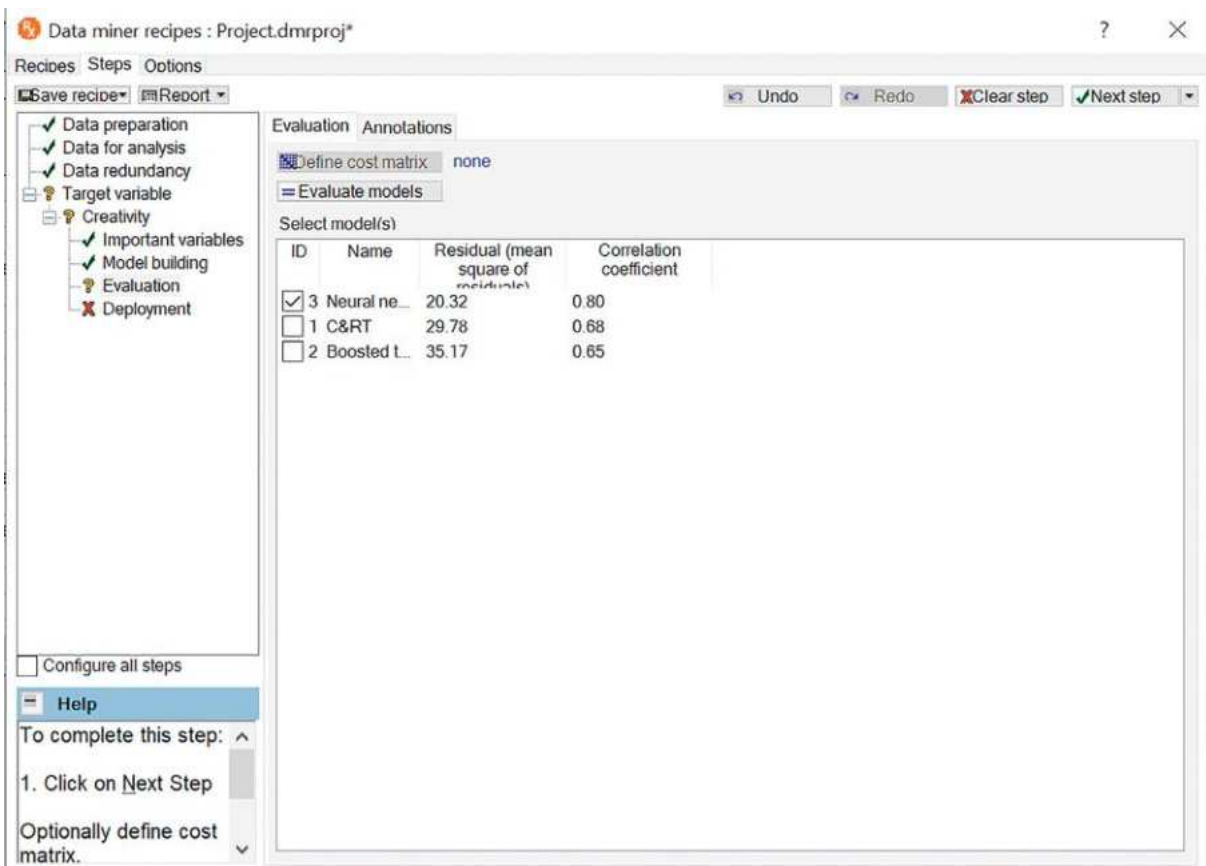


FIG. A.15 Intermediate results of the three algorithms.

B

Analysis of Hurricane data (Hurrrdata.sta) using the STATISTICA Data Miner 13

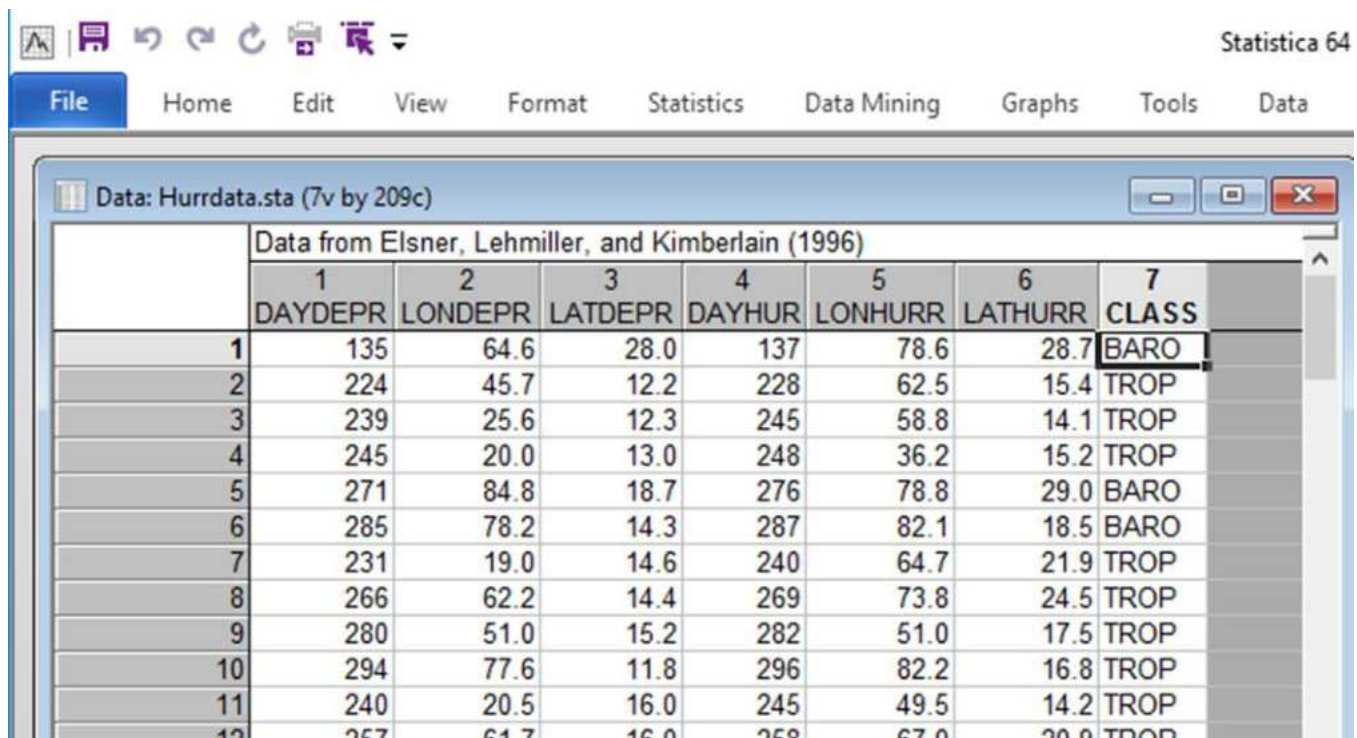
Jeff Wong

In this tutorial, we will demonstrate how to do some preliminary data mining to classify hurricanes using the “Data Miner Workspace” format in STATISTICA version 13.0 software.

Launch STATISTICA and close any data sets and/or workspaces that might automatically reopen from your previous STATISTICA session. From the File tab, click on the Open option and browse to where you saved the tutorial data. Select the Hurrrdata.sta data file to see the spreadsheet shown in Fig. B.1.

Notice that the CLASS variable includes two types (classes) of hurricanes to be predicted: BARO and TROP. Steps in building the modeling workspace:

1. Click on the File tab on the top menu and select the New option.
 - a. Accept the default operation to create a new workspace by clicking OK.
 - b. Select the data source by clicking on the single spreadsheet in the list, as shown in Fig. B.2.
2. Using visualizations to explore the data set:
 - a. Click on the Graphs tab to display the graphs menu shown in Fig. B.3.
 - b. Select two Scatterplots, a Histogram, and a Line Plot.
 - c. Connect these Graphs nodes to the Hurrrdata node, as shown in Fig. B.4.



	1	2	3	4	5	6	7
	DAYDEPR	LONDEPR	LATDEPR	DAYHUR	LONHURR	LATHURR	CLASS
1	135	64.6	28.0	137	78.6	28.7	BARO
2	224	45.7	12.2	228	62.5	15.4	TROP
3	239	25.6	12.3	245	58.8	14.1	TROP
4	245	20.0	13.0	248	36.2	15.2	TROP
5	271	84.8	18.7	276	78.8	29.0	BARO
6	285	78.2	14.3	287	82.1	18.5	BARO
7	231	19.0	14.6	240	64.7	21.9	TROP
8	266	62.2	14.4	269	73.8	24.5	TROP
9	280	51.0	15.2	282	51.0	17.5	TROP
10	294	77.6	11.8	296	82.2	16.8	TROP
11	240	20.5	16.0	245	49.5	14.2	TROP
12	257	64.7	16.0	258	67.0	20.0	TROP

FIG. B.1 The Hurrrdata data set to be used to predict the CLASS variable for a hurricane.

- d. Right-click on one of the Scatterplots nodes and choose the Edit Parameters menu option (Fig. B.5). Click the Variables button and select LONDEPR as the X variable, then select LATDEPR, DAYHUR, LONHURR, and LATHURR as Y variables.
 - Follow the same steps for the second Scatterplots node and choose LATDEPR as the X variable, then select DAYDEPR, LONDEPR, DAYHUR, LONHURR, and LATHURR as Y variables.
 - e. Right-click on the Line Plots and choose the Edit Parameters menu option. From the next dialog box, select the LONDEPR, LATDEPR, LONHURR, and LATHURR variables.
 - f. Finally, right-click on the Histograms node and choose Edit Parameters. Select all the variables, as shown in Fig. B.7.
3. In the Workspace window, click on the Run All menu option. A Reporting Documents node will

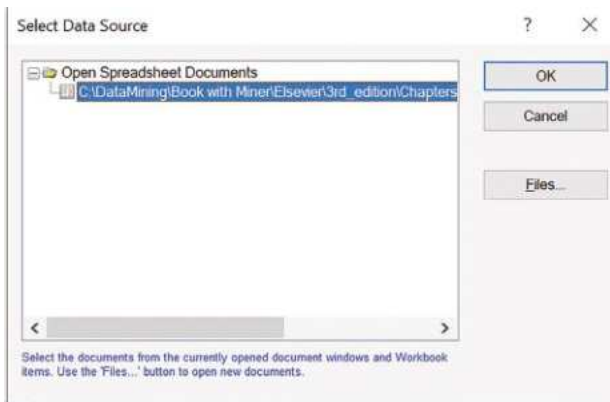


FIG. B.2 Selecting the data source to use for the model.

appear, which is connected to all of the Graphs nodes, as shown in Fig. B.8.

- a. Double-clicking the Reporting Documents node will open a window where you can browse all the plots that have been generated (Fig. B.9).
4. Explore the scatterplots, histograms, and line plots.
 - a. Click on the lower right scatterplot in the reporting document node to see a larger view, as shown in Fig. B.10.
 - b. In Fig. B.10, the LATHURR variable seems to have a positive correlation with LONDEPR.
5. Click on the Data Mining tab and select the Feature Selection option shown in Fig. B.11.
 - a. On the Feature Selection node menu, click on the Edit Parameters option.
 - b. In the dialog box that appears, click on the Variables button to get the variable selection window.
 - c. In this window, select CLASS as the categorical dependent variable and the remaining variables as continuous predictors, as shown in Fig. B.12.
6. Click OK and run the Feature Selection node.
 - a. The results of the Feature Selection node will be appended to the results listed in the Reporting Documents node.
 - b. Click on the Feature Selection results to see Fig. B.13.
7. This feature selection process permits selection of the most likely candidate predictor variables. Based on this chi-square value, we see that the LATHURR and LATDEPR are potentially the best predictor variables for CLASS. However, if we look at the Scatterplot of LATHURR versus LATDEPR (Fig. B.14), we see that these two variables are highly correlated. Using both variables may be a case of overfitting the data.

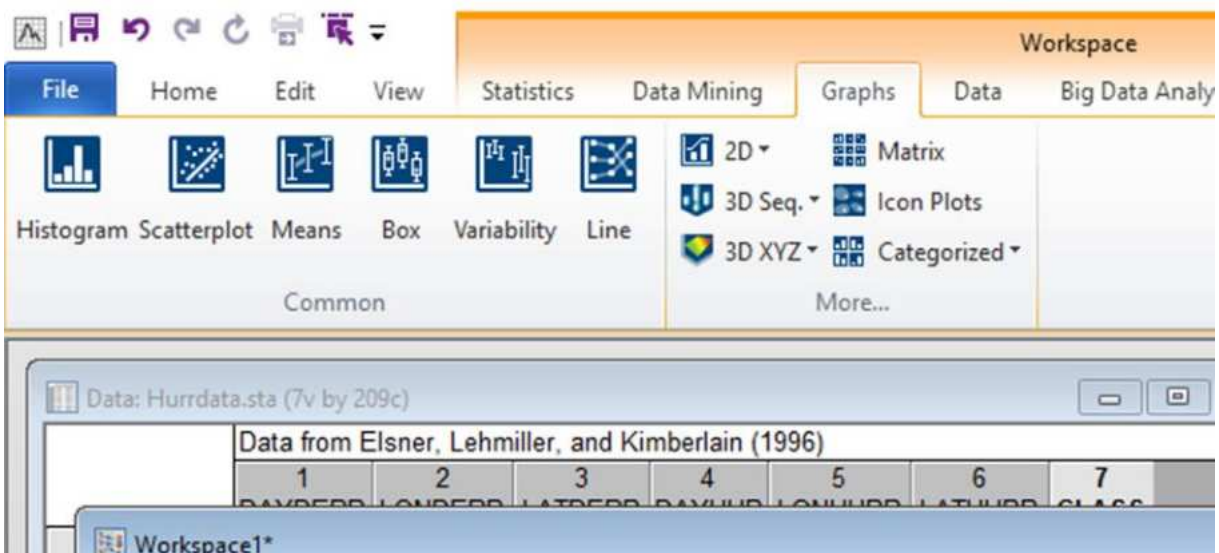


FIG. B.3 The graph selection menu.

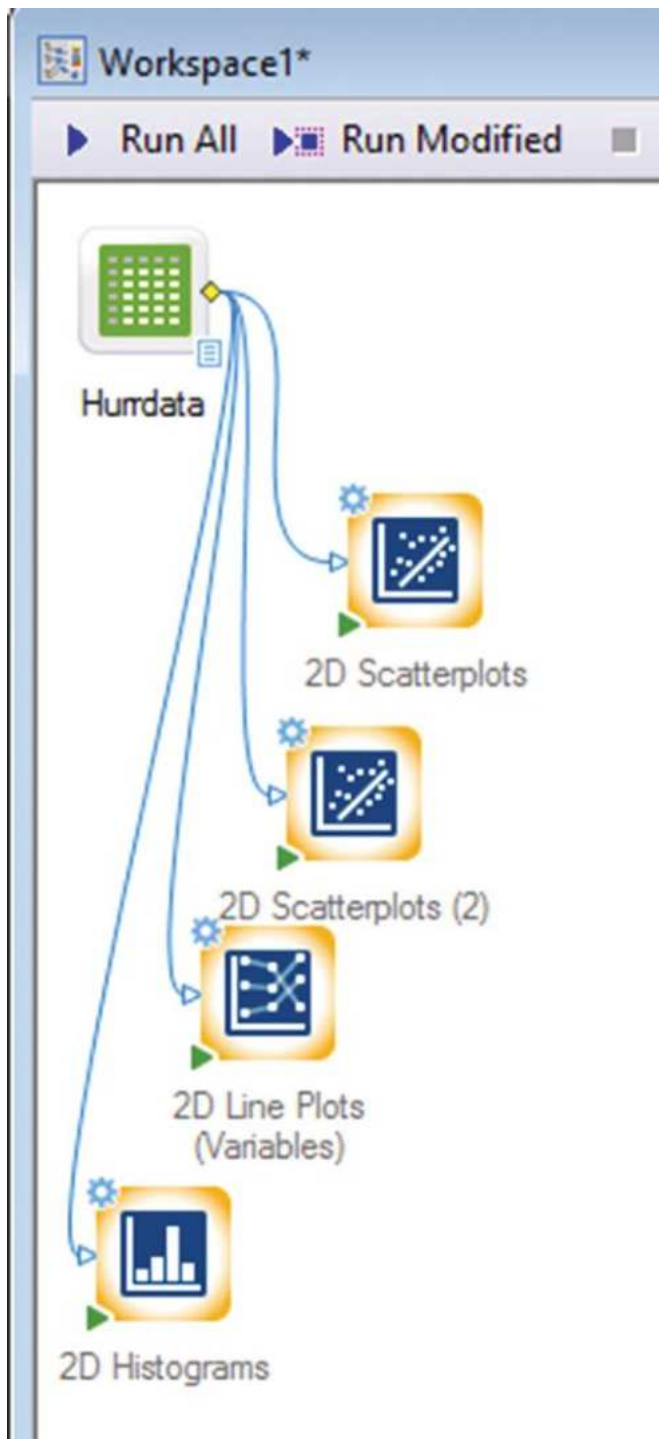


FIG. B.4 Four graphics connected to the Hurdata data source.

- a. Interestingly, the LONDEPR and LONHURR also show a strong linear relationship, but the chi-squared value of LONHURR is much weaker than that of LONDEPR.
- b. For the predictive analysis, we choose to use LATHURR and LONDEPR as the predictor variables, even though they may not have the same

units. For the time being, we will not include DAYHURR or DAYDEPR.

8. Click on the Feature Finder text box in the upper right corner of the STATISTICA menu bar.
 - a. Start to type: "select." It should provide some choices in the drop-down menu.
 - b. Choose the Select Variables option.
 - c. Then reuse the Feature Finder text box and type: "split."
 - d. Then choose the Select the Split Input Data into Training and Testing Samples (Classification). There should be two new nodes in your Workspace.
 - e. Connect the Hurdata node to the Select Variables node.
 - f. Connect the Select Variables node to the Split Input Data node.
 - g. Right-click on the Select Variables node and choose Edit Parameters.
 - h. Then click on Variables in the pop-up window that appears. Set CLASS to be the dependent categorical variable and choose LONDEPRE and LATHURR as continuous predictor variables, as shown in Fig. B.15.
9. When you run these two nodes in sequence, you should see two new nodes, labeled as Training Data and Testing Data.
10. Now use the Feature Finder again and type: "Best."
 - a. From the drop-down menu, choose: Best-Subset and Stepwise GDA ANCOVA (General Discriminant Analysis/Analysis of Covariance design) with Deployment (SVB).
 - b. Connect both the Training Data and the Testing Data to this new node that appears in the Workspace.
 - c. Right-click the Best-Subset and Stepwise GDA ANCOVA node and choose Edit Parameters.
 - d. In the pop-up window, modify the settings in the General tab to appear in Fig. B.16.
 - e. Run the node.
 - f. The results will appear in the Reporting Documents node, as shown in Fig. B.17.

In this case, we see that the algorithm correctly classified the hurricanes 86.8% of the time.

11. In the Node Browser, click on the Basic Statistics option and select the Crosstabulation Tables node (Fig. B.18). Double-click on the Crosstabulation Tables option to place it onto the Workspace.
 - a. Connect it to the Testing_PMML_GDA15 node that appeared on the Workspace after the Best-Subset and Stepwise GDA ANCOVA node was run. Leave the Training_PMML_GDA15 unconnected.

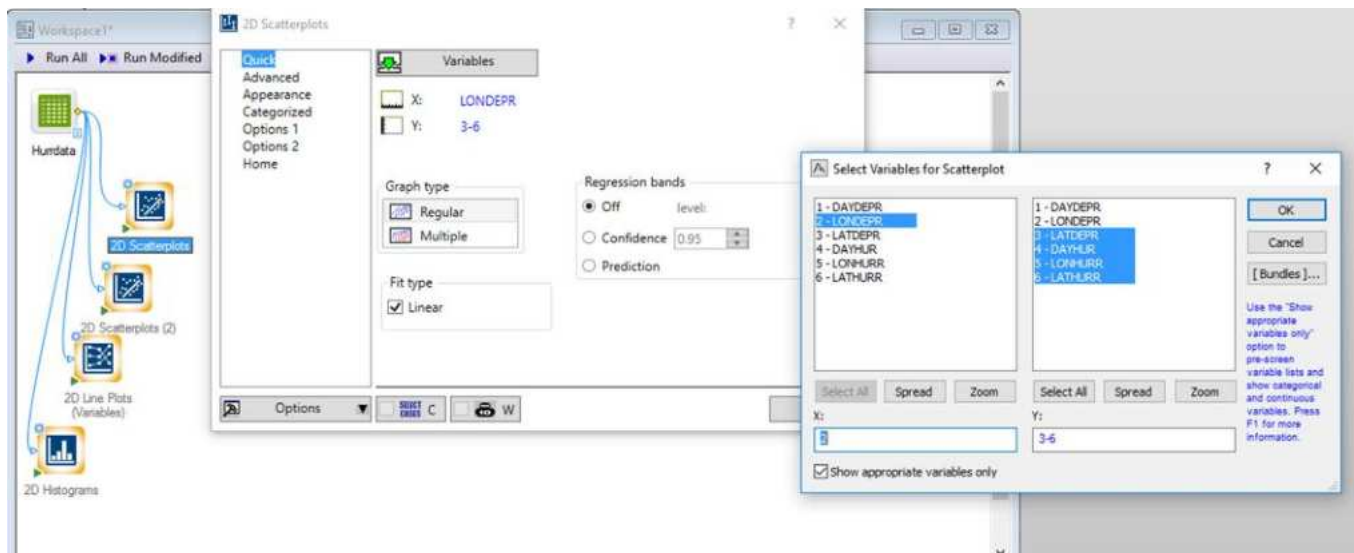


FIG. B.5 The Edit parameters menu for the Scatterplot graph.

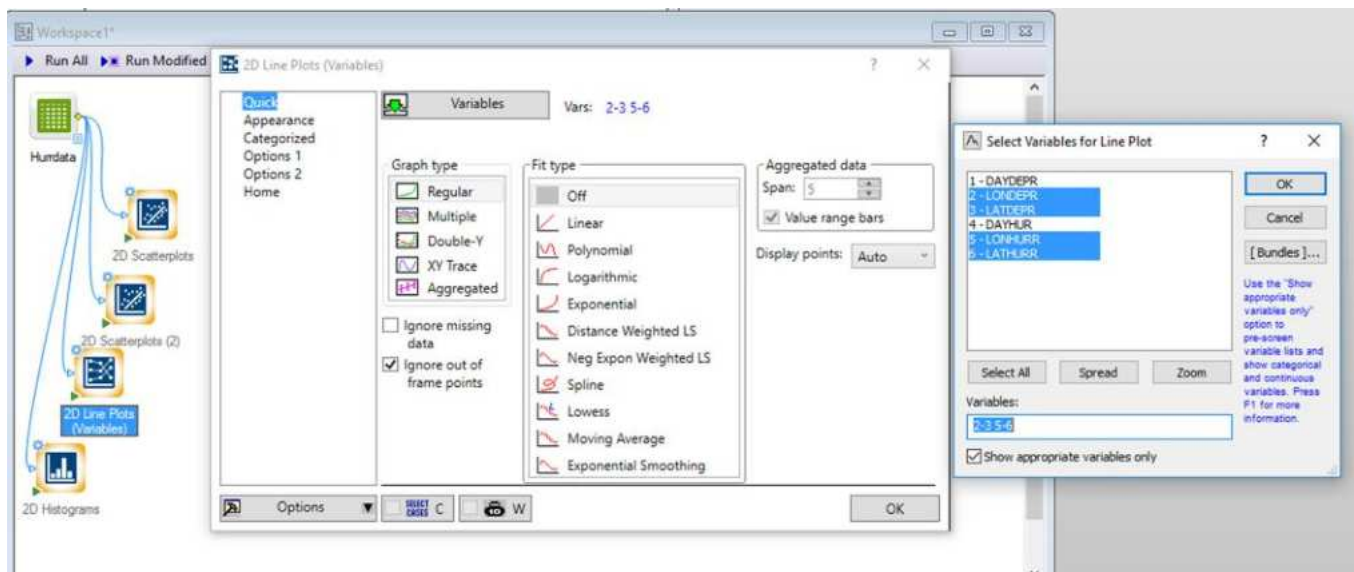


FIG. B.6 The Edit Parameters menu.

12. Right-click on the Crosstabulation Tables node in the workspace and choose Edit Parameters.
 - a. Select GeneralDiscriminantAnalysisPred in List 1 and CLASS in List 2 (Fig. B.19).
13. Run this node and take a look at the results in the Reporting Documents node, shown in Fig. B.20.
 - a. We see that the General Discriminat Analysis algorithm was reasonably successful in classifying BARO hurricanes from the Testing Data subset, with 91% accuracy ($49/103 \times 100\%$). However, it struggled a little bit in classifying TROP hurricanes, with a 65% rating ($32/49 \times 100\%$).
 - b. We could try to improve the accuracy by including some of the variables as predictor variables such as DAYHUR.
 - c. The other path is to consider other classification algorithms. The best performing algorithm will depend on the nature of the data being analyzed. The only way to determine the best one is to try as many algorithms as time allows.

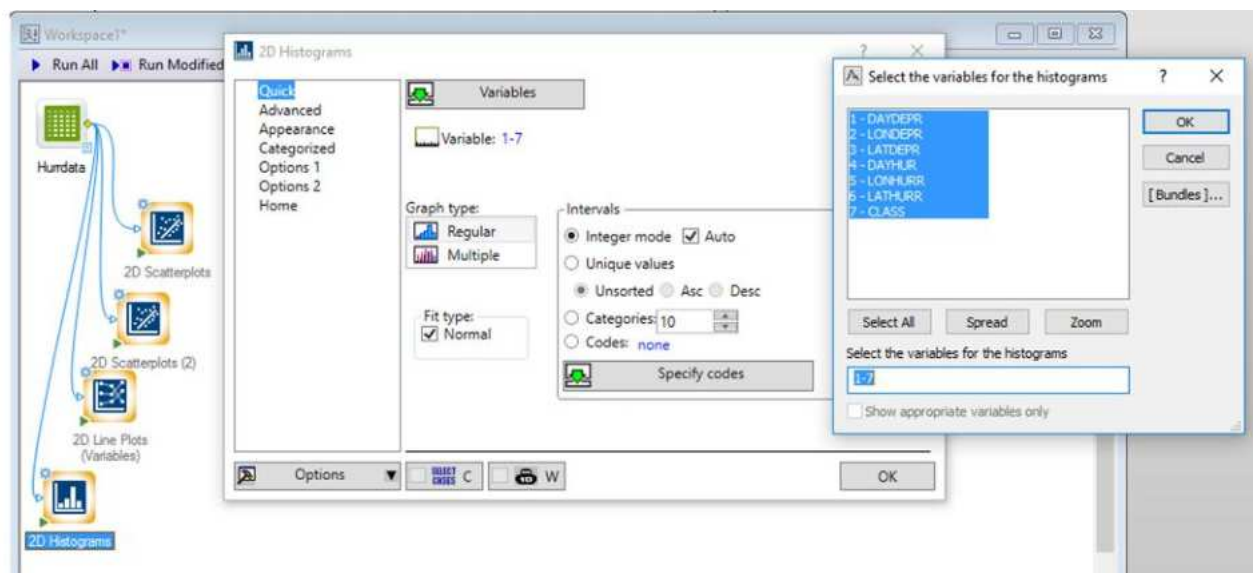


FIG. B.7 The Edit Parameters menu for the Histogram graphic.

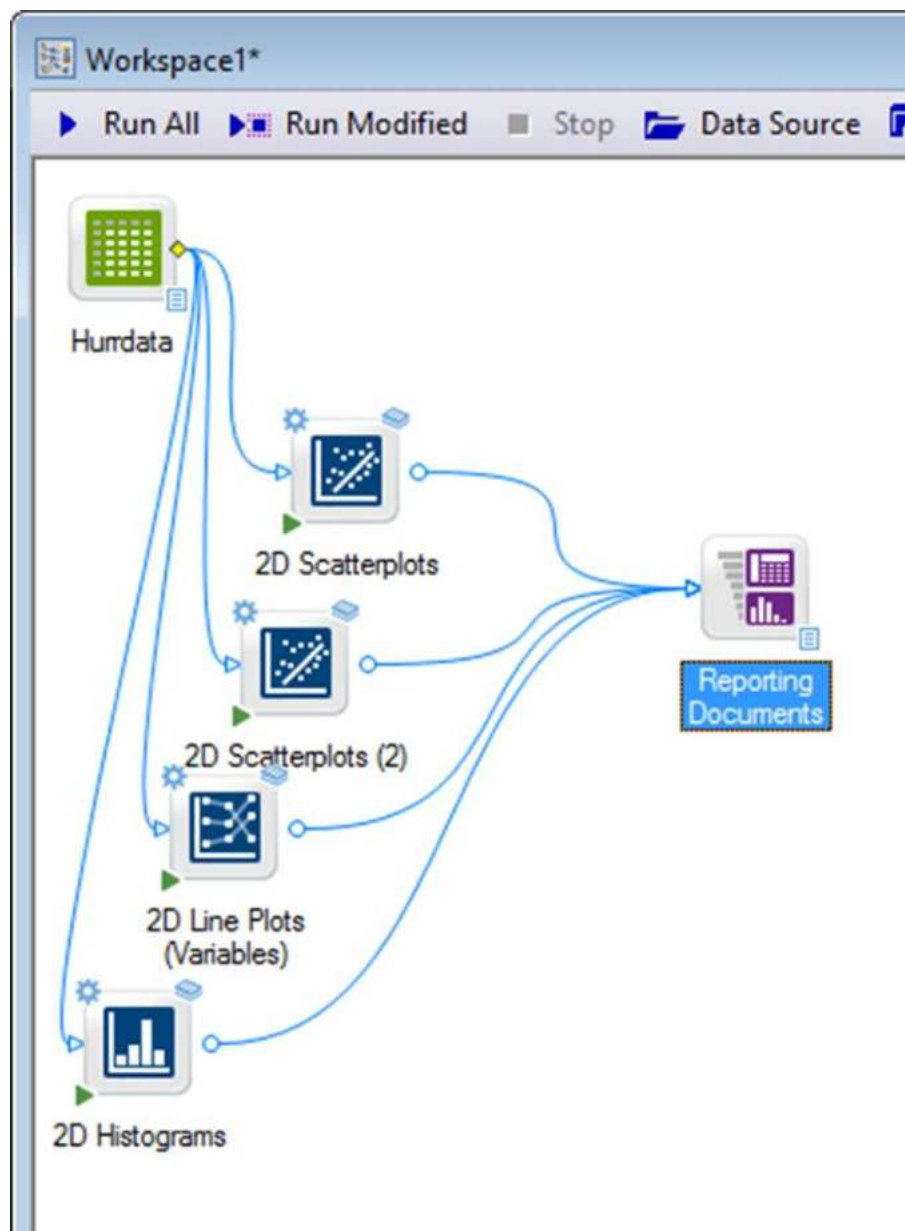


FIG. B.8 Results of the Run All option to produce the Reporting Documents node.

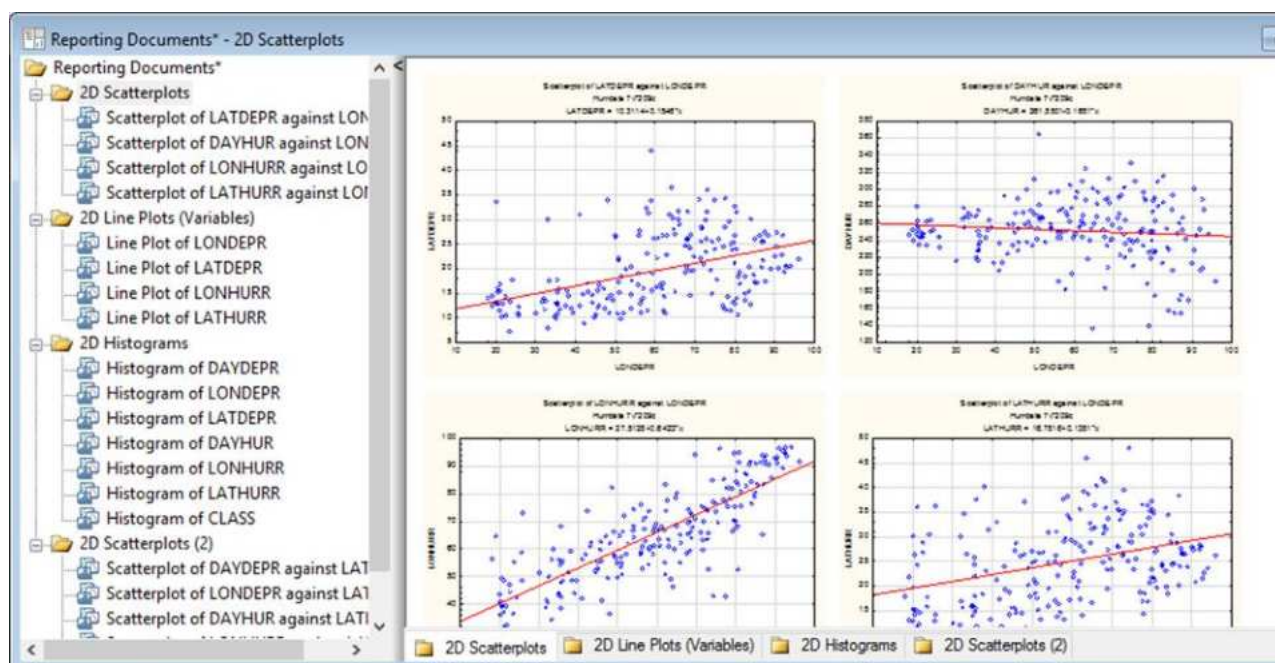


FIG. B.9 Scatterplots, line plots, and histogram plots in the reporting document.

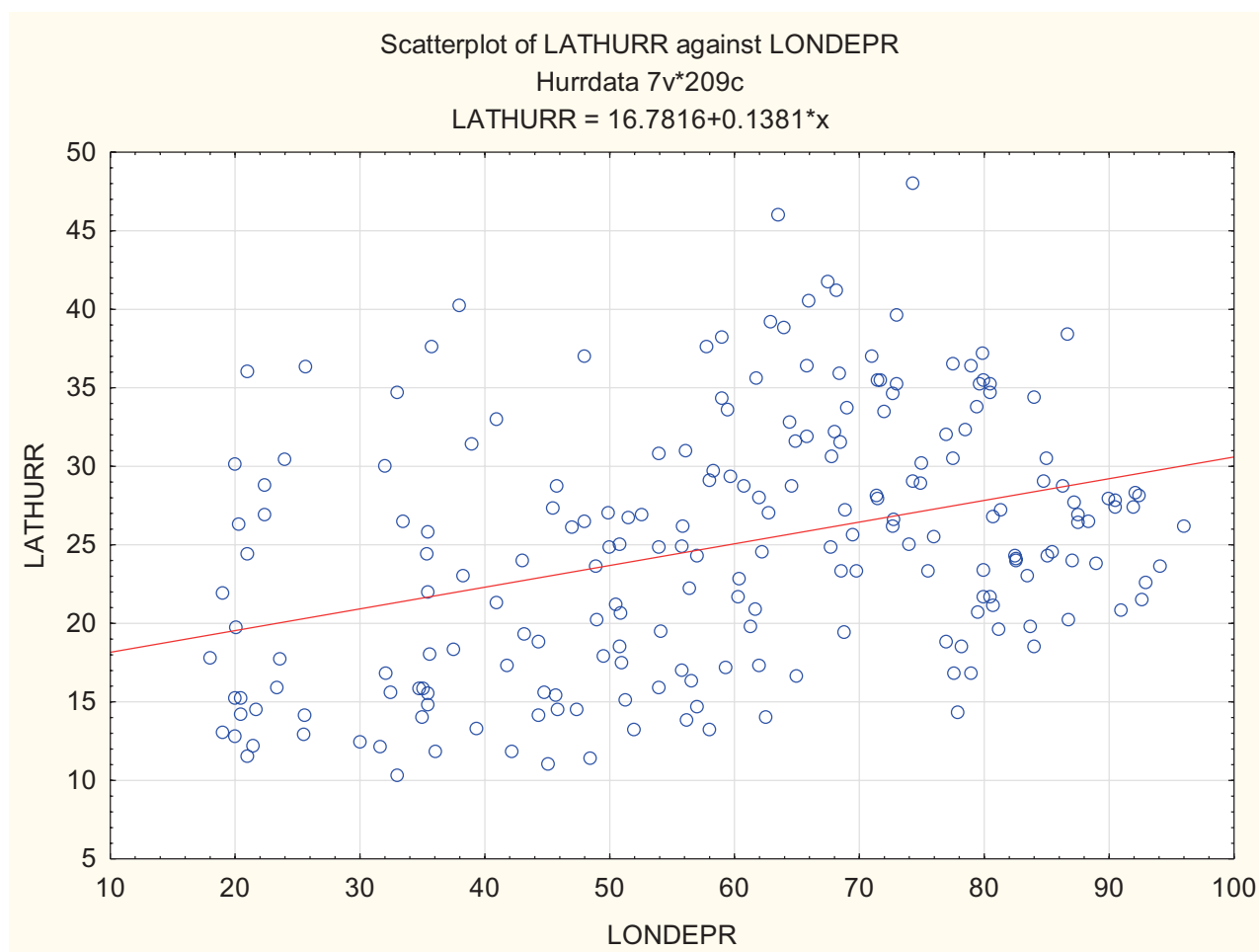


FIG. B.10 Scatterplot of LATHURR and LONDEPR variables.



FIG. B.11 The options of the Data Mining tab on the main menu.

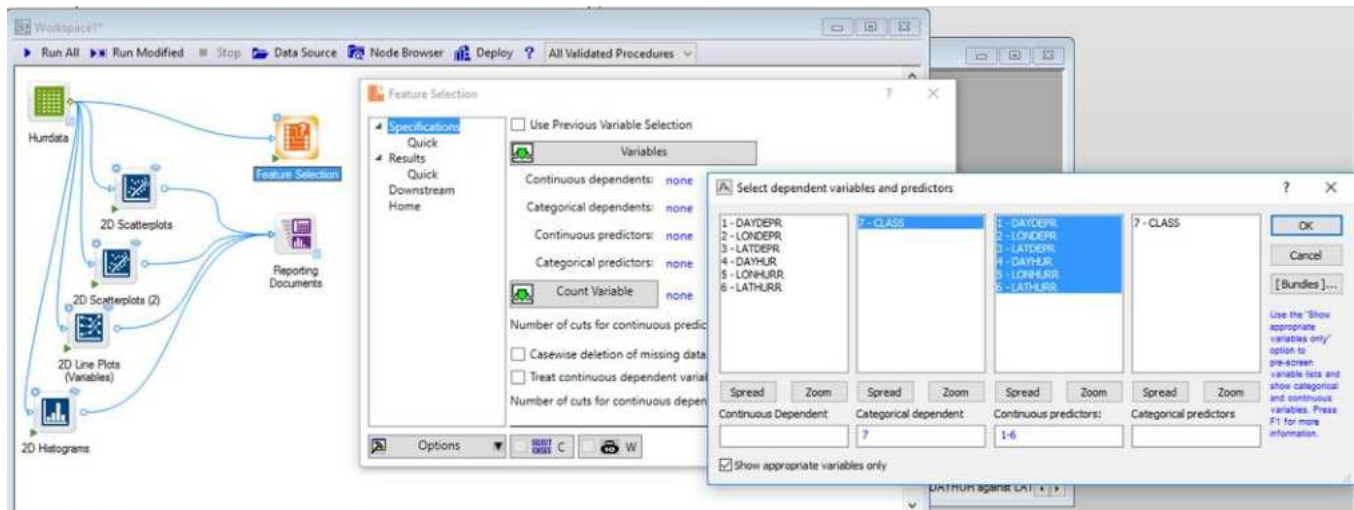


FIG. B.12 Selection of all predictor variables for uses in Feature Selection processing.

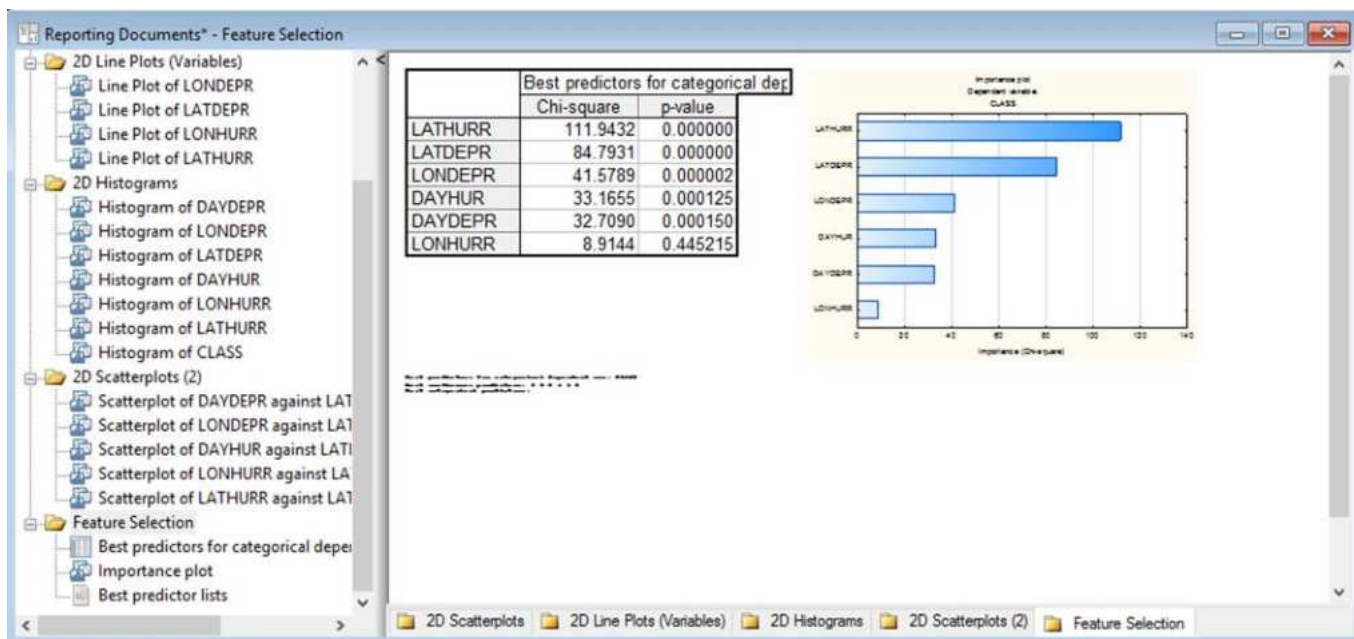


FIG. B.13 The list and importance plot of the best candidate predictors for CLASS in the Hurdata data set, based on chi-square values and p-values of each variable.

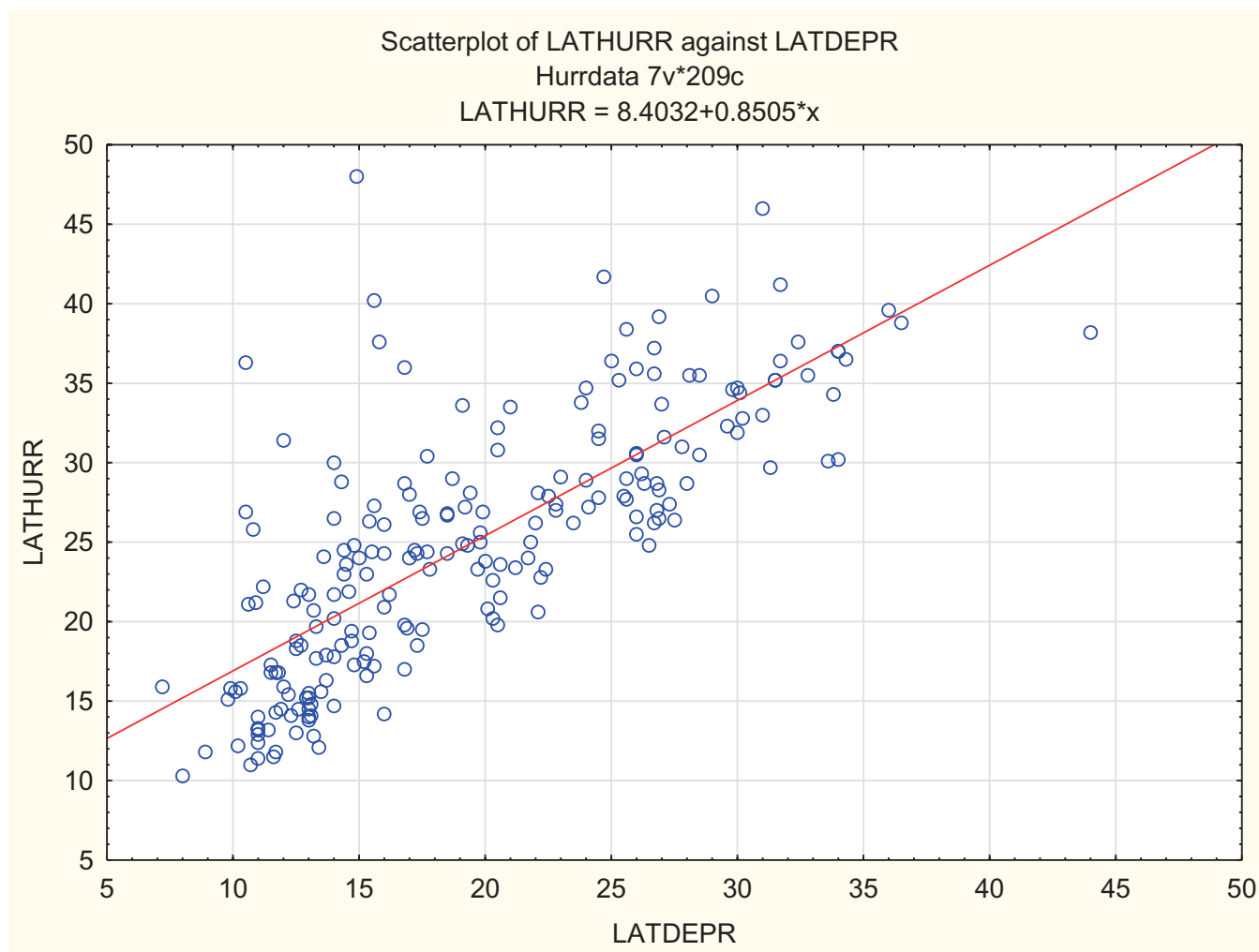


FIG. B.14 Scatterplot of LATHURR against LATDEPR.

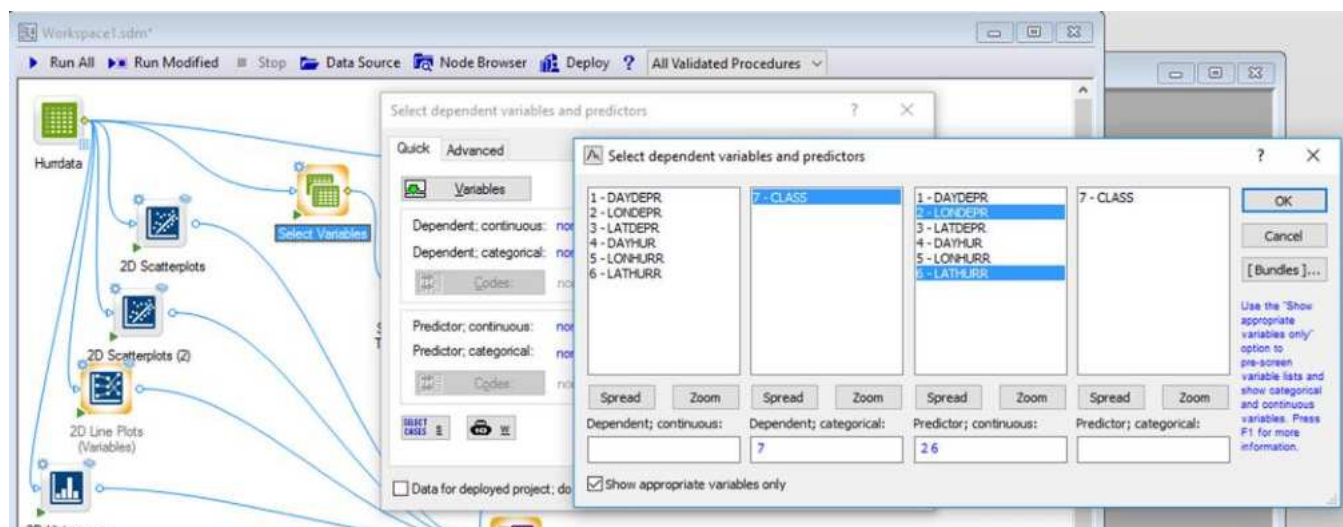


FIG. B.15 Selection of important predictors of CLASS.

Edit Parameters

Parameters for Best-Subset Selection Deployment Deployment Script

General **Parameters for Stepwise Selection**

Model building method: All effects

Detail of computed results reported: All results

Construct factorial to degree: 1

Priors: Equal

Case statistics: None

Sweep delta 1.E-: 7

Inverse delta 1.E-: 12

Generate datasource, if N for input less than: 1000

Set the prior classification probabilities, for classifying observations. The default specification is Estimated; use this option to set the prior classification probabilities proportional to the observed group (class) N's; use the Equal option to assign equal probabilities to each group or class specified in the categorical dependent variable.

Add New Parameter... Save (Create a New Node) As... OK Cancel

FIG. B.16 Edit Parameter screen selections.

Reporting Documents* - Classification Matrix (Split Input Data into Training and Testing Samples (Classification))

Classification Matrix (Split Input Data into Training and Testing Samples (Classification))
Classifications: Rows(Observed) Columns(Predicted)
(Analysis sample)

Class	Percent Correct	BARO p= .5000	TROP p= .5000
BARO	84.78261	39.00000	7.00000
TROP	88.33333	7.00000	53.00000
Total	86.79245	46.00000	60.00000

Tests of Significance of Squared Mahalanobis Distances (Split... Classification Matrix (Split Input Data into Traini...

FIG. B.17 Reporting Document showing the modeling results.

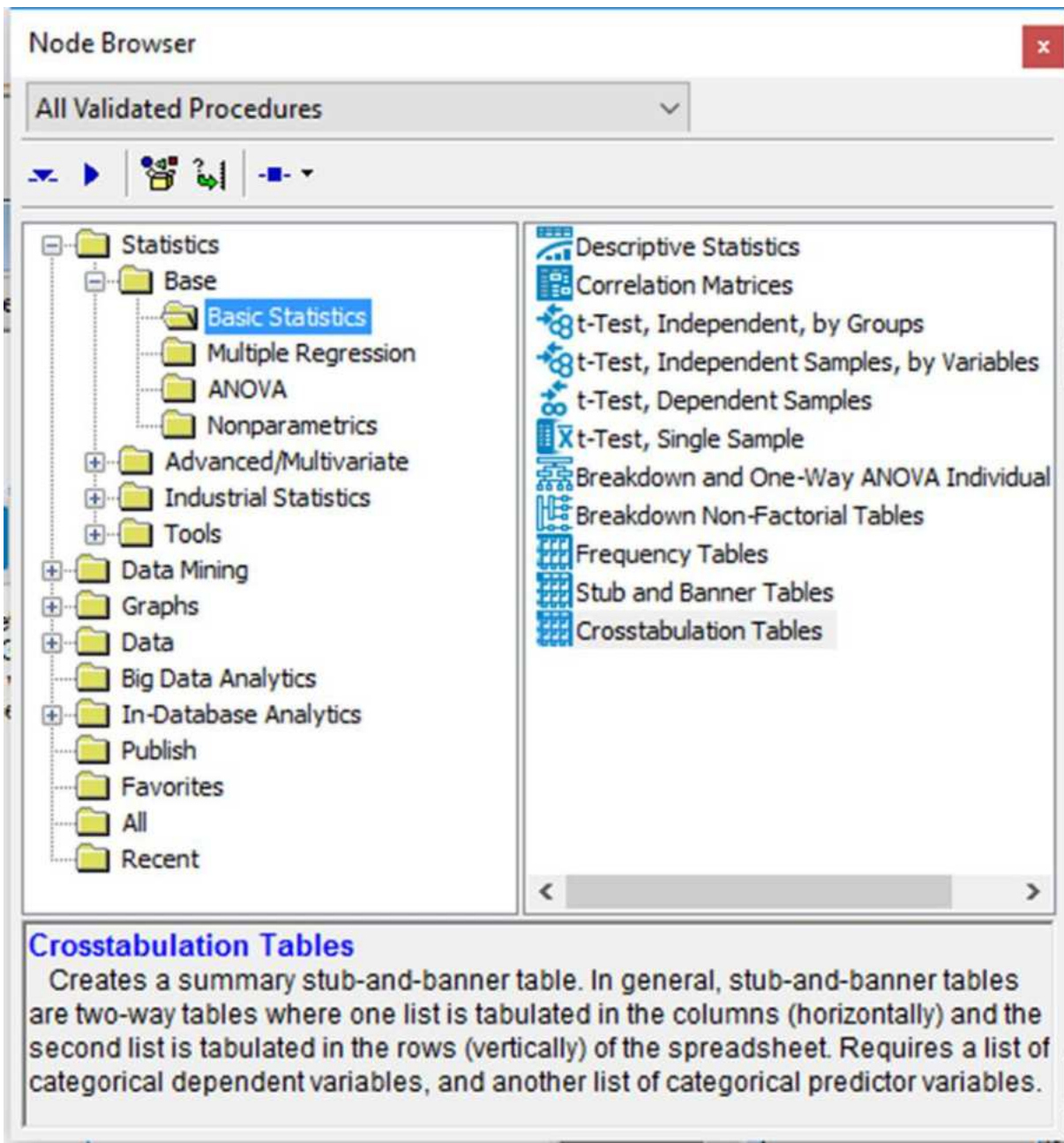


FIG. B.18 Selection of the Crosstabulation Tables options in the Node Browser.

14. The completed workspace is shown in Fig. B.21.
15. Next, we will try a machine learning algorithm.
 - a. From the Data Mining tab, access the Machine Learning drop-down menu and select the Support Vector Machines node.
 - b. Connect this node to the Training Data node.
 - c. Right-click on the Support Vector Machines node and select the Edit Parameters option. CLASS should already be set to be the categorical dependent, and LONDEPR (variable #2) and LATHURR (variable #6) are already set as the

- continuous independent variables in Fig. B.22.
- d. Run the Support Vector Machines node, and the results should appear in the Reporting Documents node (Fig. B.23).

Discussion

The Support Vector Machines algorithm is doing a better job of classifying TROP hurricanes, with an accuracy of 90% compared with the previous method's 65%

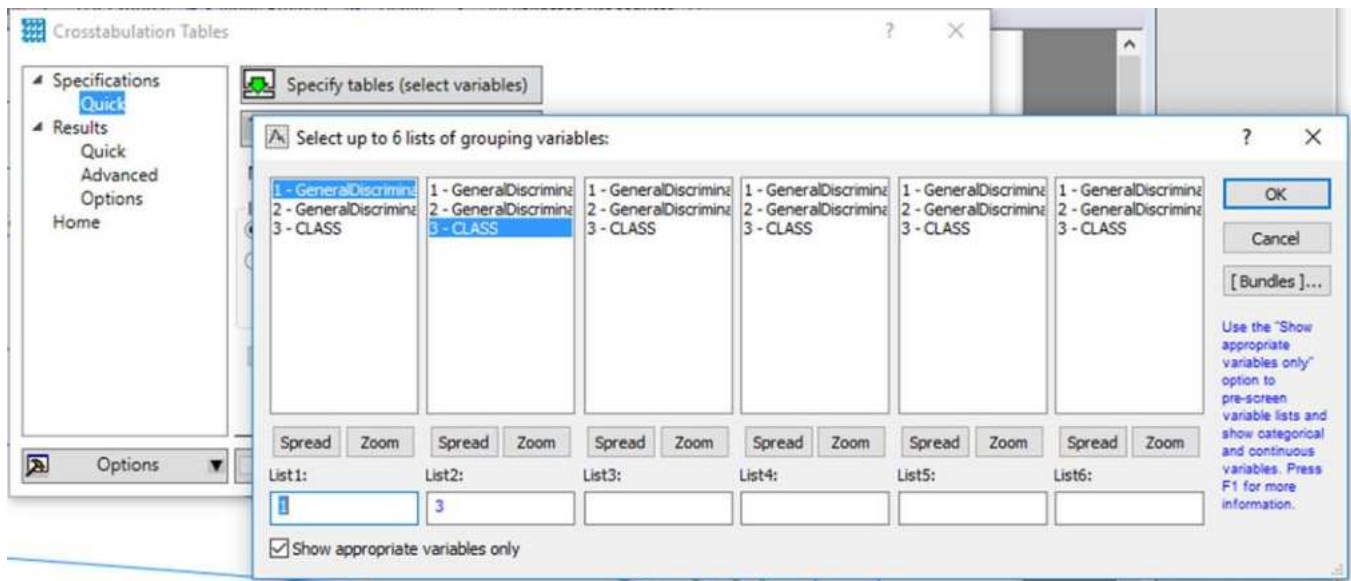


FIG. B.19 Edit Parameters screen settings for the Crosstabulation Tables node.

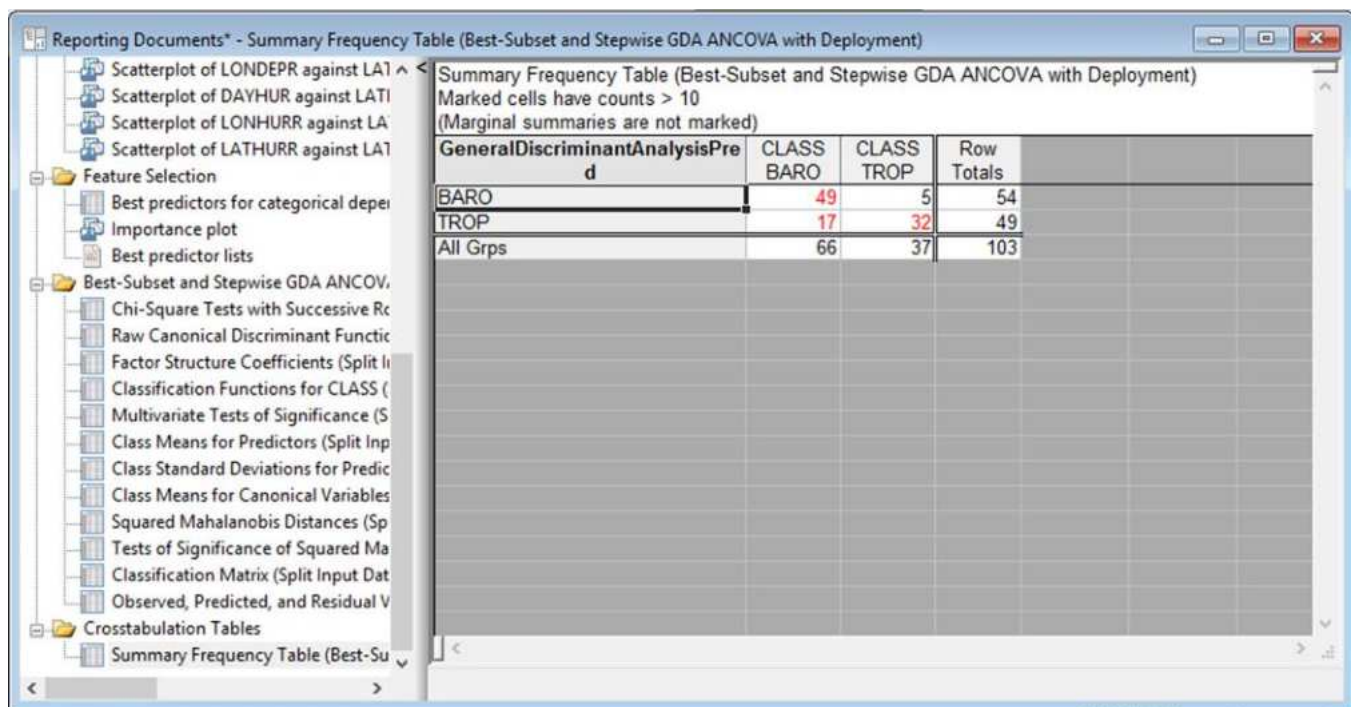


FIG. B.20 Results of the General Discriminant Analysis model.

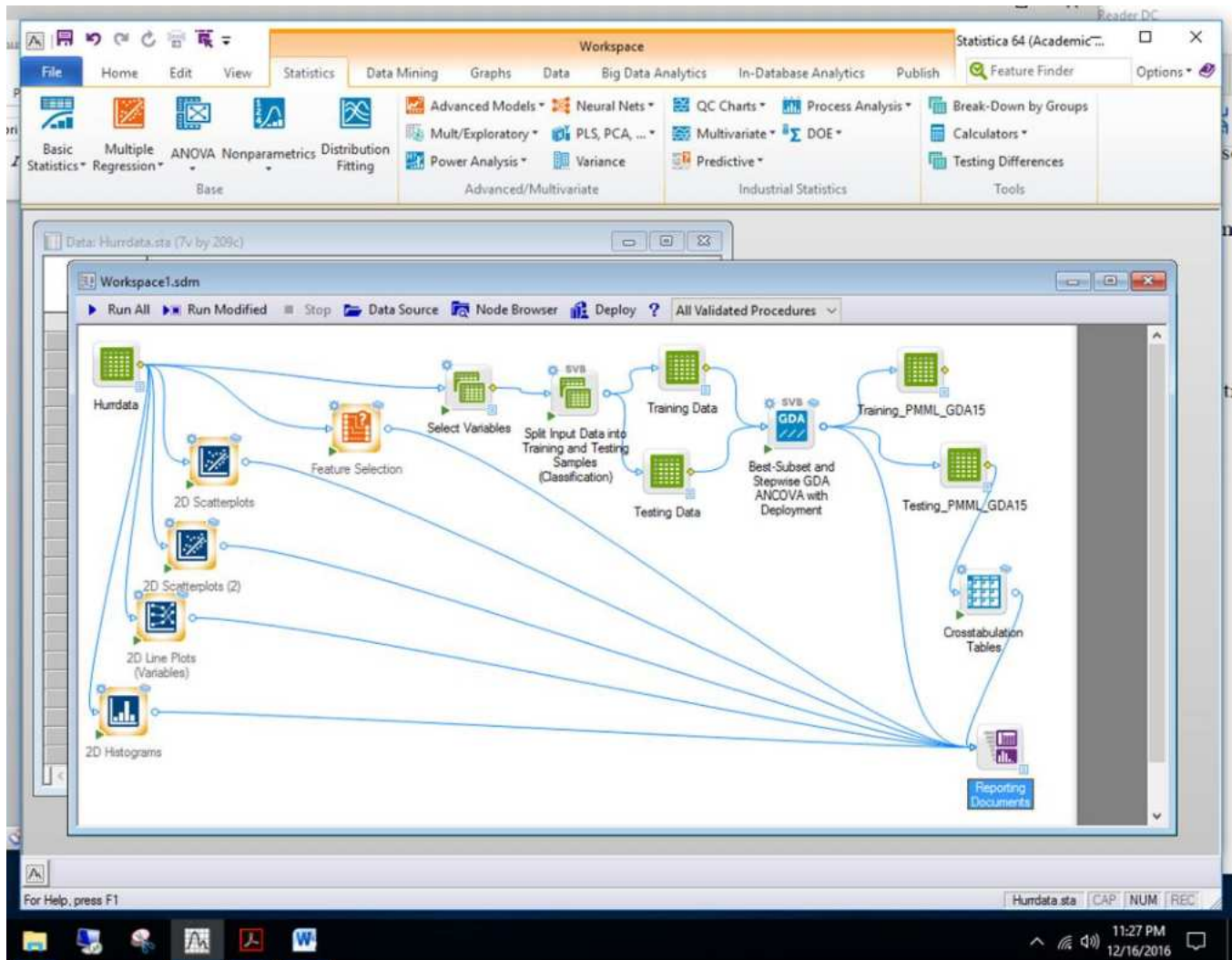


FIG. B.21 The completed workspace.

accuracy. However, SVM accuracy with BARO hurricanes is 82.6% compared to with Best-Subset and Stepwise GDA ANCOVA's 91%.

So there is a trade-off between these two methods with this data set. Which one should we choose for deployment? We can ask ourselves a few questions to help make this choice. For example, is it more important to correctly classify BARO hurricanes or TROP

hurricanes? If classifying TROP is more important, then the Support Vector Machines algorithm seems to be the better choice. However, if BARO classification is more important, then Best-Subset and Stepwise GDA ANCOVA may be the better algorithm.

Fig. B.24 shows the final configuration of the STATISTICA Workspace after our experimentation with the Hurdata.sta data set.

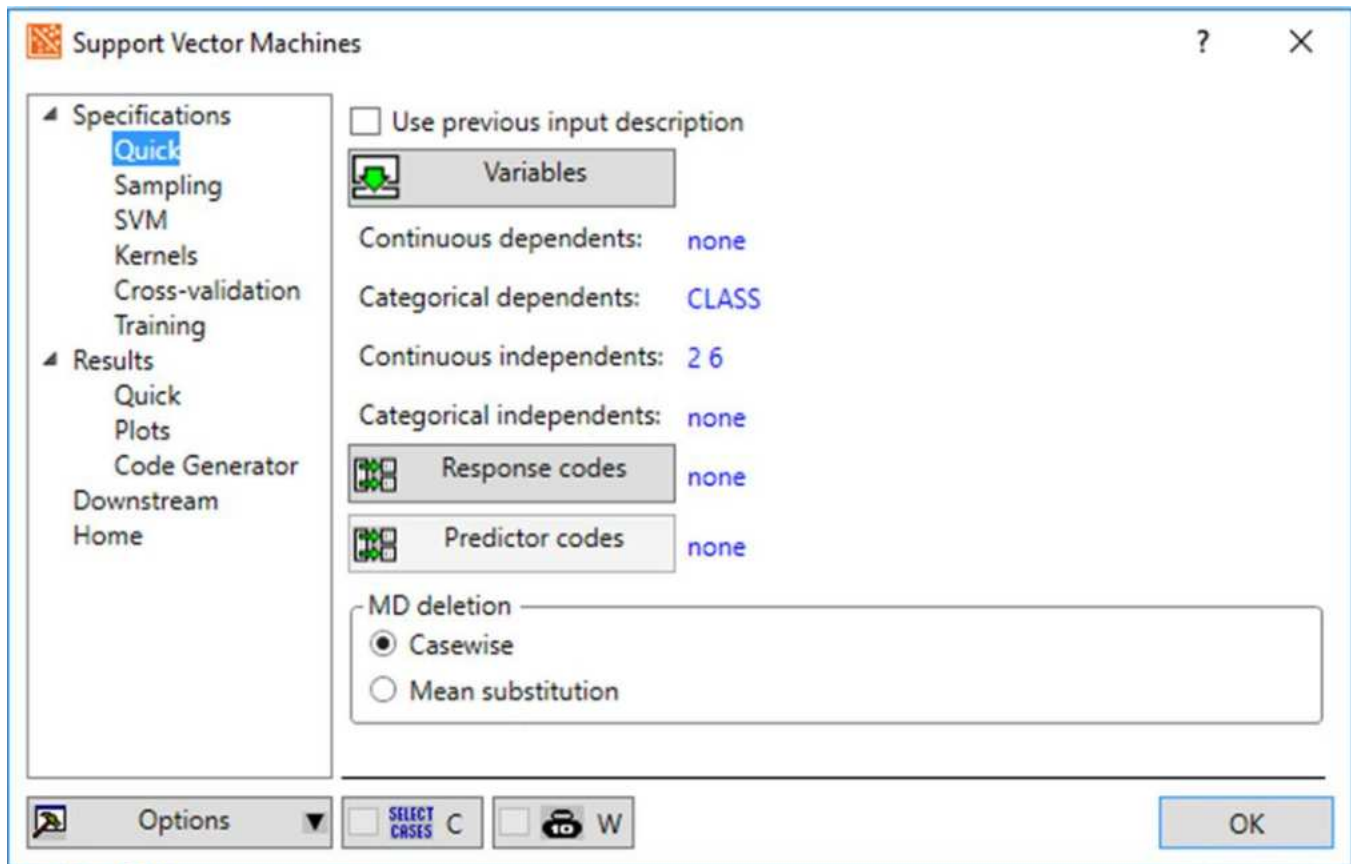


FIG. B.22 The Edit Parameters screen of the Support Vector Machines node.

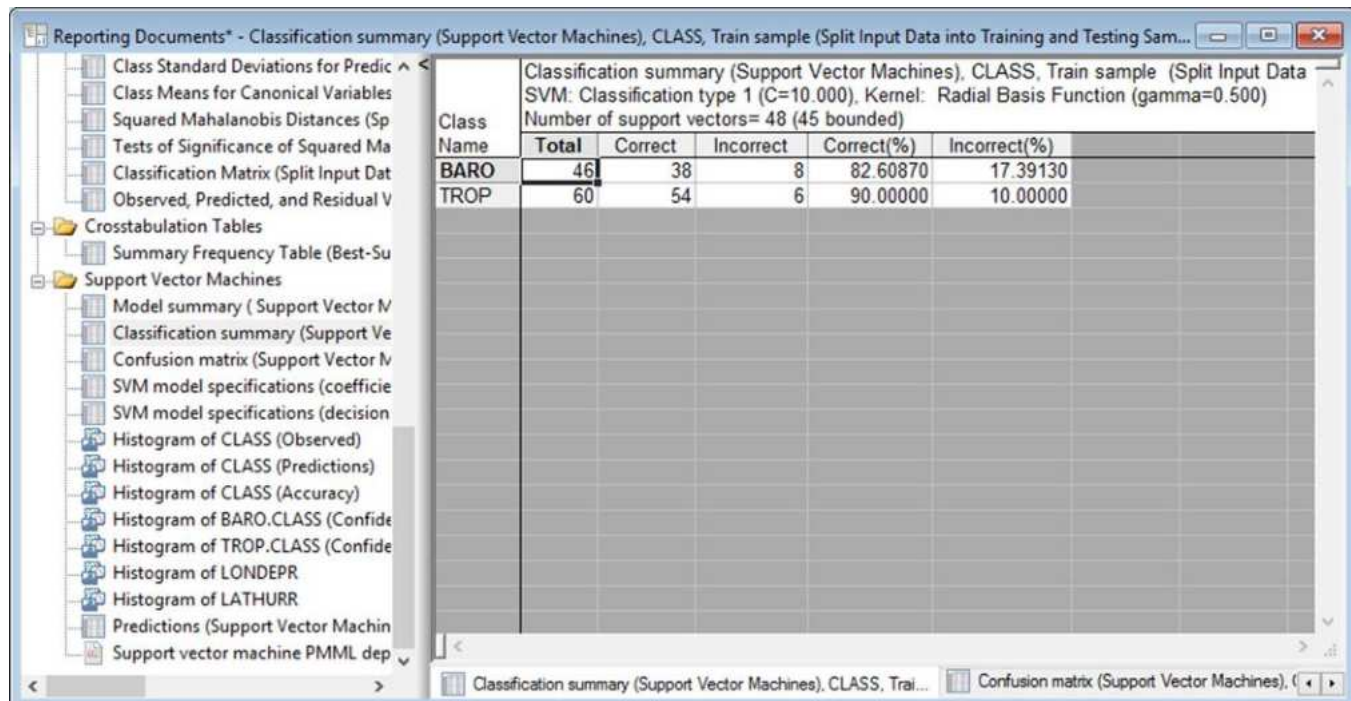


FIG. B.23 Prediction results of the Support Vector Machines algorithm.

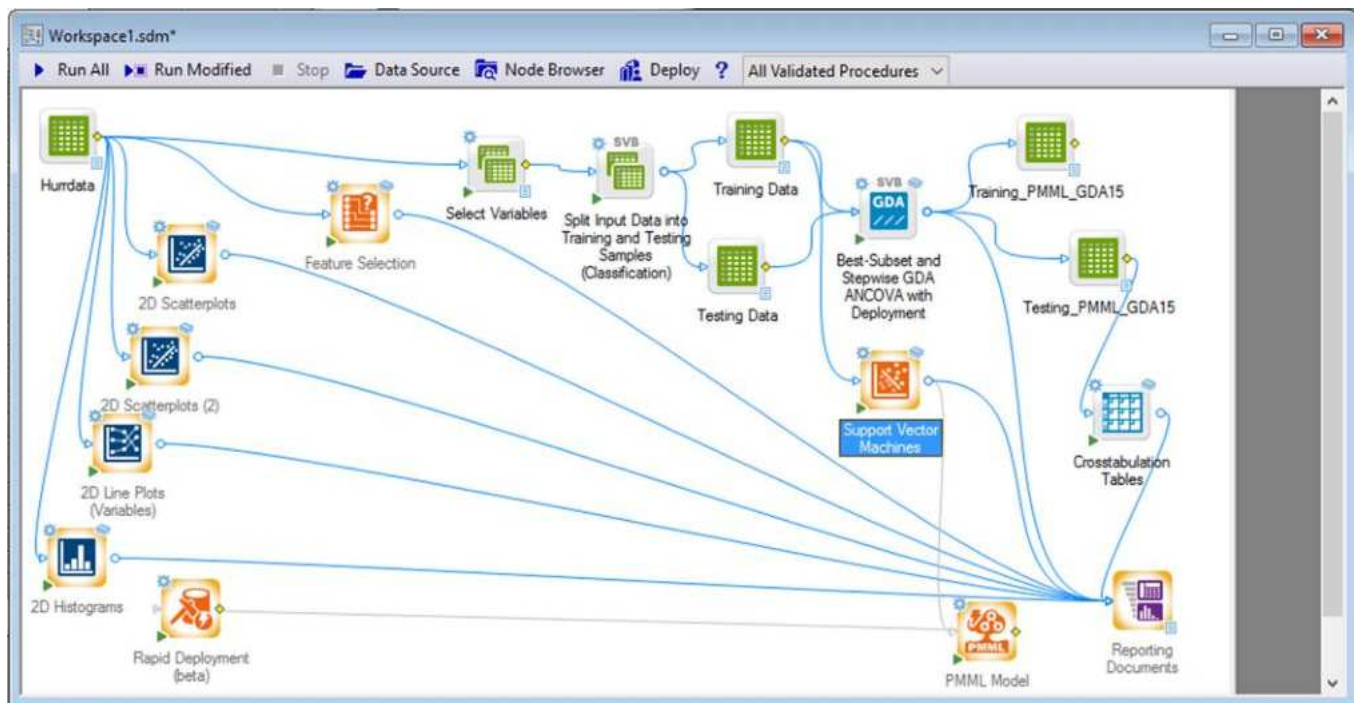


FIG. B.24 Final configuration of the STATISTICA workspace.

C

Case study – Predicting student success at high-stakes nursing examinations (NCLEX) using IBM SPSS Modeler and STATISTICA Data Miner 13

Galina Belokurova and Chiarina Piazza

NOTE: Because of confidentiality issues with the data used in this tutorial, the data cannot be presented “as used”; thus this is presented as a “case study” without an accompanying dataset. Readers can study this and find their own data of similar format or make up a dataset of similar format, if the reader wishes to work through the tutorial with data.

Introduction

Recent years saw a proliferation of proprietary software that makes building data mining applications much faster and easier. This case study deals with students in academic nursing programs who are required to pass a licensure examination (NCLEX) before they can become practicing nurses. It shows how one can create a procedure that sets apart students struggling to stay on track in their program and helps refer them to remediation professionals. In the future, we plan to use predictive modeling to choose automatically personally customized study plans for each of the aspiring nurses (or other licensed professionals), regardless of whether they require additional attention or not.

We compare two existing data mining packages – IBM SPSS Modeler and STATISTICA – to introduce predictive analytics to a wider audience in higher education and uncover the features these two packages exhibit. Both provide a number of options for running data mining routines automatically with varying degrees of control from the analyst. Both are successful at saving time spent on competitive model evaluation and can easily be used as driving engines in adaptive decision management systems.

This case study goes over five different data mining algorithms used to predict students’ success on the NCLEX. Models are chosen automatically: CHAID and C5.0 algorithms are selected by the IBM SPSS Modeler routine, while C&RT, boosted trees, and neural networks algorithms are selected by STATISTICA.

The Assessment Technology Institute (ATI) assessment and the virtual ATI (VATI) are two very commonly used assessment programs for evaluating prelicensure nurses. Overall, all models identify ATI and the VATI preparatory program completion as the two most important predictors, and this can likely be explained by the strength of these two variables. It is probably not a surprise that the best predictors for the successful passage of an exam are measures of performance at the mock exams that students take to prepare.

IBM SPSS Modeler and STATISTICA algorithms do slightly differ in terms of the exact number of ATI assessments failed that sets the likely passers apart. It varies between six and nine, and the outcome may also depend on such factors as students’ nursing GPA, their grades in a pharmacology course, or the number of didactic nursing classes failed. None of the demographic and socioeconomic variables seem to play a significant role.

The knowledge discovered by predictive algorithms can give educators much greater power and ability to customize students’ educational paths at almost any point in the nursing curriculum. For instance, the CHAID algorithm shows that failing three assessments (including repeats) already may serve as a strong signal to guide a student toward some form of early remediation. A quicker reaction can help minimize the number of

students who need to take extra coursework to be able to graduate. The earlier the remediation occurs, the less costly it is for all parties involved.

Decision management in nursing education

Universities make decisions at all levels – from strategic (opening a campus) to micro (referring a student for remediation or allowing a dropout to reenroll) – but it may be especially challenging to sustain a consistent approach to decision making when one needs to deal with high-volume, repeatable, and relatively low-stakes (for the organization) operational decisions that often require high levels of consistency over time.

Such difficulties are not unique to higher education: in fact, commercial industries have been dealing with the problem of sustaining consistency in everyday operations for quite some time. This experience has already crystallized into a new approach that requires transforming decision support systems supplying information to human decision-makers into decision management systems making the bulk of operational microdecisions on their own. The most important difference between the two is the push to take human decision-makers out of the process as much as possible to prevent delays and simplify the process.

Predictive analytics models are an integral part of this transition, because they serve as the “brain” behind any automated decision-making tool. The predictive models presented here should drive the student remediation referral process in a clear, understandable, and agile manner, helping target those at risk of falling behind. The main idea is to make a proactive decision of helping students, not reacting after it may be too late.

Case study

West Coast University (WCU) is a midsize professional school specializing in healthcare education. About 90% of its students are enrolled in prelicensure baccalaureate programs, including a bachelor of science in Nursing (BSN) and a bachelor of science in Nursing for vocational nurses (LVN to BSN track and LPN to BSN track). The total number of students it enrolls annually equals approximately 6500 – a midsize university in terms of total enrollment. Its nursing program, however, is large when compared with its peers in Southern California. WCU students comprise about 18% of the nursing student population in the state of California and 42% of the nursing student population in Orange, Riverside, and San Bernardino counties (based on 2013–14 California Board of Nursing data).

Research question

Who should be automatically referred to the university’s remediation program? At this point, WCU staff uses a decision support system that channels data from the institution’s data warehouse to its academic and business users to help them decide whether a particular student is a candidate for remediation. The final decision about referral, therefore, is made by people.

A predictive model can help isolate struggling students and automatically refer them to remediation coordinators, but for this to happen, the model behind it should be sufficiently accurate. Here we attempt building a series of models that can serve this goal and ultimately minimize the number of students in need of help who are missed by the university staff as well as the amount of human effort directed at sifting through and evaluating student files manually.

Literature review

Predicting student academic success and attrition has long been a popular topic in higher education. Nursing programs, in particular, investigated the causes that may be responsible for nursing students’ leaving their programs (Glossop, 2001) and failing at their licensure examination after graduation (Wolkowitz and Kelley, 2009; Alameida et al., 2010).

In general, a causal approach is a top-down way of thinking based on a theory from which a researcher generates testable hypotheses. Generally, the success of causal approach has been limited somewhat, as it places more weight on understanding the mechanisms and less on helping programs identify students at risk (Moseley and Mead, 2008). Educators need to identify those students they can help, while the knowledge of a general mechanism behind students’ difficulties is helpful but often takes too long a time to piece together.

A more data-driven approach focuses on discovering patterns in the data that may lead to correct predictions about individual student outcomes while keeping the mechanisms behind it in the “black box.” Moseley and Mead (2008), for instance, use rule induction to predict who drops out of nursing courses. Their application achieved 94% accuracy on a set of new data. Hung, Hsu, and Rice (2012) apply C&RT Decision Tree analysis to predict student performance and satisfaction levels toward online courses and instructors based on their activity during the class and find that more active students are more satisfied as well.

This study seeks to incorporate many predictors used in previous research, including comprehensive preparatory and exit exams (Alameida, Davis, and Renwanz-Boyle, 2010; Young, Rose, and Willson, 2013; Langford

and Young, 2013), transfer credits and GPA in nursing courses (Simon, McGinniss, and Krauss, 2013), GPA in chemistry (Lockie et al., 2013), math scores (Trofino, 2013), and the number of attempts (or failed attempts) at science courses (Shaffer and McCabe, 2013). At the current stage of this research project, we include 122 predictors, including the grades in all the general education, science, and nursing classes; results of independent ATI assessments; number of failed courses and assessments; and engagement with the preparatory program after graduation.

Dataset and expected strength of predictors

The dataset analyzed here is drawn from the university's main operational database (CampusVue). The data are stripped of any personal identifiers. As Table C.1 illustrates, there are seven groups of predictors included in the models: demographic and socioeconomic, institutional, measures of course and aggregate academic performance, students' scores at entrance exams, independent ATI assessments, and after-graduation preparation computer training called VATI. In total, we

used 122 predictors in the analysis, but only the most important are described in detail to simplify the presentation.

Most of the variables are indicators of students' academic progress during their time at the university. Some of them are traditional and include various grade point averages both cumulative and by field, and the number of failed classes in total and by field. In addition to that, nursing students take independent assessments delivered by a third-party contractor ("ATI Nursing Education") at the end of each of their core nursing classes. Table C.2 lists all the independent assessments students take in the order in which they are introduced in the program.

One may ask whether the ATI assessments as predictors function differently from students' final grades in the corresponding nursing courses. Unlike the latter, the ATI assessment score is not assigned by the faculty teaching the class. Thus these assessments are likely to be closer to an unbiased estimate of students' achievement than final grades, because faculty may, as some works in the literature suggest, inflate the latter to avoid unfavorable student evaluations (Isely and Singh, 2005; Eiszler, 2002; Babcock, 2010). Thus one can expect that ATI assessments should emerge among the strongest predictors of student performance at NCLEX.

Another set of predictors with strong potential consists of measures characterizing the depth of engagement with the exam materials that the students exhibit after they graduate. Among these are the percentage of VATI training completed and VATI Green Light status. Unlike ATI assessments, however, VATI timing (after graduation) makes it less valuable from a practical standpoint.

TABLE C.1 Groups of predictors.

Predictor Groups	Definition	Comments
Demographic and socioeconomic	Stable individual attributes	Race and ethnicity, gender, economic dependency, marital status, etc.
Institutional	Status and "location" within the institution	Program track, campus, history (reentry)
Course performance	Course details	Min, max, average, grades
Aggregate performance	Aggregate measures	Cumulative, science, nursing theory GPA, number of failed courses
Entrance exams	SAT, ACT, HESI scores	Min, max, average, entrance score
Independent assessments	ATI nursing assessments administered as a part of nursing courses (nine in total)	Min, max, average scores in each assessment, total number of assessments failed, unique assessments failed, etc.
Virtual Coach (VATI) training	After graduation NCLEX exam preparation	Received the "green light" to test, percent completed, etc.

TABLE C.2 Full list of ATI assessments in the order nursing students take them.

Sequential Number	Assessment
1.	Registered Nursing Fundamentals
2.	Mental Health
3.	Nutrition
4.	Maternal Newborn
5.	Nursing Care of Children
6.	Community Health
7.	Adult Medical Surgical
8.	Leadership and Management
9.	Pharmacology

IBM SPSS Modeler has the capability to make a preliminary assessment of predictor importance, which evaluates how strongly each of the predictors affects the target. It can be an F statistic (how F changes if you drop a predictor) or a p-value when comparing different groups of observations formed during the classification process. Figure C.1 shows that the number of failed ATI assessments is the best predictor of NCLEX failure, followed by the VATI engagement.

Data mining with IBM SPSS Modeler

Modeling workflow

In IBM SPSS Modeler, one can present the entire modeling process as a workflow. Fig. C.2 presents it graphically. The original data node is located in the upper left corner and is called “Excel.” It shows that the original data file is in Excel format and displays the file name

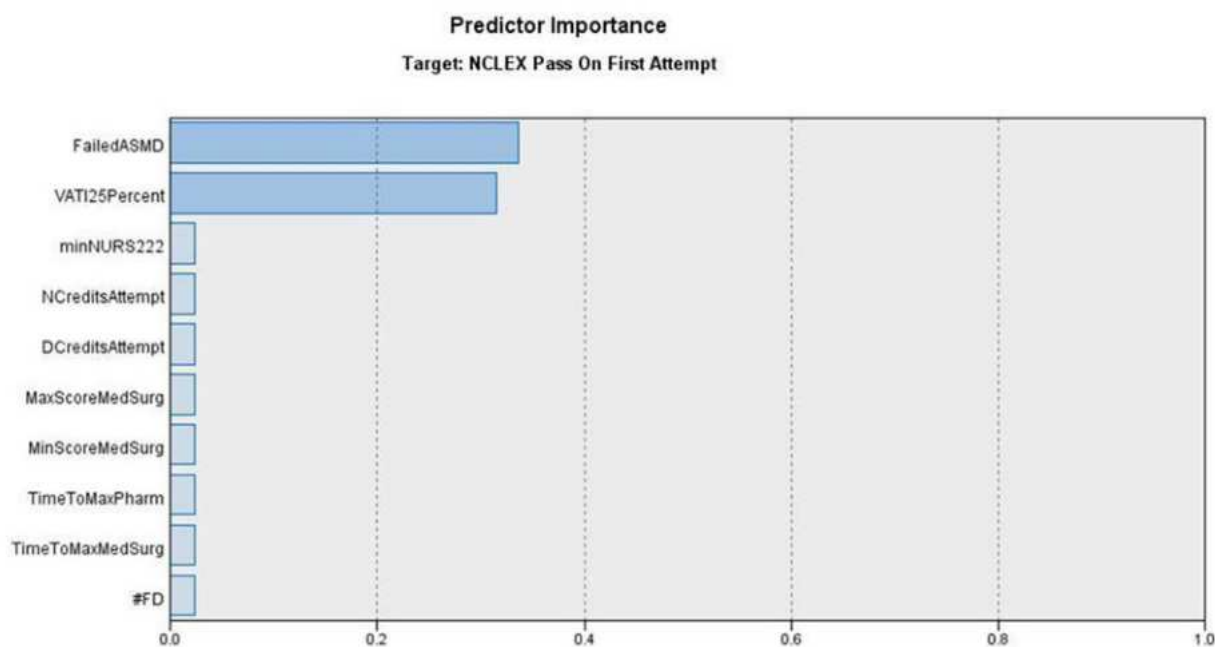


FIG. C.1 Predictor importance in IBM SPSS Modeler.

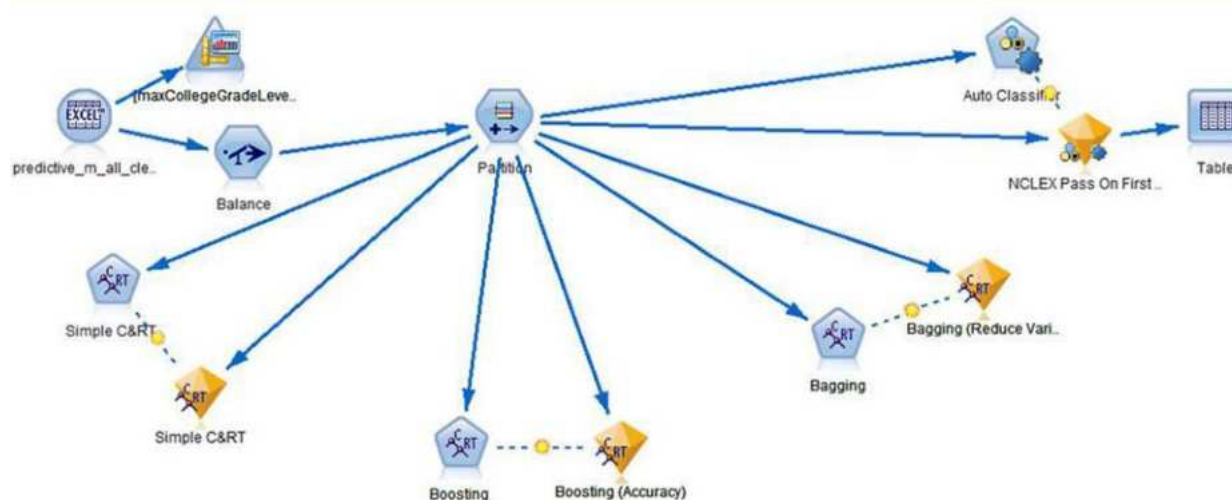


FIG. C.2 Modeling workflow in IBM SPSS Modeler.

below. An arrow pointing up to a triangle with a picture of a ruler inside shows one way a researcher can learn more about the data before starting the modeling process.

Fig. C.3, for instance, presents a histogram characterizing the frequency distribution of nursing students by the number of ATI assessments failed. It shows a shape skewed slightly to the right, with approximately half of the students failing five or fewer assessments. The majority fails fewer than 10. Even this rough estimate gives one an idea that students failing more than 10 assessments in total (including repeats) are likely to have trouble passing their licensure test after graduation.

Before starting any model-building exercise based on data mining techniques, a researcher needs to think

about balancing the sample and partitioning it into the training and testing subsamples. Fig. C.2 shows that the modeling process goes to the right (“East”), to the node called “Balance.” Here the modeling stream “balances” the original data: it takes all the records with NCLEX failures and randomly matches them with the same number of records with positive outcome. This procedure is necessary, because about 80% or more of the students do pass their NCLEX exam on the first attempt, but educators, understandably, need to know about the factors driving NCLEX failure as well.

Fig. C.4 shows the output for a model based on an unbalanced dataset. The coincidence matrix illustrates how the model performs at predicting positive and negative outcomes of the NCLEX.

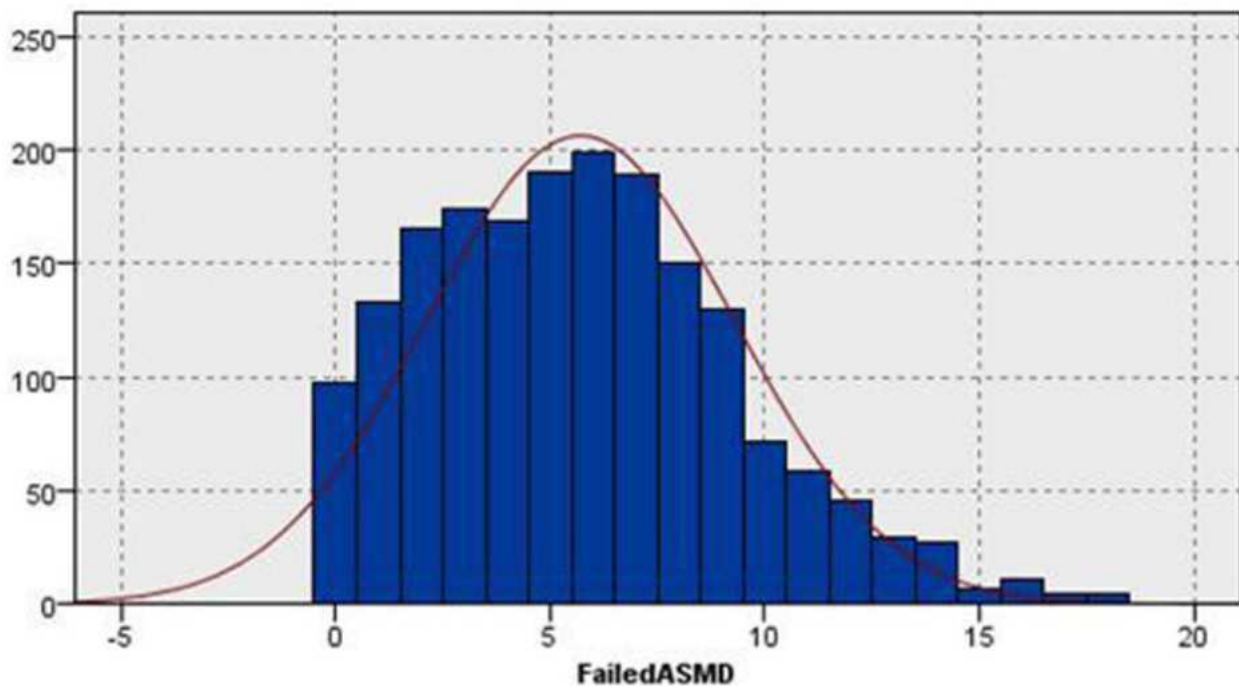


FIG. C.3 Distribution of the number of failed ATI assessments.

3 Coincidence Matrix for \$R-NCLEX Pass On First Attempt (rows show actuals)

'Partition' = 1_Training		NO	YES
NO		49	97
YES		34	1,102
'Partition' = 2_Testing		NO	YES
NO		12	47
YES		18	497

FIG. C.4 Results for a simple C&RT model – unbalanced.

The actual outcomes are shown in the rows of the coincidence table and the predictions are in the columns. One can see there are only 59 NCLEX failures in the testing subsample, and 47 of them (almost 80%) are false-positives (negative outcomes that were predicted as positive). On the contrary, out of 515 actual NCLEX passes, only 18 (3.4%) were false-negatives. This model has much higher accuracy for the positive outcomes when compared with the negative ones. In fact, assuming a 50/50 chance of passing the NCLEX is better than following the guidance of the unbalanced model for NCLEX failures.

To improve model accuracy, one needs to create a sample containing an equal or close to equal number of positive and negative outcomes. The workflow stream in Fig. C.1 does just that by introducing a sample that includes all the negative outcomes, randomly matching them with the equal number of positive outcomes. During the estimation, this process is repeated several times to make sure that the random selection of positive outcomes is not producing an unusual sample just by chance. Fig. C.5 shows the coincidence matrix for the same simple decision tree (C&RT) model, which shows that in the testing subsample, 52% of negative outcomes were correctly predicted as NCLEX failures. This is not great, but it is an improvement over the unbalanced model. There are other and more effective means of improving model accuracy, which are discussed below, but it always helps to consider balancing your sample first.

Improving model accuracy and stability: boosting and bagging

The workflow in Fig. C.1 shows four modeling streams coming out of the partition node that follows the “Balance” node. One of them is the simple decision tree model we discussed in the previous section (“Simple C&RT”). The next two models, called “Boosting” and “Bagging,” represent two elaborations on the simple

C&RT model, improving its accuracy and stability. Fig. C.6 provides a screenshot of the menu associated with building decision tree models. The submenu “Build Options” gives one a choice of creating a single tree, enhancing accuracy by boosting, and/or improving stability by bagging.

Boosting creates an ensemble (or combination) of models. Each of these can have varying overall accuracy. They all, however, have the advantage of working better than alternatives for some parts of the data. A meta-model combining a series of such models is usually more accurate than each of its components. Fig. C.7, for instance, provides the details for each of the component models in the ensemble. One can see that some of the component models have high overall accuracy, while others do not. Still, the boosted decision tree model’s accuracy exceeds 99%, which is greater than the accuracy of any of the component models. One should also always be aware of the possibility of overfitting when using any accuracy-boosting techniques.

In contrast to boosting, bagging seeks to stabilize model outcome from replication to replication. It splits the data into the training and testing subsamples, fits a chosen number of models, and selects the best fit in terms of accuracy by comparing predicted and actual outcomes in the testing sample. Then it starts the process from scratch by creating different training and testing subsamples and picking the winner on the newly split subsets of data. The resulting ensemble combines all the “winners” from each run and averages them out. Fig. C.8 shows the details for each of the bagging model’s components. One can see the accuracy of these component models are very close to one another and, overall, they do create a stable ensemble.

IBM SPSS Modeler automated model selection procedure and evaluation

IBM SPSS Modeler features an automatic model selection procedure that fits all the possible models to the

Coincidence Matrix for \$R-NCLEX Pass On First Attempt (rows show actuals)

'Partition' = 1_Training		NO	YES
NO		39	57
YES		44	367
'Partition' = 2_Testing		NO	YES
NO		56	53
YES		43	377

FIG. C.5 Results for a simple C&RT model – balanced.

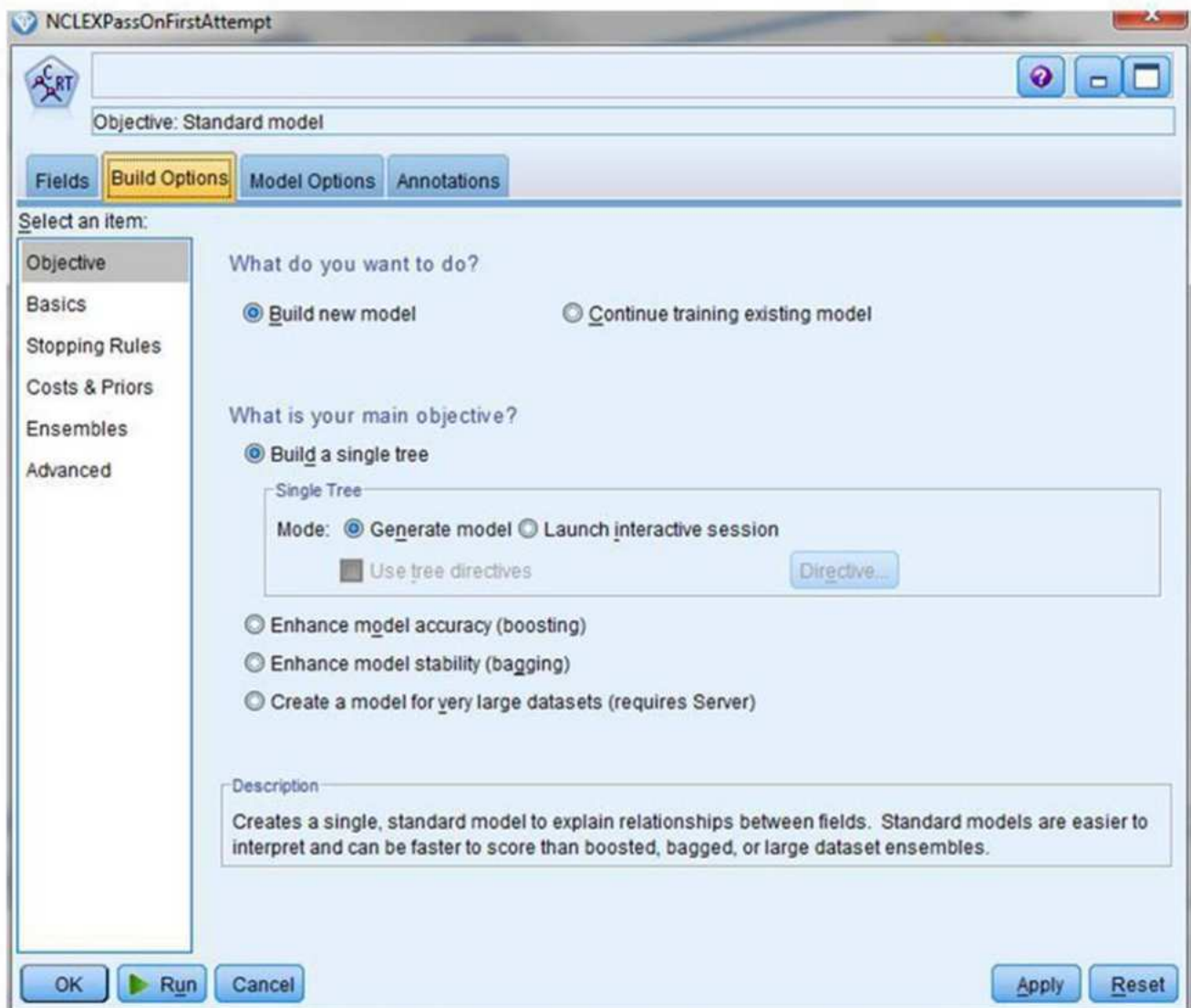


FIG. C.6 Build options for C&RT models.

data, estimates the predictive accuracy of each of them, and finally leaves only those models that feature an accuracy rate higher than a certain threshold set by a researcher in advance. We fixed this threshold at 80%, expecting each of the models to predict NCLEX outcome correctly at least 8 times out of 10.

IBM SPSS Modeler's automated model selection routine relies on a wide variety of statistical and artificial intelligence routines, including decision trees, regression, neural nets, discriminant analysis, Bayes net, and some others. In the output, however, one sees only the models that have passed the 80% accuracy threshold. Fig. C.9 below shows that only three of the candidates – the Quest, CHAID, and C&RT algorithms – passed this accuracy test.

The table in Fig. C.9 is generated automatically and includes several columns that do not make much sense in the higher education context. For example, maximum profit and lift are regularly used in commercial context to reduce the cost of advertising by focusing budgets on the potential customers likely to respond and ignoring those who are not. The predictive model assigns higher scores to those customers who are a better bet from the point of view of spending the budgets or investing in them.

Applying predictive modeling to pinpoint the students likely to have trouble passing the NCLEX exam is not meant to increase profit. Rather, the model lift serves as a "goodness of fit" metric. This case study, however, does not focus on this.

Component Model Details

Model 	Accuracy 	Method 	Predictors 	Model Size (Nodes) 	Records 
1	86.7%		34	13	596
2	79.4%		44	23	596
3	76.8%		28	13	596
4	73.2%		56	23	596
5	77.3%		57	25	596
6	73.0%		44	19	596
7	79.0%		50	21	596
8	75.8%		49	27	596
9	82.4%		72	29	596
10	62.2%		36	13	596

FIG. C.7 Build options for C&RT models – boosting.

Interpreting model output

Evaluating model lift and accuracy is important, but these metrics are not useful in practical terms. Now that we have selected the appropriate models, it is time to see

how they can help divide students into groups according to their likelihood of passing the NCLEX on the first attempt. The figures that follow present the model output as a series of hierarchical rules, becoming more detailed with each level. The blue text in squared

Component Model Details

Model	Accuracy	Method	Predictors	Model Size (Nodes)	Records
1	88.3%		43	19	583
2	88.7%		55	21	583
3	89.2%		47	21	583
4	87.7%		25	11	583
5	86.3%		31	13	583
6	87.1%		32	15	583
7	86.8%		30	13	583
8	86.3%		22	9	583
9	86.8%		40	15	583
10	90.4%		36	17	583

FIG. C.8 Build options for C&RT models – bagging.

brackets shows the mode outcome in each of the groups and subgroups. In the green-colored round parentheses, one can see the number of cases included in the group and the percentage of them who have the mode outcome.

Fig. C.10, for example, demonstrates that CHAID algorithm divides students into seven groups according to the number of independent ATI assessments they failed (“FailedASMD” variable) during their academic career at the university. One can immediately see that

the mode outcome for all the groups with the number of failed assessments less than or equal to 9 is “Yes,” while for students who failed more, it is “No.” This is a useful rule, but it is not sufficiently detailed.

Fig. C.11 details the factors that affect the outcome of the NCLEX for each of the groups isolated at Level 1. For example, students who fail no more than one independent assessment pass the NCLEX with a probability of 92.3% to 100%. Those belonging to the ethnic minority



Use?	Graph	Model	Build Time (mins)	Max Profit	Max Profit Occurs in (%)	Lift(Top 30%)	Overall Accuracy (%)	No. Fields Used	Area Under Curve
☒		Quest 1	< 1	2,031.411	93	1.038	84.743	6	0.603
☒		C&R Tree 1	< 1	1,975	100	1	82.753	222	0.5
☒		CHAID 1	< 1	2,031.304	94	1.2	81.592	19	0.799

FIG. C.9 Automated model selection in IBM SPSS Modeler.

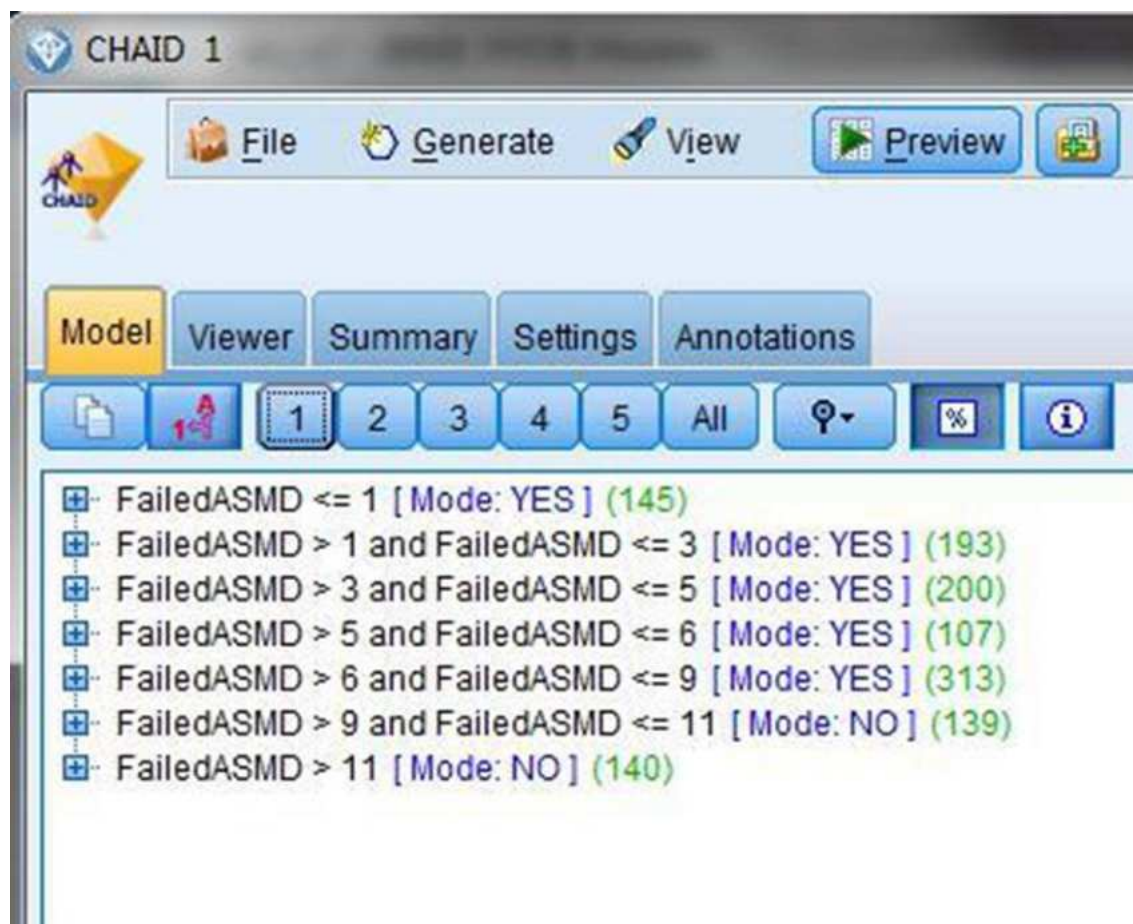


FIG. C.10 CHAID tree – Level 1.

in the student population (that includes Alaska Natives and Pacific Islanders) historically passed the NCLEX at slightly lower rate of 92%, while all others passed at 100%.

The group of graduates who fail two or three independent assessments is subdivided into two subgroups: those who completed at least 50% of the VATI preparatory course and those who have not. The mode outcome

for both groups is an NCLEX pass on the first attempt, but those who complete less than 50% of the VATI program are less likely to pass. The mode outcome for them is still a yes, but if they also fail one of the didactic nursing classes, it switches to a no (the diagram in Fig. C.6 does not show the third level).

Graduates who fail four or five ATI assessments are still likely to pass but, to achieve the passage rate of 95%,

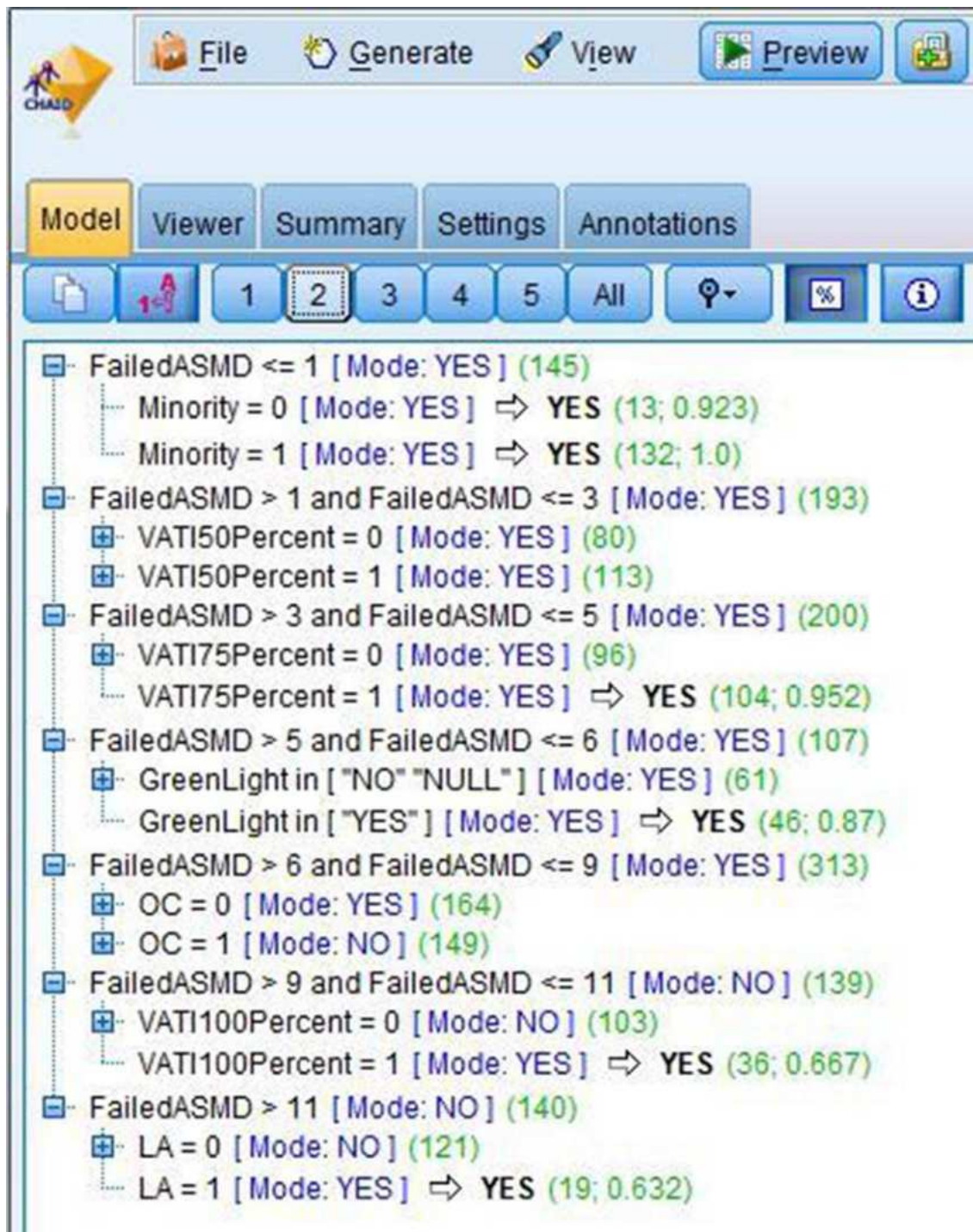


FIG. C.11 CHAID tree – Levels 1–3.

they need to complete at least 75% of the VATI preparatory program. If that threshold is not achieved, students need to have a nursing GPA of at least 3.48 to stay in the group with the mode outcome of pass. Test takers with lower nursing GPAs more frequently fail to pass the NCLEX on the first attempt.

Students who fail six ATI assessments need to achieve the final pass, known as the “Green Light,” in their VATI training program to still have an 87% probability of passing the NCLEX. For those who have not achieved the VATI Green Light, passing the “Nursing Leadership” independent assessment on the first attempt becomes

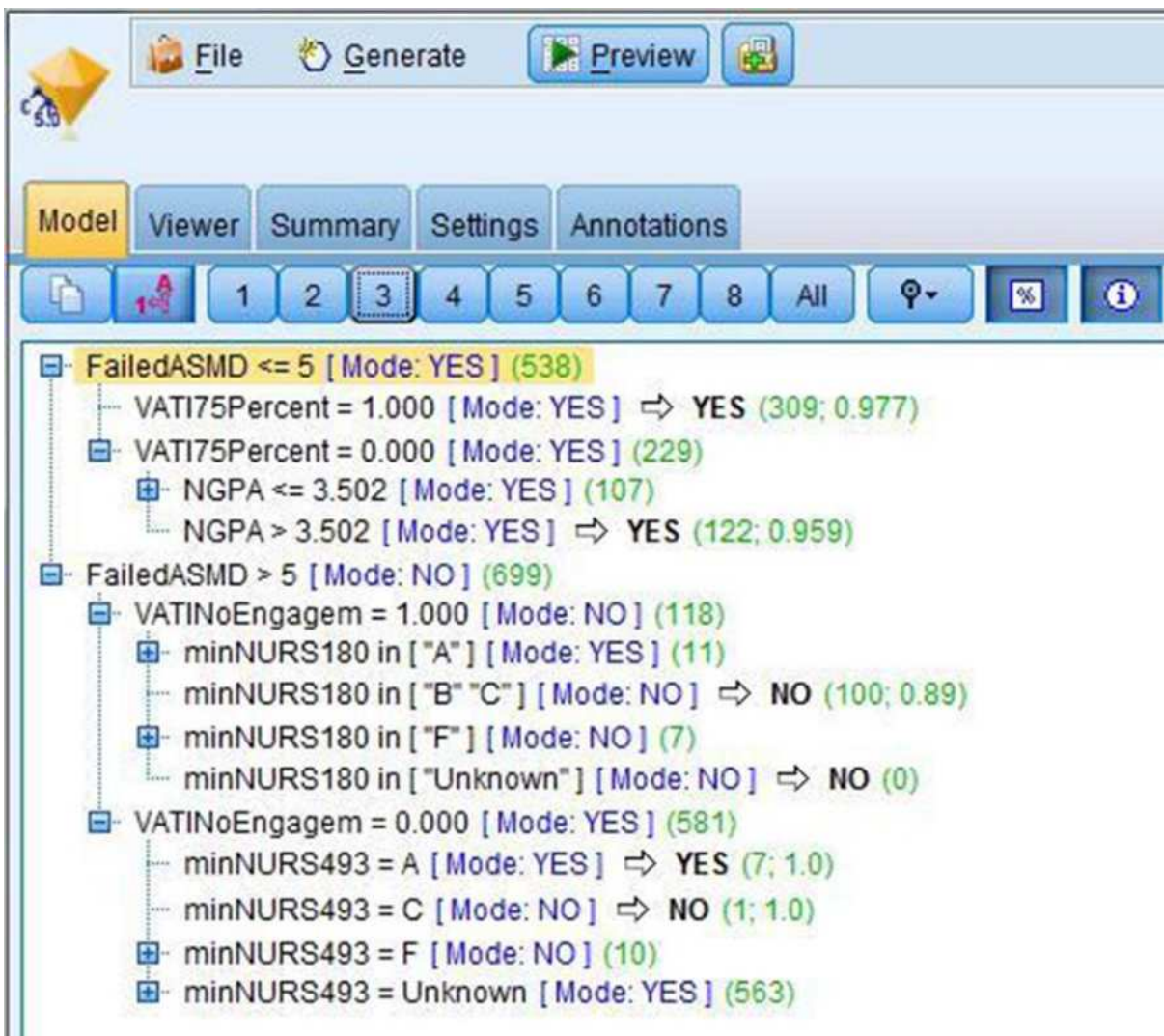


FIG. C.12 C5.0 tree – Levels 1–3.

crucial. If they fail, they end up in the subgroup with the mode outcome of an NCLEX fail.

Graduates failing between seven and nine ATI assessments are still in a group with the mode outcome of a “Yes,” but they are more likely to pass if they are not attending a certain campus or if they achieve the Green Light. All the rest who fail more than nine ATI assessments immediately fall into groups with the mode outcome of an NCLEX fail. These graduates definitely are in need of remediation.

As this example shows, the number of ATI assessment failures can be used as a guiding rule for assigning students to different remediation and study paths that can potentially improve their chances of passing the NCLEX on the first attempt. The CHAID algorithm discussed previously separates groups on the basis of statistical significance between two group averages using the chi-squared distribution. We also fit a model with the SPSS proprietary C5.0 algorithm that splits observations into groups on the basis of information entropy. As you will

observe, the splits that the different algorithms achieve are not necessarily the same, but there is enough of an overlap among them to make the bigger picture clearer.

Keeping that in mind, one can now perform a similar interpretation of the output produced by the C5.0 algorithm. Fig. C.12 shows the first three hierarchical levels of the model. Fig. C.13 shows the first four hierarchical levels of the model. The number of failed ATI assessments is still at the top of the list of the most important predictors. C5.0 draws the first line between graduates who failed no more than five assessments and those who failed more. The mode outcome for the former group is a yes, and for the latter group, it is a no. Depending on whether graduates who failed five or fewer ATI assessments complete 75% or more of the VATI preparatory program, their passing probability ranges between 95% and 100%.

Graduates who failed more than five independent assessments over the course of their studies fall into the

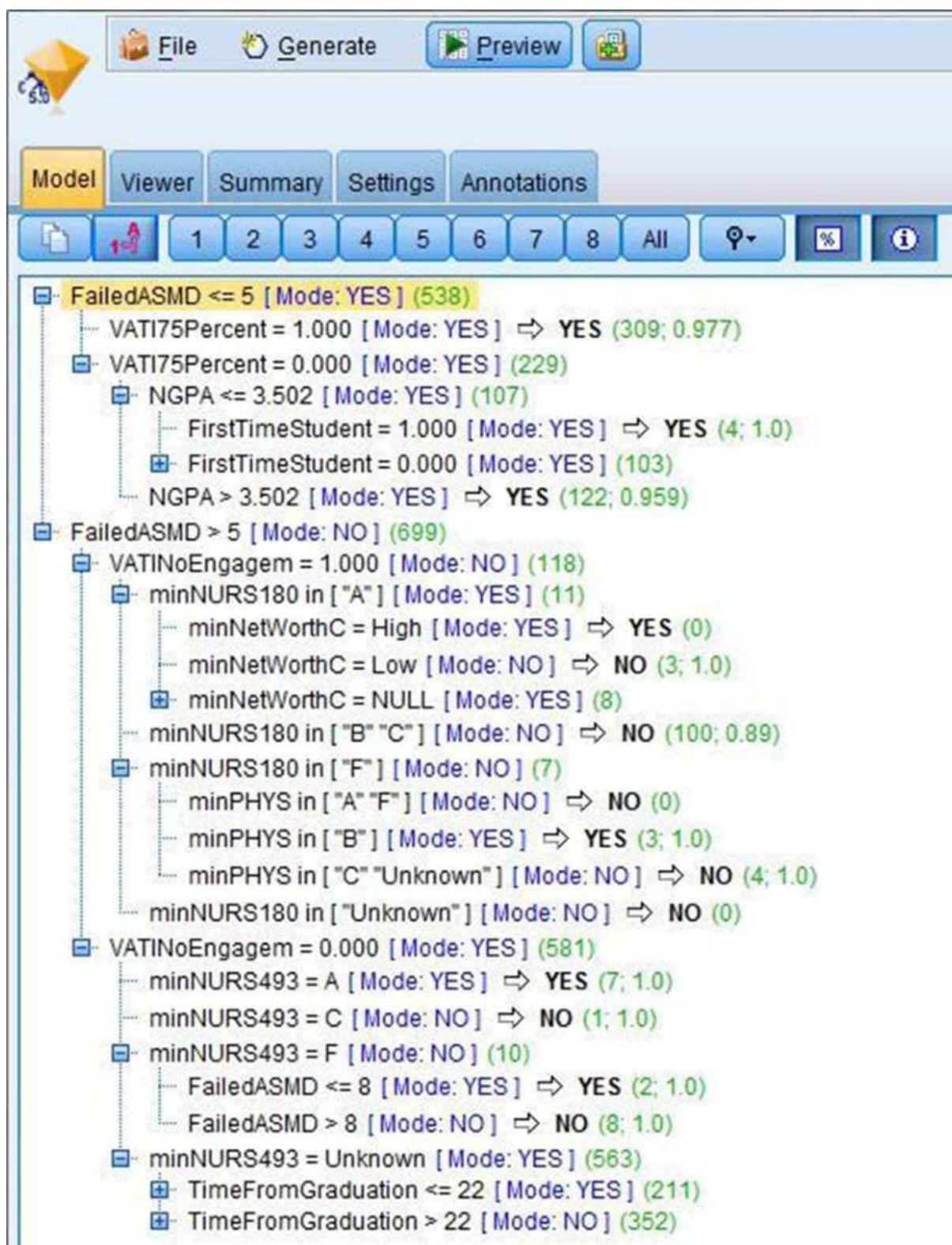


FIG. C.13 C5.0 tree – Levels 1–4.

group with the mode outcome of an NCLEX fail. Not engaging with the VATI preparatory program is another aggravating factor. There are, however, several redeeming factors as well. Passing pharmacology and physics courses with a grade of an “A” puts graduates in subgroups with the mode outcome of “pass NCLEX.” Interestingly, having a higher “Net Worth” as determined by the financial aid department helps struggling graduates to pass the NCLEX on the first attempt, while those with a net worth that is lower than average are less likely to succeed.

Those who did engage with the VATI and were referred to the remediation class “Integration of Nursing Concepts” (NURS493) are likely to pass the NCLEX if they received an “A.” Those who were not referred to the remediation program are more likely to pass if they take NCLEX within 22 months after graduation. The CHAID and C5.0 algorithms do produce slightly different results, but they also share much in common. Among these commonalities, one should point out the number of failed ATI assessments as followed by the graduates’ engagement with the VATI preparatory program.

Data mining with STATISTICA

STATISTICA Data Mining Recipe

STATISTICA does not present the modeling workflow on a canvas like interface – each model is saved separately as a “Data Mining Recipe.” Data Mining Recipe is an automated model selection procedure – an analog to the one in IBM SPSS Modeler – that relies on a slightly different selection of data mining algorithms, especially C&RT, boosted trees, and neural networks. The IBM SPSS Modeler’s automated model selection procedure competitively considers these models among all others; STATISTICA, on the other hand, prioritizes them.

Using the Data Mining Recipe is the easiest way to get a data mining project done in a very short time. The

built-in algorithm performs a competitive evaluation of different models and determines which one has the highest accuracy (or the lowest error rate). Both experienced data miners and those just starting can benefit from using the Data Mining Recipe first, before investigating any further. In many cases, the results may satisfy one’s needs without investing any more time.

To open a new Data Miner Recipe, go to the Data Mining menu and look for the Data Miner Recipes icon (Fig. C.14).

After clicking on the icon, the Data Miner Recipe dialog comes up. Click on the “New” option. Then choose the Open/Connect button on the screen that appears (see Fig. C.15).

After the data is loaded, choose the third down button on the same screen called “Select variables” to specify the dependent variable (target) and predictors. The “NCLEX Pass On First Attempt” variable is the target. It is categorical in essence (Pass or Fail) but coded as numeric (Pass = 1; Fail = 0, so STATISTICA counts it as continuous (see Fig. C.16). We rerun the same analysis using “NCLEX Pass” – a categorical analog of “NCLEX Pass On First Attempt” – as a categorical target to obtain the lift chart and cross tables of predicted and actual outcomes. This description reports the results of both.

Now one needs to decide whether it is worthwhile to conduct additional data preparation procedures. This process may include creating an analytical dataset centered on the unit of analysis of interest and eliminating duplicate cases and suspicious outliers. A researcher can also choose to create a validation sample to use afterward. For the purposes of this study, we created a balanced stratified random sample of students who took the NCLEX. Fig. C.17 shows that the “use sample” option is on.

To set up the stratified random sampling, click on the “Advanced” tab pointed at in Fig. C.18, then click on “Stratified Random” and “Options.” The next screen lets one select the strata variable and presents a choice



FIG. C.14 Data Miner Recipes in STATISTICA.

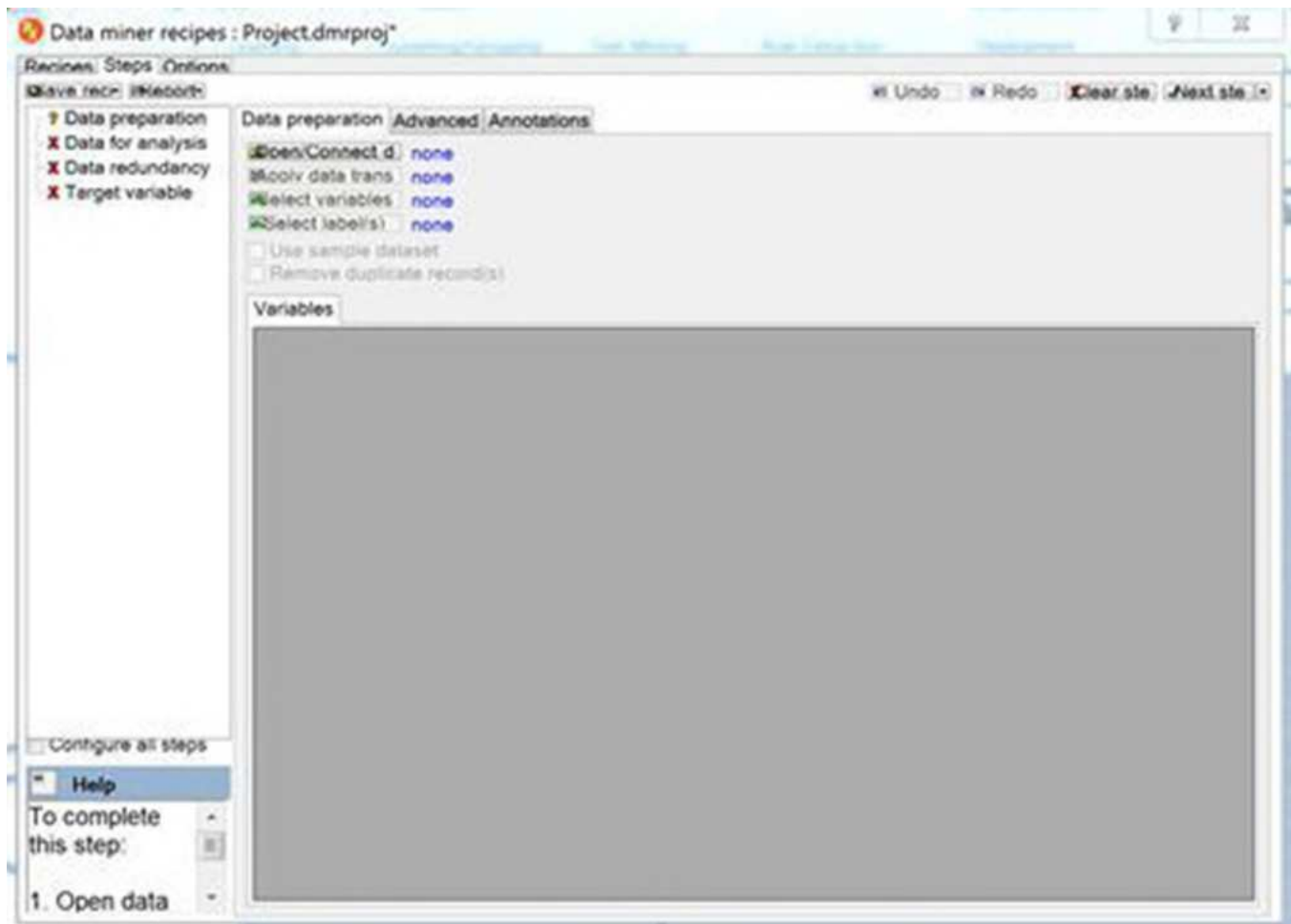


FIG. C.15 Data Miner Recipes in STATISTICA: connect to data.

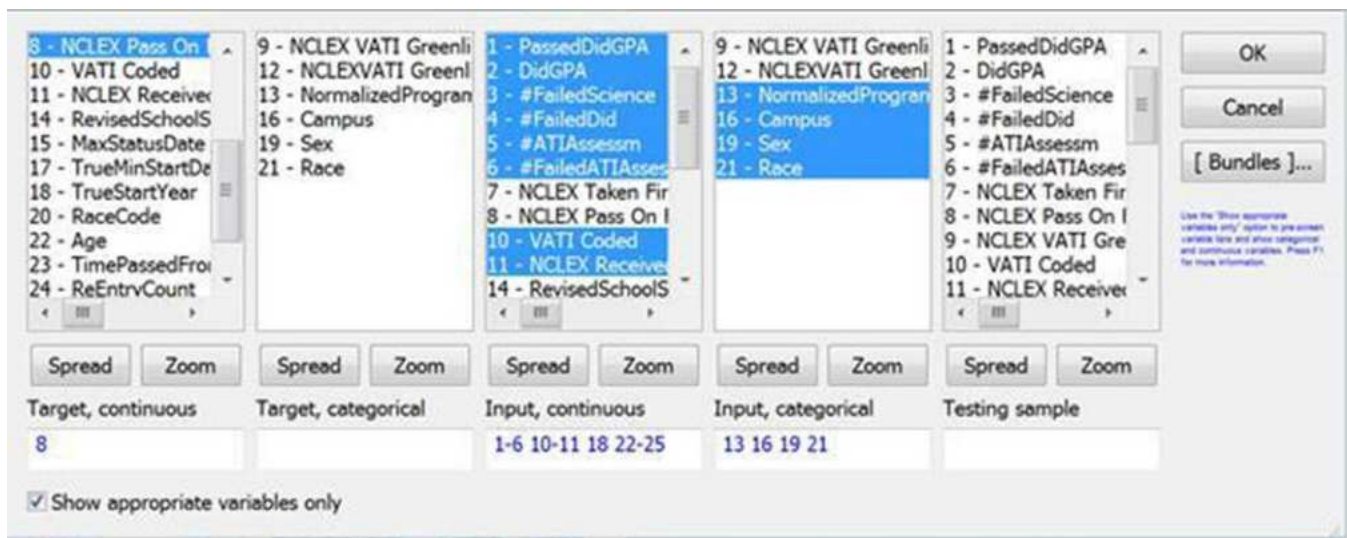


FIG. C.16 Data Miner Recipes in STATISTICA: choose predictors.

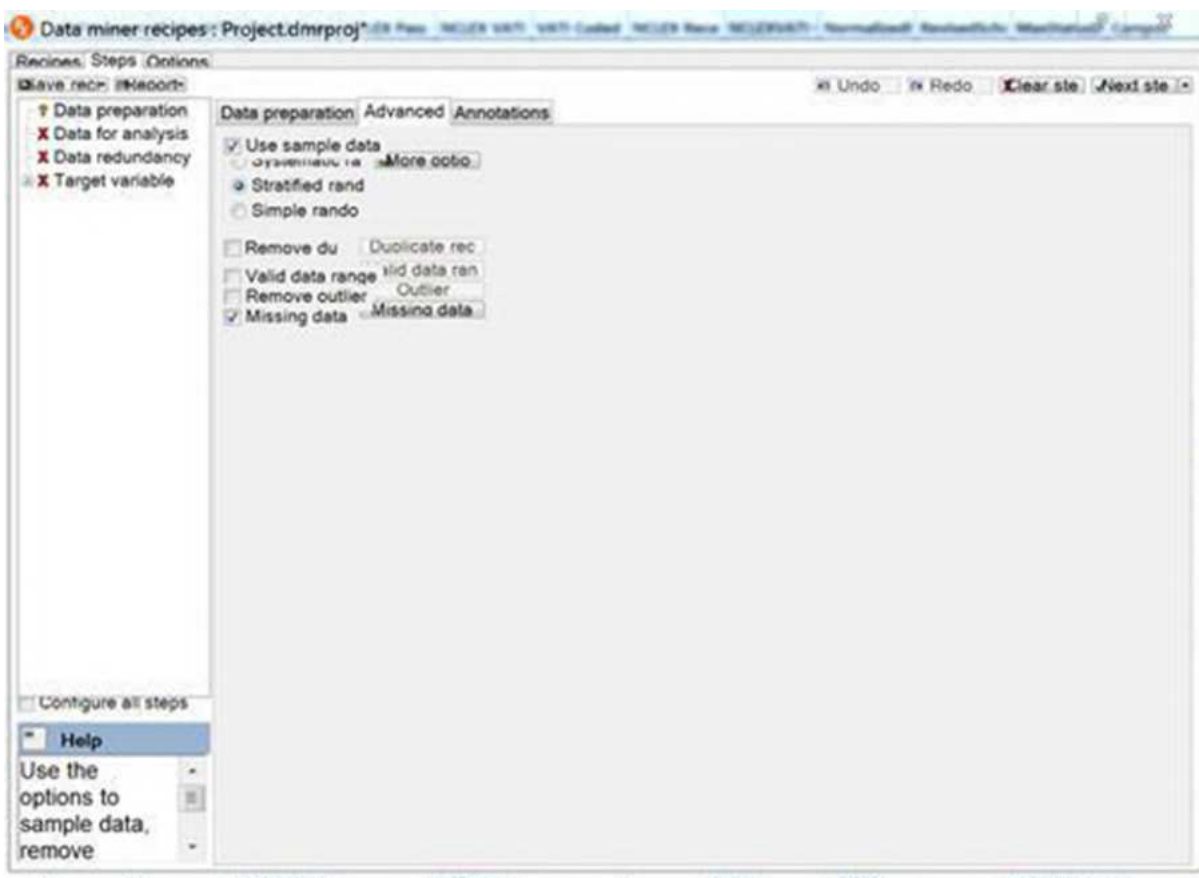


FIG. C.17 Data Miner Recipes in STATISTICA: create a balanced sample.

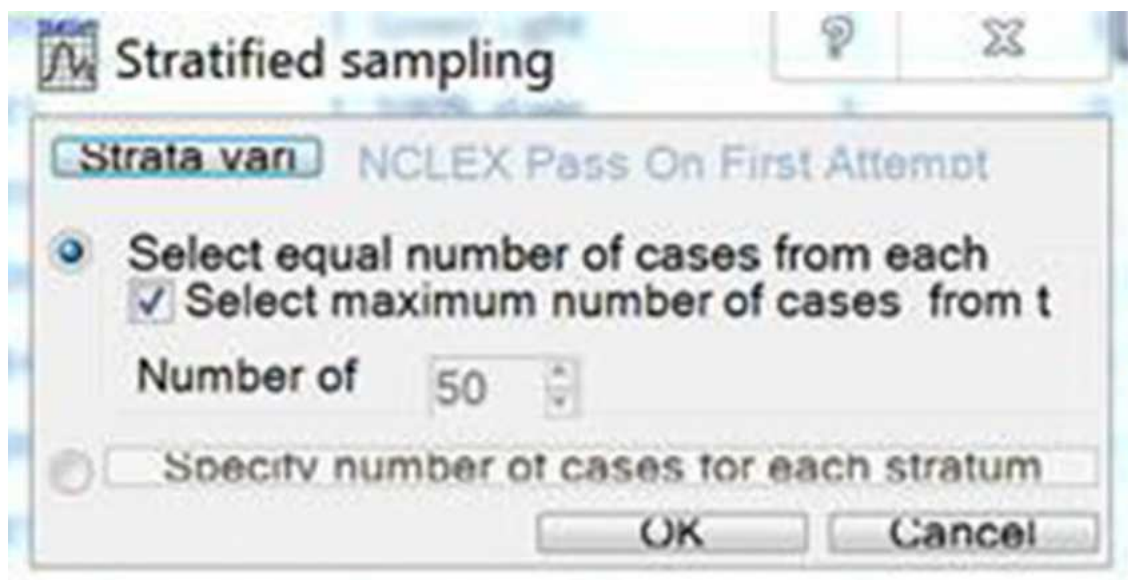


FIG. C.18 Data Miner Recipes in STATISTICA: choose options for stratified sampling.

Testing sample	Percentage of cases	
Specify %	20	
Number of cases in training data	741	
Number of cases in testing data	181	
Table	Step options	
	Date and time	6/15/2016 3:17:58 PM

FIG. C.19 Data Miner Recipes in STATISTICA: review the balanced sample.

Model selected for deployment			
Model Evaluation Summary	Name	Residual (mean square of residuals)(Testing sample)	Correlation coefficient(Testing sample)
	C&RT	0.17	0.57
	Neural network	0.17	0.57
	Boosted trees	0.18	0.55
Table			
	6/15/2016 3:18:28 PM		

FIG. C.20 Data Miner Recipes in STATISTICA: review model accuracy.

	Summary of Deployment (Error rates) (re_entry Validation)		
	1-C&RT	2-Boosted trees	3-Neural
Error rate	0.173362	0.181326	0.170415

FIG. C.21 Data Miner Recipes in STATISTICA: model accuracy comparison.

between an equal and specified number of cases that need to be drawn from each stratum.

We choose “NCLEX Pass/Fail” as the variable defining the strata, because there are significantly more people who pass the exam than those who fail, and we still need to balance our sample. Fig. C.19 lets one review the resulting balanced sample.

At the data analysis stage, the Data Miner Recipe algorithm calculates basic statistics, including the mean, standard deviation, skewness, kurtosis, minimum, and maximum, for the training and testing subsamples that one can review afterward. The testing subsample constituted 20% in this project.

Model output and evaluation

STATISTICA automates the model-building process – it fits a specified number of different types of predictive models (classification and regression trees, boosted trees, support vector machines, or neural networks) and selects the best ones afterward. This is one of the most time-consuming processes within the Data Miner Recipe’s routine. At this stage, STATISTICA fitted three models – C&RT, neural network, and boosted trees – to determine which ones should be evaluated.

At the model evaluation step, Data Miner Recipe’s routine tests the predictive models selected in the previous step by comparing the predicted outcomes to actual ones in the testing sample. Fig. C.20 presents the same information: all three types of models show a similar error rate (accuracy) that changes between 17% and 18% (for a continuous target). Neural networks have the lowest error rate (the complement of the accuracy rate) of 17%. One must note that IBM SPSS Modeler’s C5.0 algorithm generally exhibits a lower error rate ranging from 8% to 11%, depending on the model.

Figs. C.22 to C.24 show the coincidence matrices for each of the models used, including C&RT, boosted trees, and neural networks. The boosted trees model in STATISTICA in Fig. C.23 shows the best and quite high accuracy rate for the NCLEX fails – 85.19%.

Creating rules

Like IBM SPSS Modeler, STATISTICA can create decision trees that may be extremely helpful in the deployment process. Fig. C.25 shows a fragment of the decision tree generated by the boosted trees ensemble fitted on the balanced sample. As one can see, the number of ATI assessment failures and VATI program engagement are still the most important predictors for NCLEX success,

Summary Frequency Table (Prediction)				
Table: NCLEXPass(2) x 1-C&RT				
Prediction(2)				
	NCLEXPass	1-C&RT Prediction Fail	1-C&RT Prediction Pass	Row Totals
Count	Fail	384	91	475
Column Percent		82.58%	18.96%	
Row Percent		80.84%	19.16%	
Total Percent		40.63%	9.63%	50.26%
Count	Pass	81	389	470
Column Percent		17.42%	81.04%	
Row Percent		17.23%	82.77%	
Total Percent		8.57%	41.16%	49.74%
Count	All Grps	465	480	945
Total Percent		49.21%	50.79%	

FIG. C.22 Data Miner Recipes in STATISTICA: review C&RT coincidence matrix.

Summary Frequency Table (Prediction)				
Table: NCLEXPass(2) x 2-Boosted trees				
Prediction(2)				
	NCLEXPass	2-Boosted trees Prediction Fail	2-Boosted trees Prediction Pass	Row Totals
Count	Fail	397	78	475
Column Percent		85.19%	16.28%	
Row Percent		83.58%	16.42%	
Total Percent		42.01%	8.25%	50.26%
Count	Pass	69	401	470
Column Percent		14.81%	83.72%	
Row Percent		14.68%	85.32%	
Total Percent		7.30%	42.43%	49.74%
Count	All Grps	466	479	945
Total Percent		49.31%	50.69%	

FIG. C.23 Data Miner Recipes in STATISTICA: review boosted trees coincidence matrix.

which is similar to what we previously observed in models selected by IBM SPSS Modeler.

The tree graph in Fig. C.25 starts on the node where students' chances of success on the NCLEX are about 50% (the probability of passing in the root node of

the balanced sample is 48%). Failing less than seven ATI assessments boosts the probability of passing to 74% on average. However, this group has a mixture of individuals with varying degrees of engagement with the VATI preparatory program.

Summary Frequency Table (Prediction)				
Table: NCLEXPass(2) x 3-Neural network Prediction(2)				
	NCLEXPass	3-Neural network Prediction Fail	3-Neural network Prediction Pass	Row Totals
Count	Fail	382	93	475
Column Percent		77.64%	20.53%	
Row Percent		80.42%	19.58%	
Total Percent		40.42%	9.84%	50.26%
Count	Pass	110	360	470
Column Percent		22.36%	79.47%	
Row Percent		23.40%	76.60%	
Total Percent		11.64%	38.10%	49.74%
Count	All Grps	492	453	945
Total Percent		52.06%	47.94%	

FIG. C.24 Data Miner Recipes in STATISTICA: review neural network coincidence matrix.

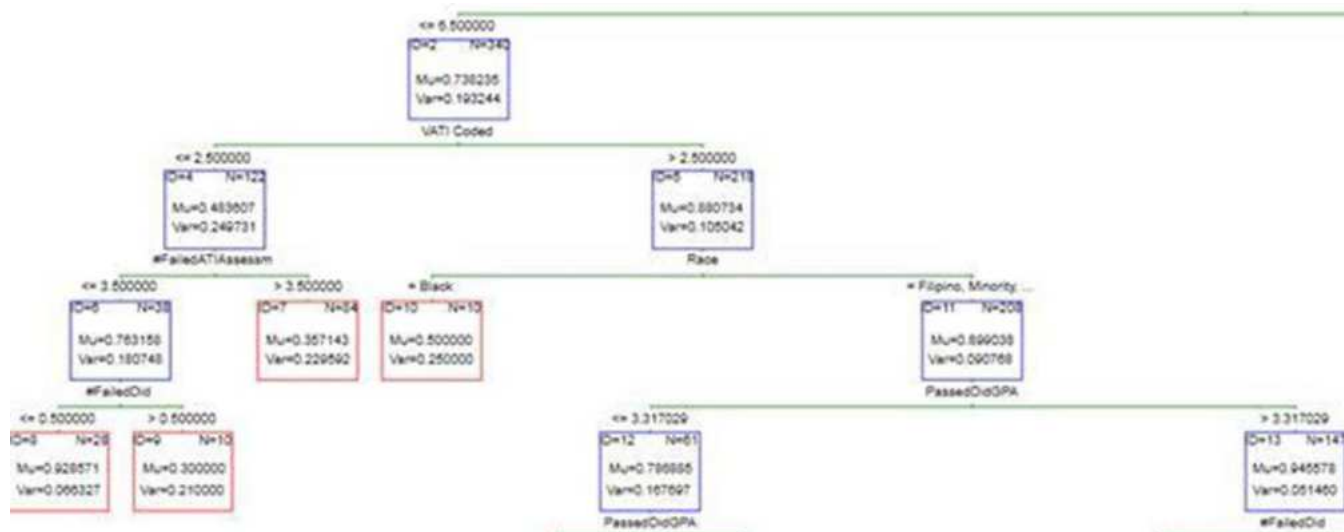


FIG. C.25 Data Miner Recipes in STATISTICA: review the balanced sample.

Those with more than half of the VATI program done increase their probability of passing to 88% or higher, depending on whether they have done well in didactic nursing classes. Those who complete less than half the VATI program decrease their chances back to 48%. If these individuals fail less than four ATI assessments, their chances may still vary from 30% to 92%, depending on whether they failed more than one didactic nursing class.

Among the students who fail more than seven ATI assessments (Fig. C.26), the pass rate drops to 26%. It seems that even finishing the VATI coach program and receiving the Green Light bumps their chances up to only about 40%.

Conclusion

Higher education analysts have not generally been in the vanguard of implementing predictive methods in their work. Yet, predicting various student outcomes, including retention, graduation, placement, and licensure exam passage rates can provide college administrators with valuable information about their students and graduates and may help devise ways to assist those at risk before it is too late. This case study is an illustration of how one can approach such a problem and what can be done in a reasonably short period of time.

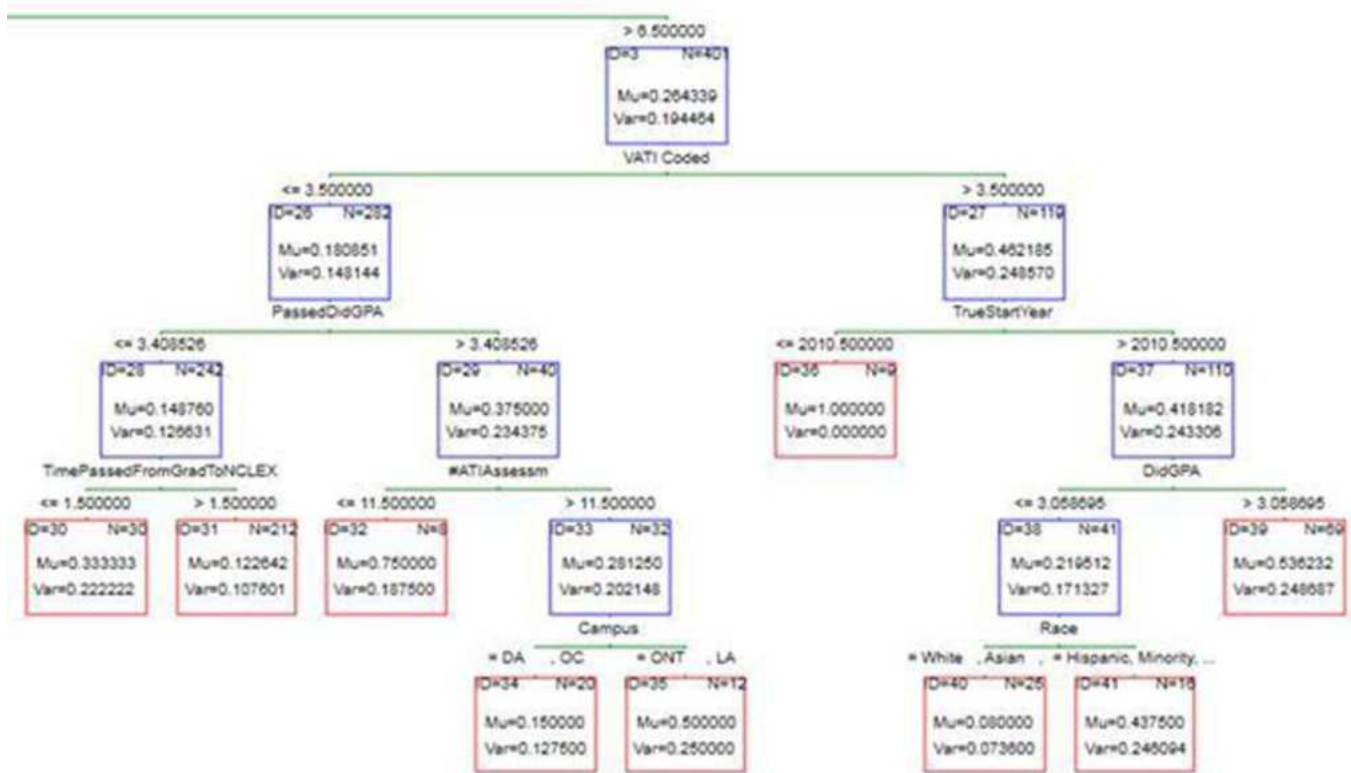


FIG. C.26 Data Miner Recipes in STATISTICA: review the balanced sample.

While working on this project, we have developed several principles that higher education researchers, as well as analysts in other industries, can benefit from. First, it is definitely worth searching for strong predictors that make sense from your theory's standpoint before moving on to utilizing complex data mining techniques in hopes of making your many weak predictors work better.

In many ways, consumer goods companies that have been at the forefront of applied data mining research have had a disproportionately large influence on the way data mining procedures developed. These companies operate in a world lacking credible information: quite often, their researchers work with self-reported data from consumers or potential buyers, and the quality of such data can never be fully ensured. Higher education analysts, on the other hand, usually have access to a wealth of information about their students, from their demographic and socioeconomic profiles to day-to-day performance. Assembling, restructuring, and making use of this information is the most important part of any predictive analytics project in higher education.

Simply collecting the data and incorporating it into your models may not be sufficient either. Sometimes, the predictors need to be transformed or combined before they become useful. For instance, in our project, we go

through several different measures of students' performance at ATI assessments. There are at least three ways to construct such a measure: as the level (from one to three) that students achieve on each independent assessment; as the number of unique assessments a student failed (or passed); or as the number of assessments failed, counting all the retakes. It turns out that the last measure is the most effective at separating students at risk of failing the NCLEX, but we did not know that in advance.

The quality of your predictors is likely to have a significant impact on the stability of your models. Data mining algorithms rely on sampling and simulation techniques and, therefore, the results (rules, decision trees, classifiers, etc.) may differ quite significantly, depending on the particular algorithm used, the number of replications, and/or the methods of creating ensembles. As this study shows, these differences are not dramatic – the big picture remains quite stable and provides practitioners with many useful clues. The stability, however, only holds if the predictors are as strong as the number of ATI assessment failures and VATI engagement are, in the case of our nursing students. We could be left with a much less stable and almost unusable model if the strong predictors were to be removed from our models. The previous rendition of our NCLEX success data

mining project relied on students' GPAs in different disciplines as the main predictor, and the resulting model was more volatile.

These three principles can inform those researchers who have just begun working on their data mining projects and are thinking through their software choices. As this case study illustrates, proprietary data mining software packages do exploit slightly different algorithms and may produce results that are not identical to each other. Yet, these differences are minor if the models use strong predictors. Working with weak predictive variables is more challenging: variations in algorithms and ensemble-building routines utilized may lead to more significant variations in output. Still, one is likely better off focusing on their research design and data collection processes before blaming software packages for mixed results.

Essentially, the choice of a proprietary data mining package probably should be based on other characteristics: user friendliness, cost, maintenance, availability of skills, or usability of help files. Your pick may also be driven by more idiosyncratic factors like the presence of a particular feature – stratified random sampling in STATISTICA or C5.0 algorithm in IBM SPSS Modeler, for instance. In short, our comparison of IBM SPSS Modeler and STATISTICA shows very little difference in terms of performance – the packages did deliver very similar results.

References

- Alameida, M., Davis, H. C., & Renwanz-Boyle, A. (2011). Predicting NCLEX-RN success in a diverse student population. *Journal of Nursing Education*, 50(5), 261–267.
- Babcock, P. (2010). Real costs of nominal grade inflation? New evidence from student course evaluations. *Economic Inquiry*, 48, 983–996.
- Bristol, T. (2012). The National Council Licensure Examination across the curriculum: low-tech learning strategies for student success. *Teaching and Learning in Nursing*, 7, 80–84.
- De Lima, M. (2011). Looking at the past to change the future: a retrospective study of associate degree in nursing graduates' National Council Licensure Examination scores. *Teaching and Learning in Nursing*, 6, 119–123.
- Eiszler, C. F. (2002). College students' evaluations of teaching and grade inflation. *Research in Higher Education*, 43, 483.
- Glossop, C. (2001). Student nurse attrition from pre-registration courses: investigating methodological issues. *Nurse Education Today*, 21(3), 170–180.
- Hand, D., Mannila, H., & Smyth, P. (2001). *Principles of Data Mining* (pp. 546). The MIT Press.
- Horton, C., Polek, C., & Hardie, T. L. (2012). The relationship between enhanced remediation and NCLEX success. *Teaching and Learning in Nursing*, 7, 146–151.
- Hung, J. -L., Hsu, Y. -C., & Rice, K. (2012). Integrating data mining in program evaluation of K-12 online education. *Educational Technology and Society*, 15(3), 27–41.
- Hyland, J. (2012). Building on evidence: interventions promoting NCLEX success. *Open Journal of Nursing*, 2, 231–238.
- Isely, Paul, & Singh, Harinder (2005). Do higher grades lead to favorable student evaluations? *The Journal of Economic Education*, 36(1).
- Langford, R., & Young, A. (2013). Predicting NCLEX-RN success with the HESI exit exam: eighth validity study. *Journal of Professional Nursing*, 29(25), S5–S9.
- Lockie, N., Van Lanen, R., & McGannon, T. (2013). Educational implications of nursing students' learning styles, success in chemistry, and supplemental instruction participation on National Council Licensure Examination-Registered Nurses performance. *Journal of Professional Nursing*, 29(1), 49–58.
- Moseley, L. G., & Mead, D. M. (2008). Predicting who will drop out of nursing courses: a machine learning exercise. *Nurse Education Today*, 28, 469–475.
- Schroeder, J. (2013). Improving NCLEX-RN pass rates by implementing a testing policy. *Journal of Professional Nursing*, 29(2S), S43–S47.
- Shaffer, C., & MacCabe, S. (2013). Evaluating predictive validity of preadmission academic criteria: high-stakes assessment. *Teaching and Learning in Nursing*, 8, 157–161.
- Shaffer, C., & McCabe, S. (2013). Evaluating the predictive validity of preadmission academic criteria: high-stakes assessment. *Teaching and Learning in Nursing*, 8, 157–161. <https://doi.org/10.1016/j.teln.2013.07.005>.
- Simon, E. B., McGinniss, S. P., & Krauss, B. J. (2013). Predictor variables for NCLEX-RN readiness exam performance. *Nursing Education Research*, 34(1), 18–24.
- Thomas, M. H., & Baker, S. Scott (2011). NCLEX-RN success: evidence-based strategies. *Nurse Educator*, 36(6), 246–249.
- Trofino, R. (2013). Relationship of associate degree nursing program criteria with NCLEX-RN success: what are the best predictors in a nursing program of passing the NCLEX-RN the first time? *Teaching and Learning in Nursing*, 8, 4–12.
- Wolkowitz, A. A., & Kelley, J. (2010). Academic predictors of success in a nursing program. *Journal of Nursing Education*, 49(9), 498–503.
- Young, A., Rose, G., & Willson, P. (2013). Online case studies: HESI exit exam scores and NCLEX-RN outcomes. *Journal of Professional Nursing*, 29(2S) p. S.

Uncited references for Additional Information

Alameida et al. (2011); Bristol (2012); De Lima (2011); Hand et al. (2001); Horton et al. (2012); Hyland (2012); Schroeder (2013); Shaffer, MacCabe (2013); Thomas, Baker (2011); Wolkowitz, Kelley (2010)

D

Constructing a Histogram using MidWest Company Personality Data with KNIME

Linda A. Miner, Original Tutorial

Robert Nisbet, Rewritten for KNIME

The following tutorial instructs the user in forming a Histogram in KNIME using data I called MidWest Company Personality Data, copyrighted in 2004 by Right Brain, Inc. The company is fictitious, though the data are “real” in that they have been created from actual cases.

First-time KNIME users will be trained in the tutorial to download the KNIME Analytics Platform program and configure the user interface for use with the tutorials in this book that uses the KNIME Analytics Platform as a tool.

First, download the CSV data, which may be obtained from Elsevier’s companion Web page into a Data Sets folder on your computer. To download the current version 5 of KNIME, go to KNIME.org and click on the Download KNIME Analytics Platform box. See Fig. D.1.

Steps in the download process:

1. Enter your contact information on the next screen (Fig. D.2), then click download.
2. Go to the section for your operating system (Windows, Linux, or Mac).
3. Select: KNIME Analytics Platform for Windows (installer).
4. Follow the prompts to set up the program on your computer.

Changing to the KNIME classical interface

The default format of the KNIME Analytics Platform version 5 uses the modern user interface. The KNIME tutorials in this book use the older classical user interface. To change the default user interface, click on the Menu tab on the upper right of the initial screen you see when loading the KNIME Analytics Platform (Fig. D.2) and select the option to Switch to classical user interface.

Opening KNIME

1. In the classical user interface screen, click on File in the top menu -> New -> New KNIME Workflow, as may be seen in Figs. D.3 and D.4.
2. Select the *New KNIME Workflow* option on the screen.
3. Click on the *Next* box at the bottom of the screen.
4. The next screen gives you the option to name the workflow *Histogram*, as shown in Fig. D.5.
5. Click *Finish* to create the new Histogram workflow.
6. Fig. D.6 shows a list of workgroups for a given user.
7. To load a data set, we must select the proper IO node in the Node Repository window.
 - a. Click on the IO node category.
 - b. Double-click on the Read subgroup.
 - c. Double-click on the *File Reader* node, as shown in Fig. D.7, to paste the node onto the workflow screen. Note: you can also drag and drop nodes onto the workflow screen.
8. Right-click on the file reader node to configure it.
 - a. Navigate to where you have stored the Tutorial D MidWest Personality Data C 2004-Right Brain Inc. csv file and click on it.
 - b. Double-click on the File Reader node to configure it in the screen shown in Fig. D.8.
 - c. Click on browse and navigate to where you saved the MidWest data file, as shown in Fig. D.9.
9. Click Open to see the data contained in Fig. D.10.
 - a. Click OK to return to the workflow screen.
 - b. Notice the traffic light symbol under the node. A red light to the left means that the node is not configured properly; a yellow light means the node is configured properly, which we see in the workspace at this time.

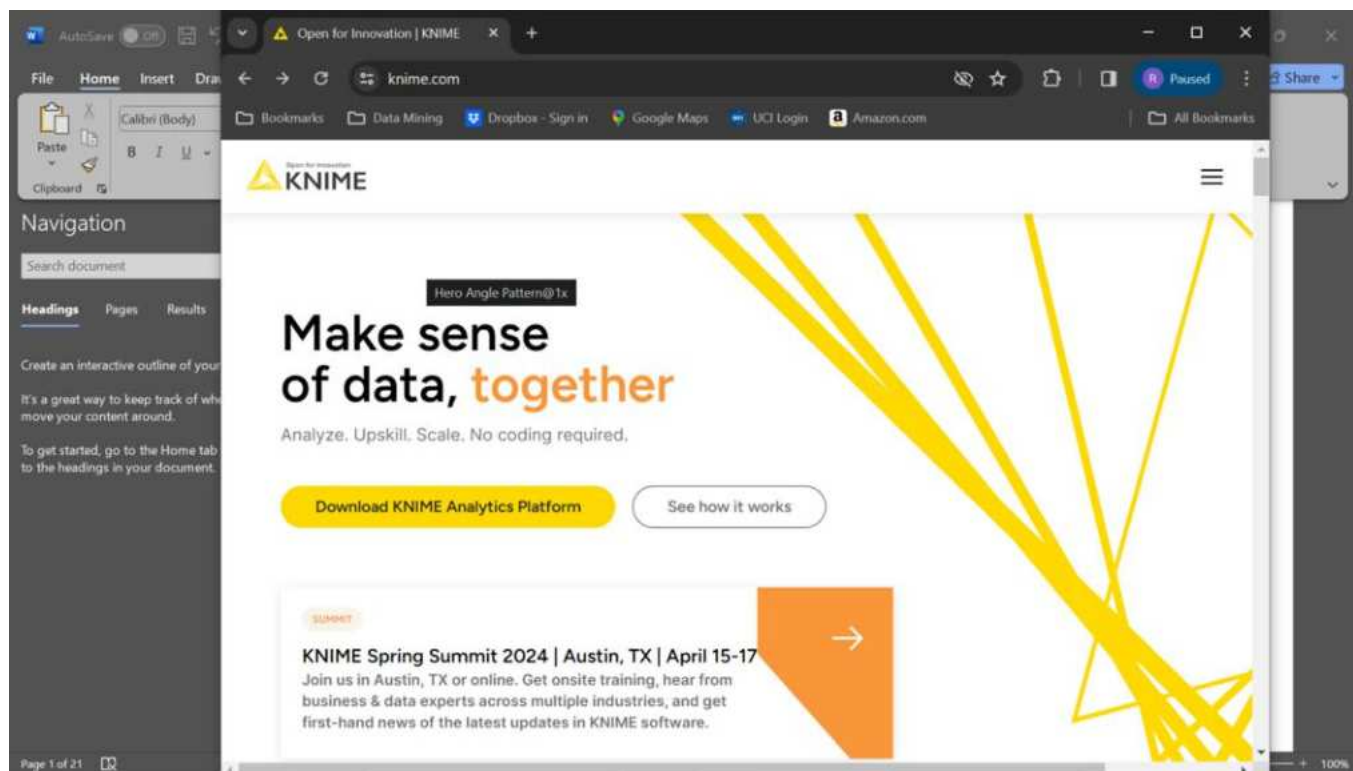


FIG. D.1 The KNIME download option.

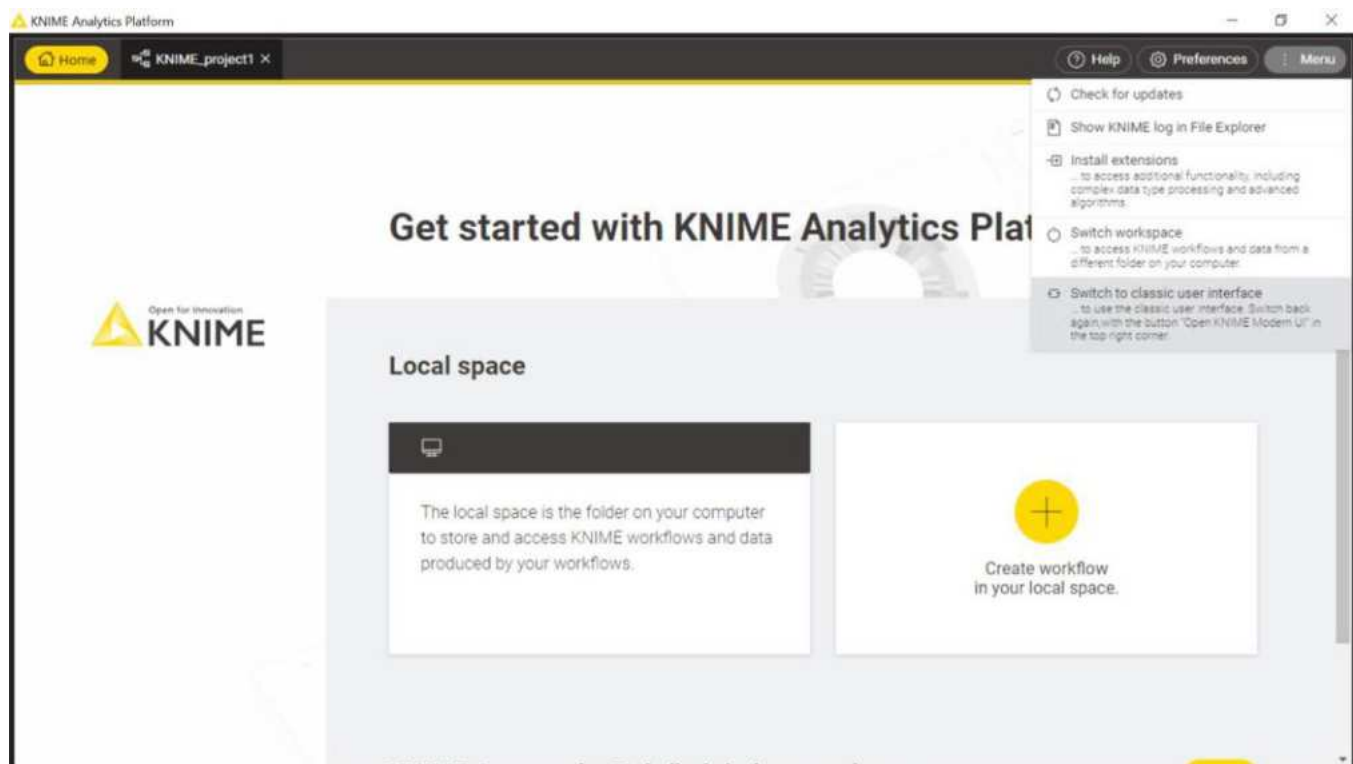


FIG. D.2 Selection of the classical user interface.

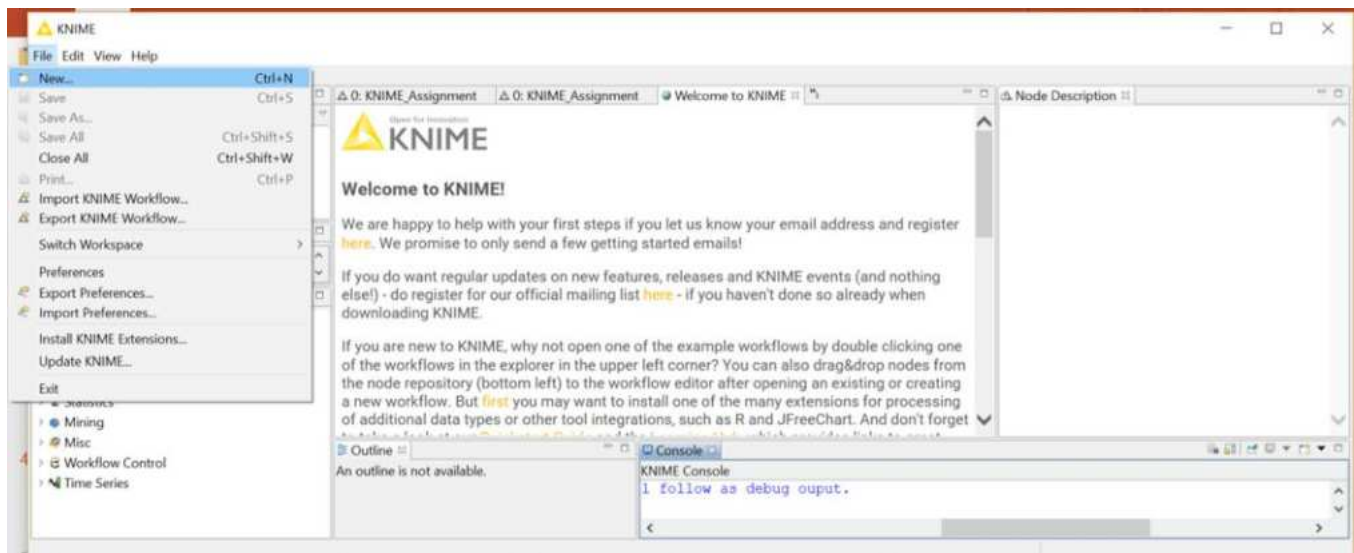


FIG. D.3 Click on File and then New to create a new workflow or workflow group. A workflow group is simply a category name of a group of workflows. For example, you can create a workflow group for Data Preparation workflows. For this tutorial, select the New KNIME Workflow option.

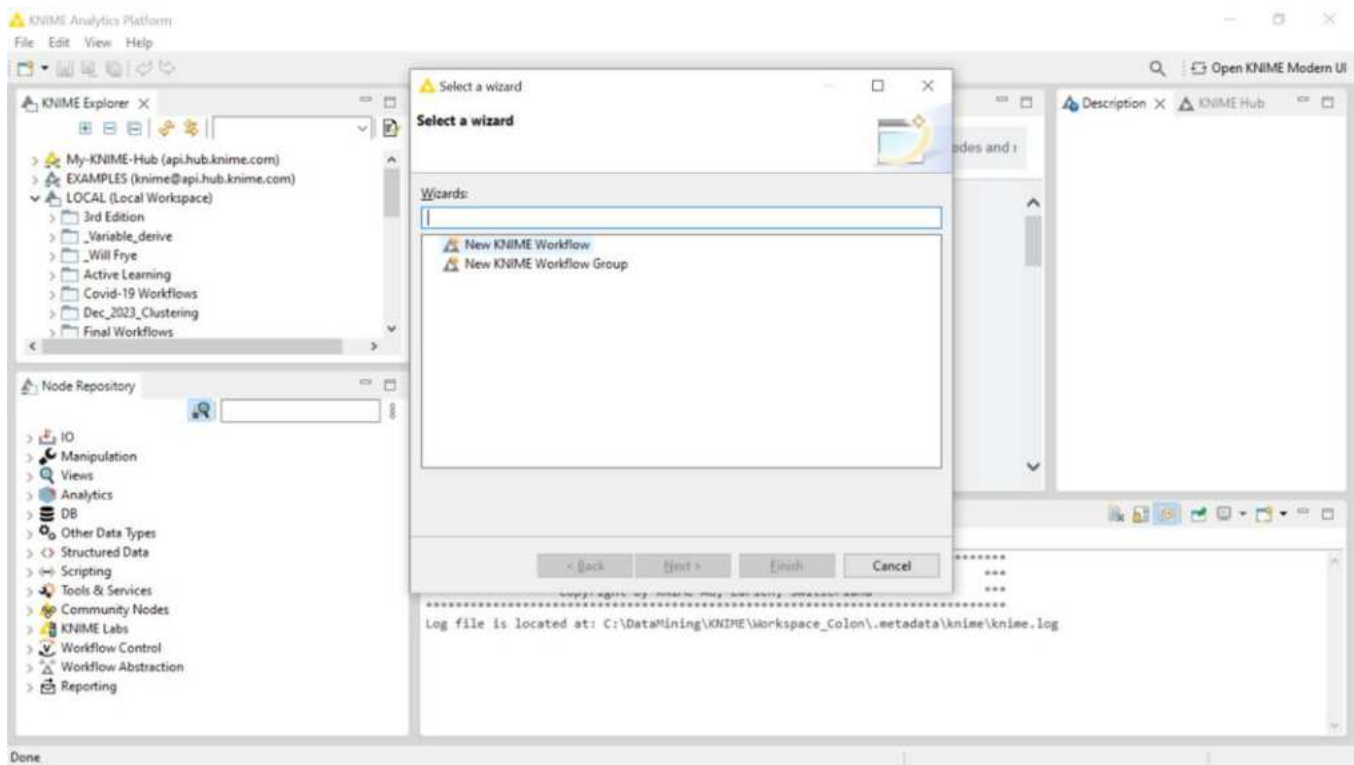


FIG. D.4 Click on New KNIME workflow option.

- c. Right-click on the File Reader node and select the *Execute* option.
- d. After execution is completed, the traffic light symbol will show a green light.
10. Click on the search string in the Node Repository and enter Histogram.
- a. Select the Histogram (Legacy) node under Data Views and then see the Histogram node, as in Fig. D.11. Drag the Histogram to the workflow (Fig. D.12).
11. Left-click and hold the cursor on the File Reader node, then drag and drop it onto the workflow, as shown in Fig. D.12.

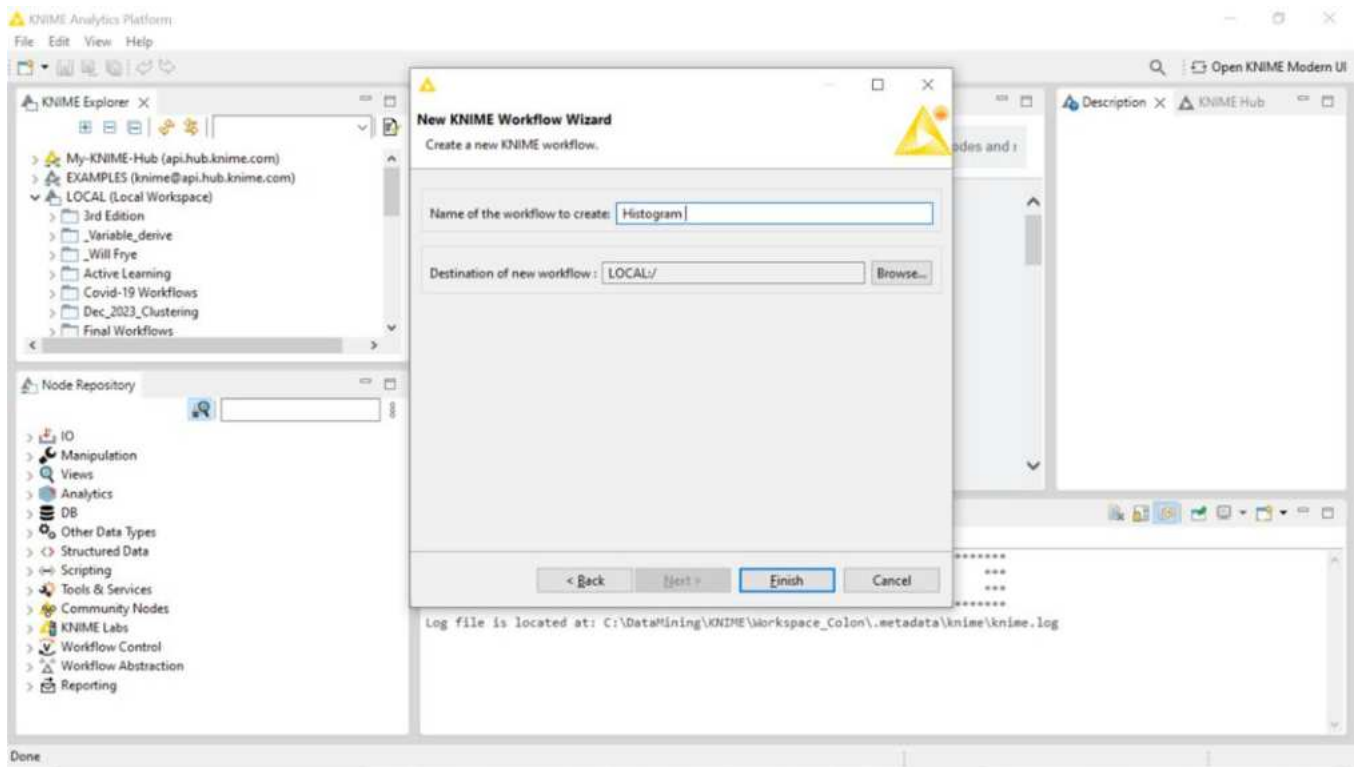


FIG. D.5 Specifying the workflow name and the destination folder in the KNIME Explorer workflow repository. Any workflow groups you create in Fig. D.5 will be listed as subfolders under the LOCAL/home group.

- a. We will connect the nodes later.
- b. Alternatively, you can also highlight the File Reader node in the workflow and double-click on nodes in the Repository to paste the node automatically onto the workflow screen and connect it automatically to the highlighted node.
12. Fig. D.13 shows how to drag a connection from the data to the Histogram node, and we could configure from there.
 - a. To connect two nodes, click on the right-hand output port of the left node and draw an arrow to the left-hand input port of the node right-hand node.
 - b. You can also highlight the File Reader node and double-click on the new node in the Node Repository to connect it automatically to the File Reader node.
13. We can add some color to the Histogram with the color management node.
 - a. Enter the word color into the search box on the Node Repository to display the Color Manager.
 - b. Note: The search box in the Node Repository uses a fuzzy matching algorithm to list all nodes similar to the search string. This capability is very useful for finding specific nodes among the hundreds that may be in the Repository.
 - c. Drag the Color Manager into the workflow space.
14. Fig. D.14 shows how to use the fuzzy search capability to find the Color Manager node.
15. We can use another feature of the KNIME classical user interface to insert a node into a connection between two existing nodes.
 - a. Drag and drop the Color Manager node onto the workspace.
 - b. Drag the white arrow symbol on the Color Manager node over to the connection between the File Reader node and the Histogram node until the connection turns red, then release the mouse button.
 - c. See the inserted node in Fig. D.15.
16. Hover the mouse pointer over the yellow exclamation point under the Color Manager to read the notification that the default column selected is "Years with the Company."
17. We will execute the default color mapping before changing it.
 - a. Right-click on the Color Manager and select the Execute option (Fig. D.16).
18. Double-click on the Color Manager to display the configuration dialog screen.
 - a. In the Available Columns window, click on *Creativity*.
 - b. Click on the Add button to add the variable to the Aggregated columns window, as shown in Fig. D.17.
 - c. Click OK to return to the workflow screen.

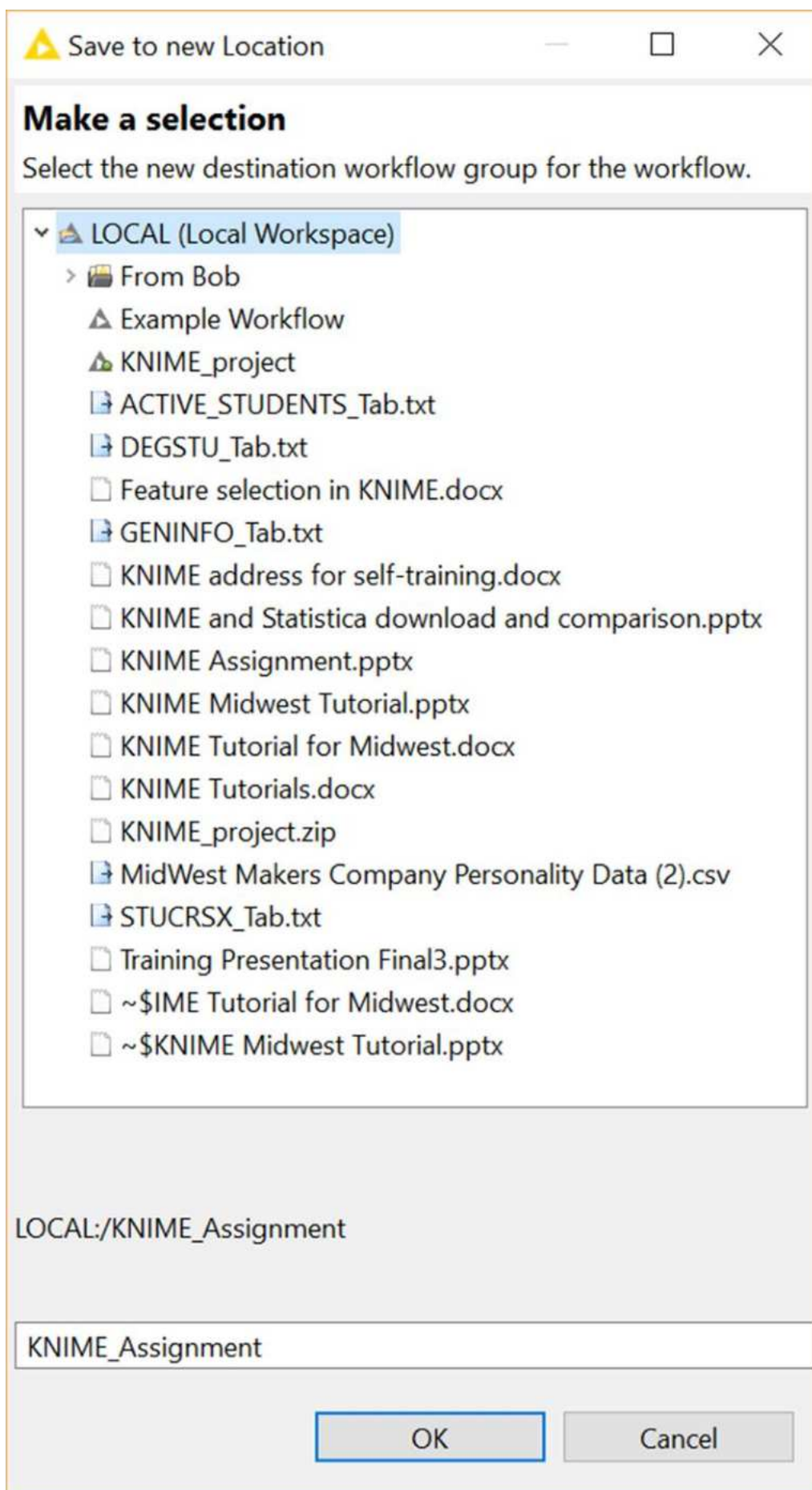


FIG. D.6 Workflow groups under the LOCAL home group.

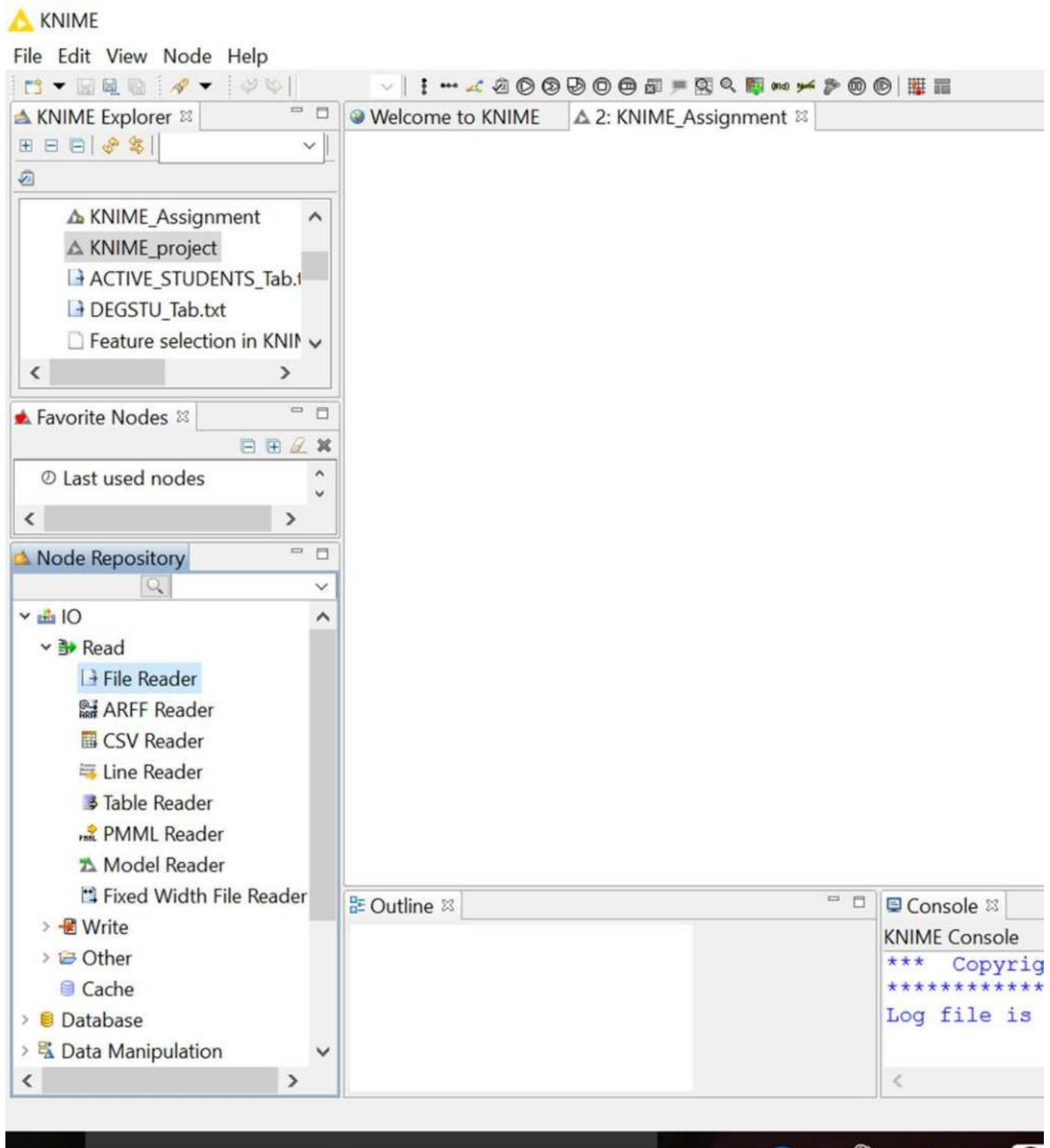


FIG. D.7 Finding the file reader. The file reader module will appear in the workflow.

19. To create a better separation in the colors of the Histogram bars:
 - a. Double-click on the Color Manager node.
 - b. Click on the “1 - 5 years” category, then click on the yellow box (last to the right) of Set 1.
 - c. Click on the “Less than 1 year” category, then click on the tan box (seventh to the right) of Set 1.
 - d. We will leave the other colors the same, as shown in Fig. D.18.

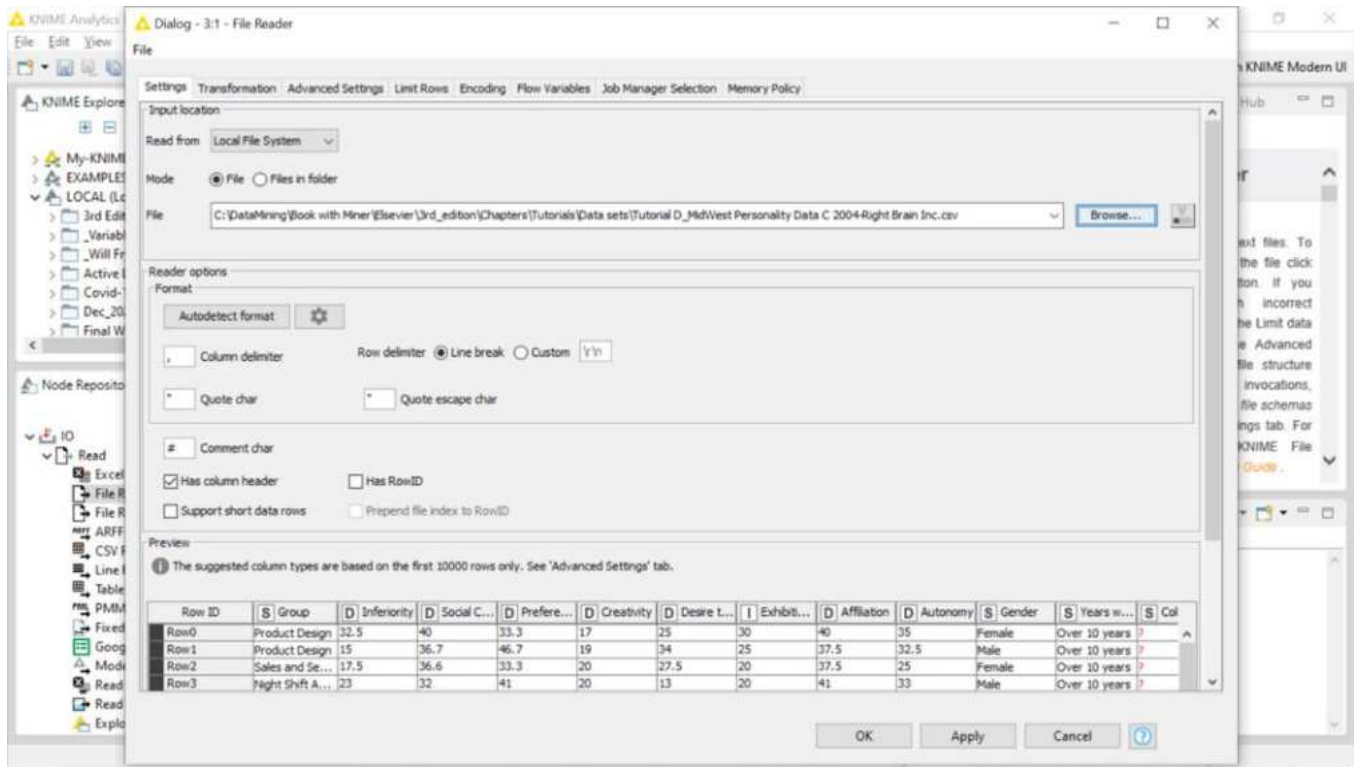


FIG. D.8 The dialog box for configuring the file reader.

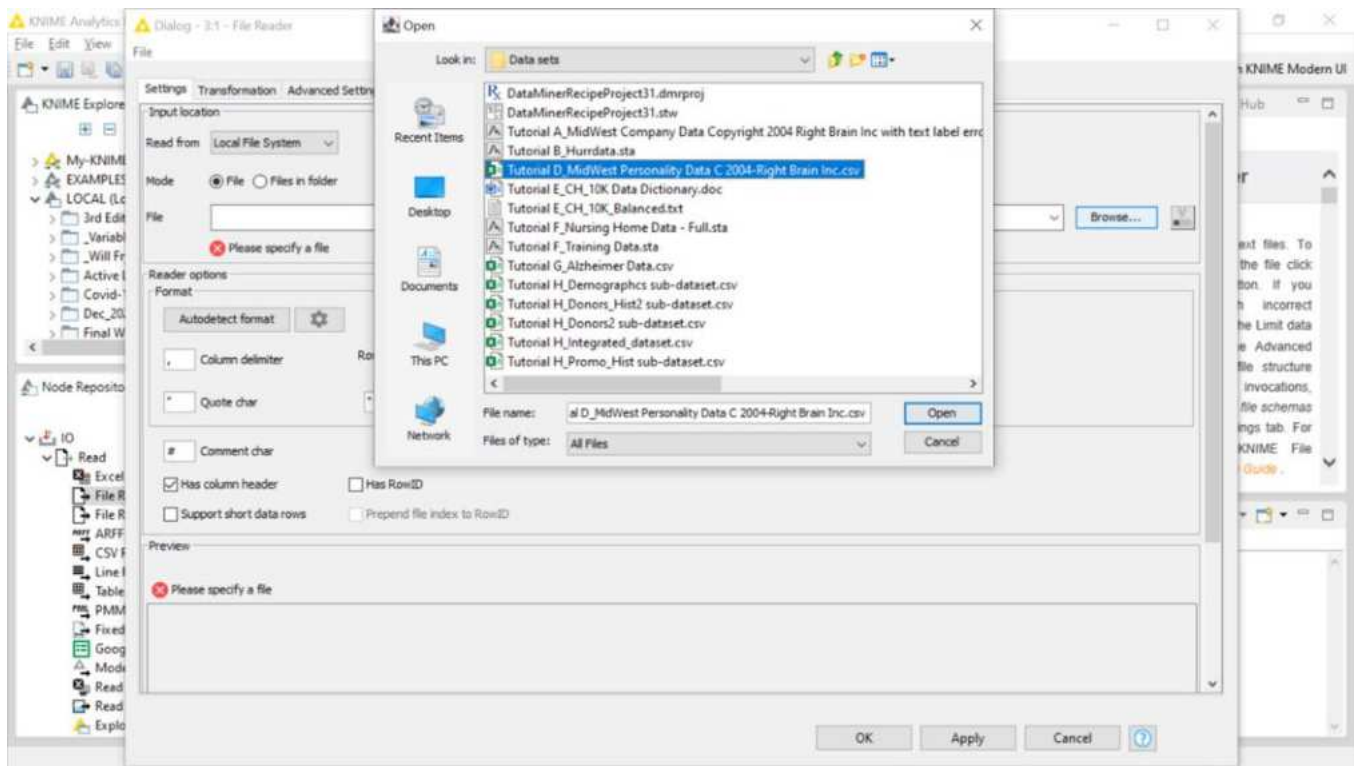


FIG. D.9 Navigating to the MidWest Data in the Data Sets folder.

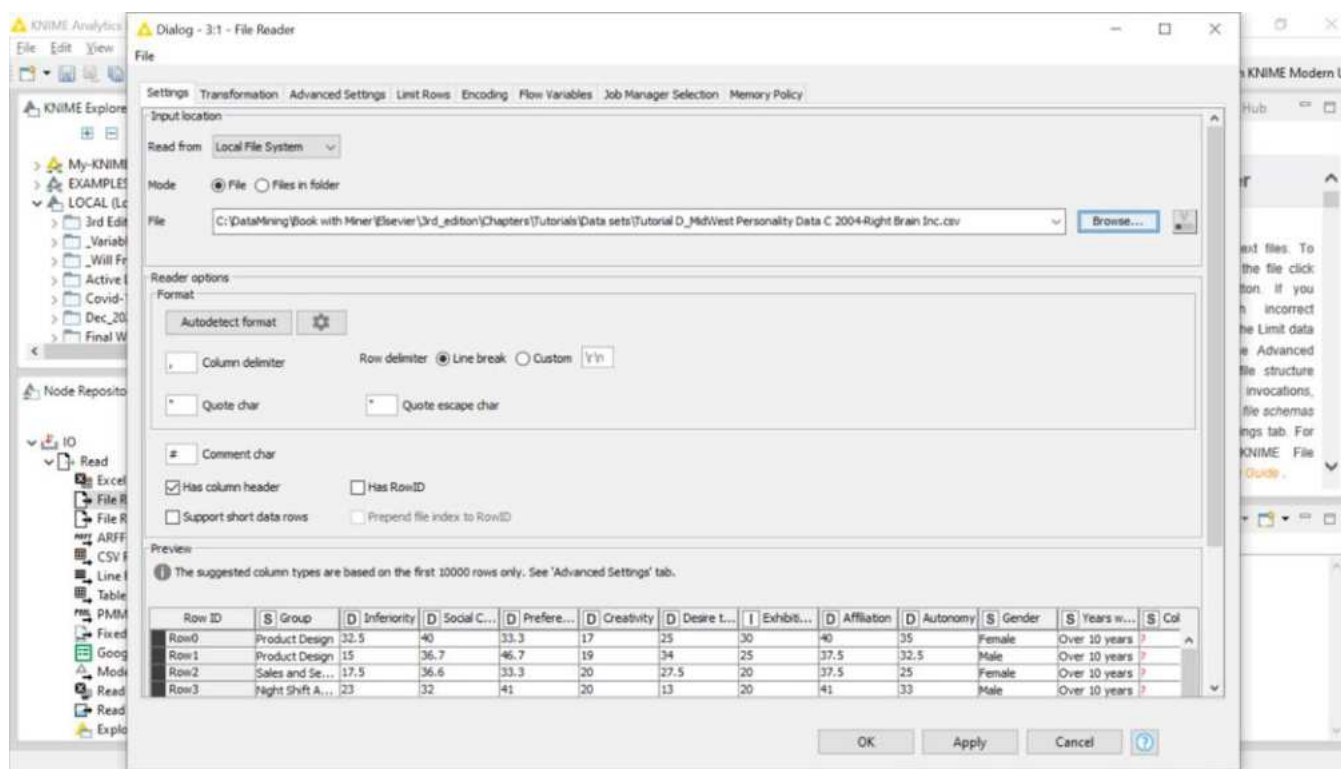


FIG. D.10 Preview of the MidWest data set; cases are listed with a Row ID in the first column, and the variable names are listed in columns of the first row.

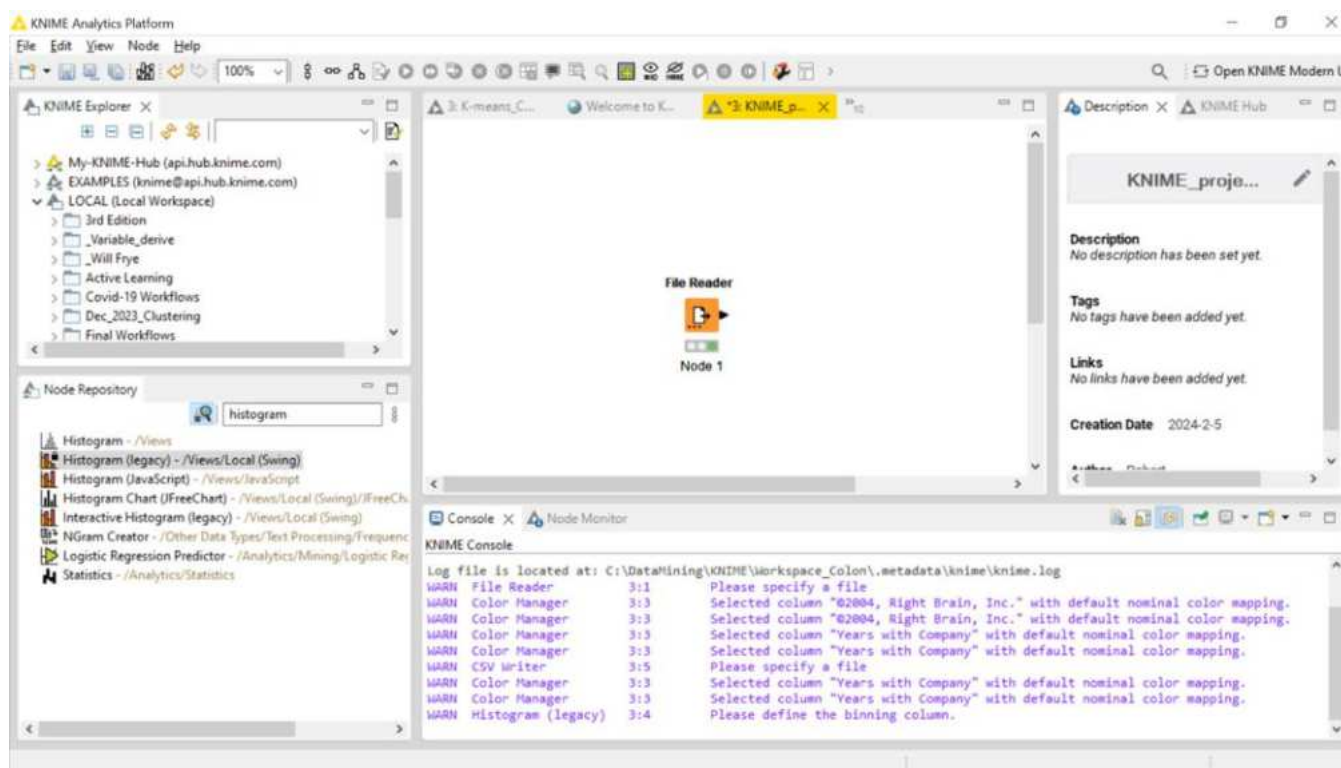


FIG. D.11 Finding the Histogram (Legacy) node using the fuzzy search string.

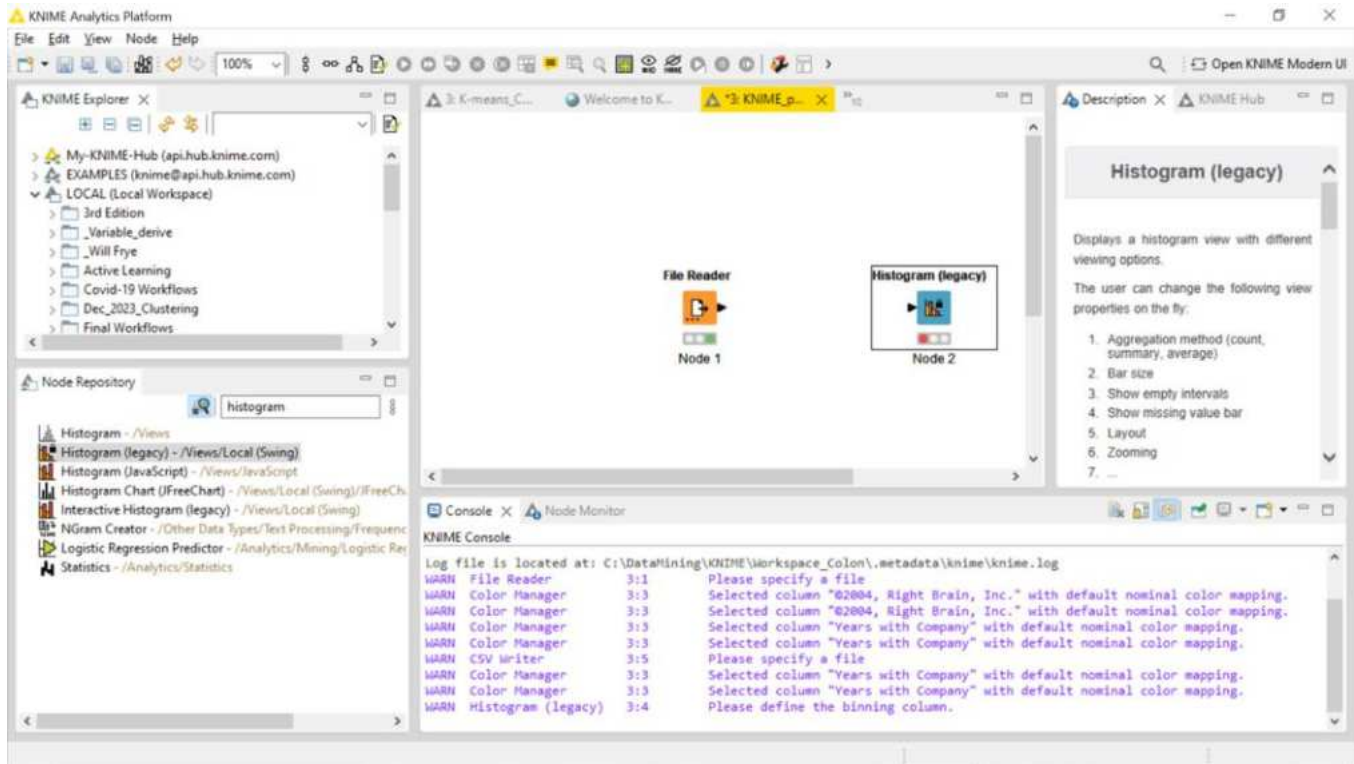


FIG. D.12 Showing the Histogram (Legacy) node next to the File Reader node.

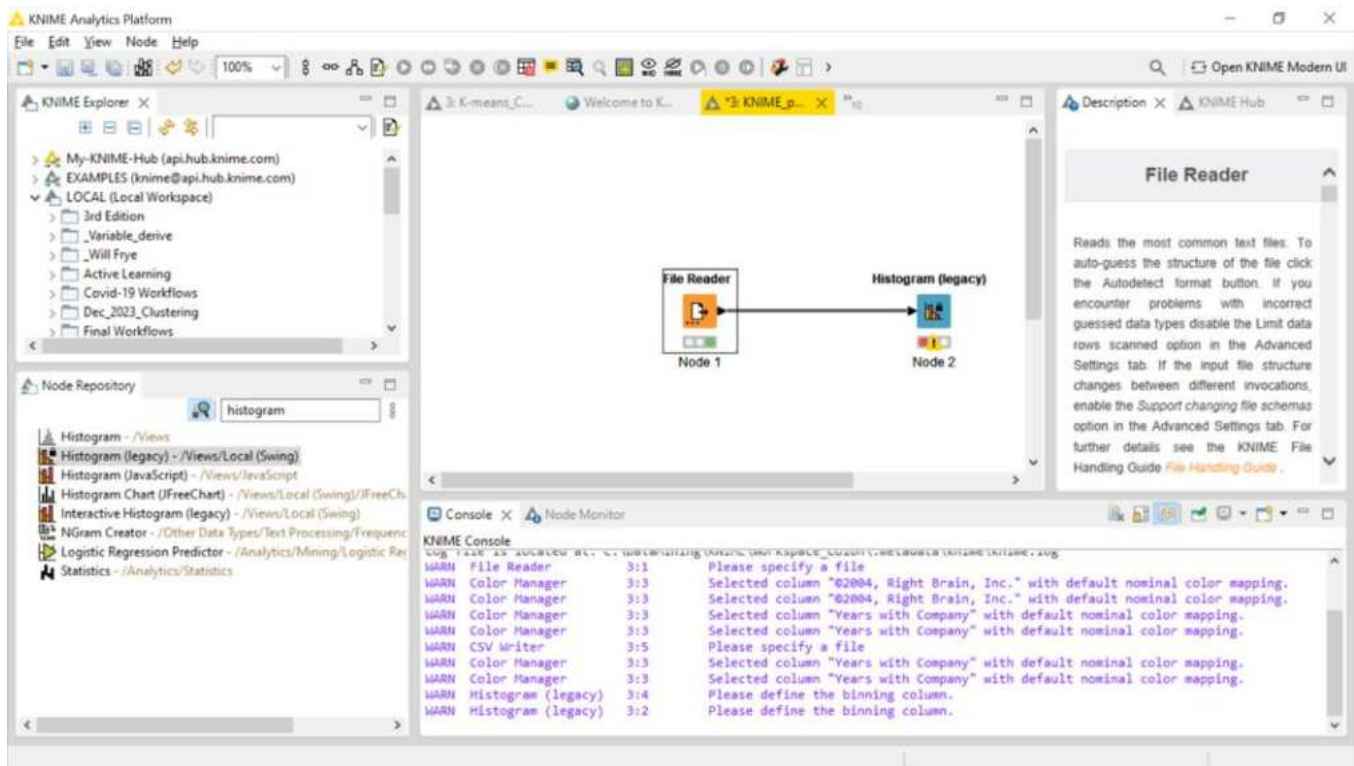


FIG. D.13 Dragging a connection from the File Reader node to the Histogram node.

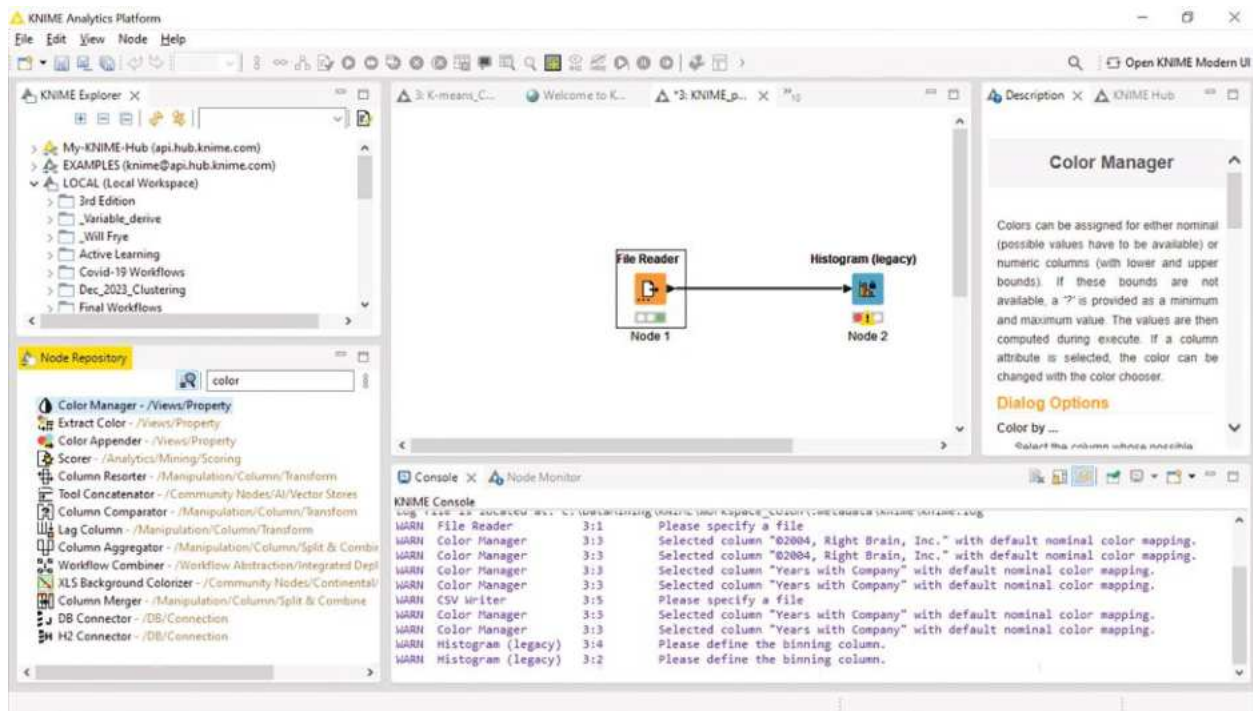


FIG. D.14 Using the fuzzy matching feature of the Node Repository to find the Color Manager.

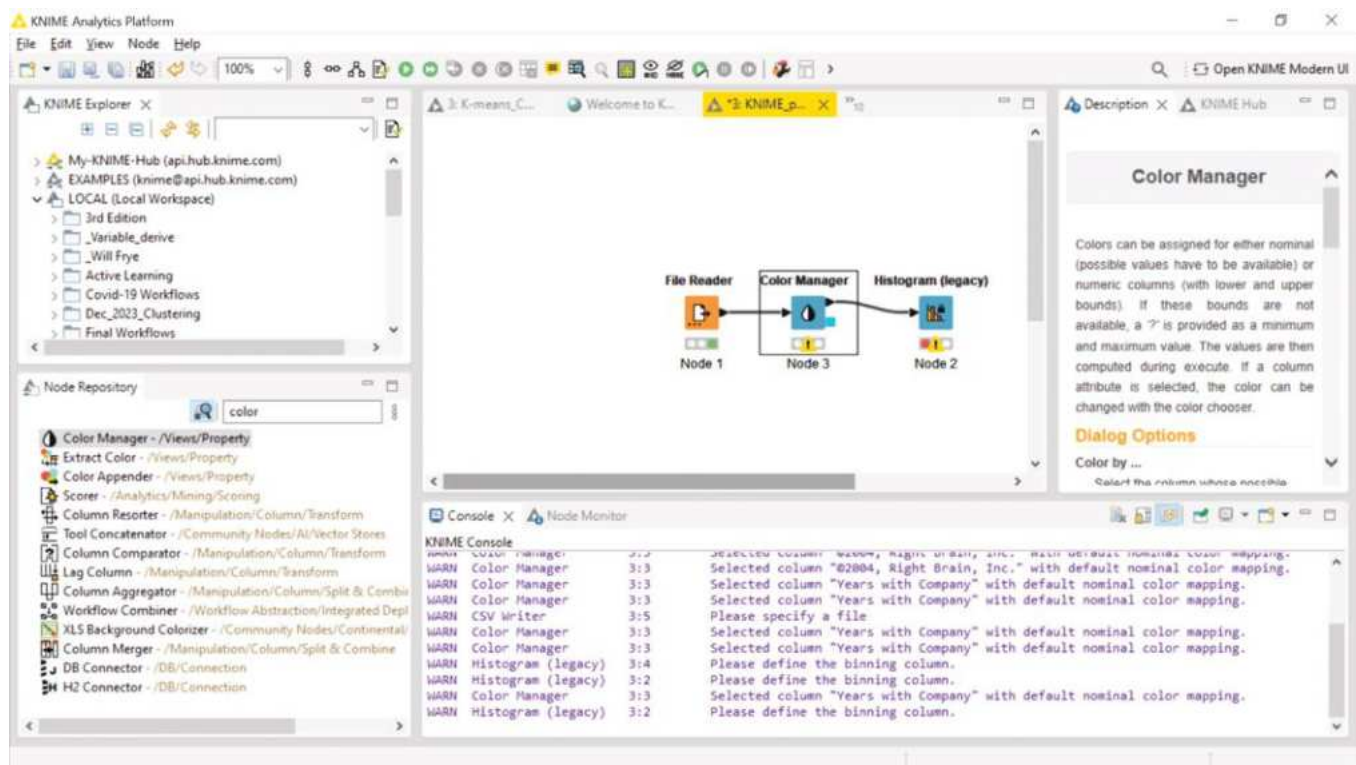


FIG. D.15 Inserting the Color Manager node between the File Reader node and the Histogram node.

- e. Click OK to return to the workflow screen.
- f. Right-click on the Color Management node and select the Execute option, as shown in Fig. D.19.
20. Right-click on the Histogram node and select Execute.
21. Right-click on the Histogram node and select the View: Histogram View.

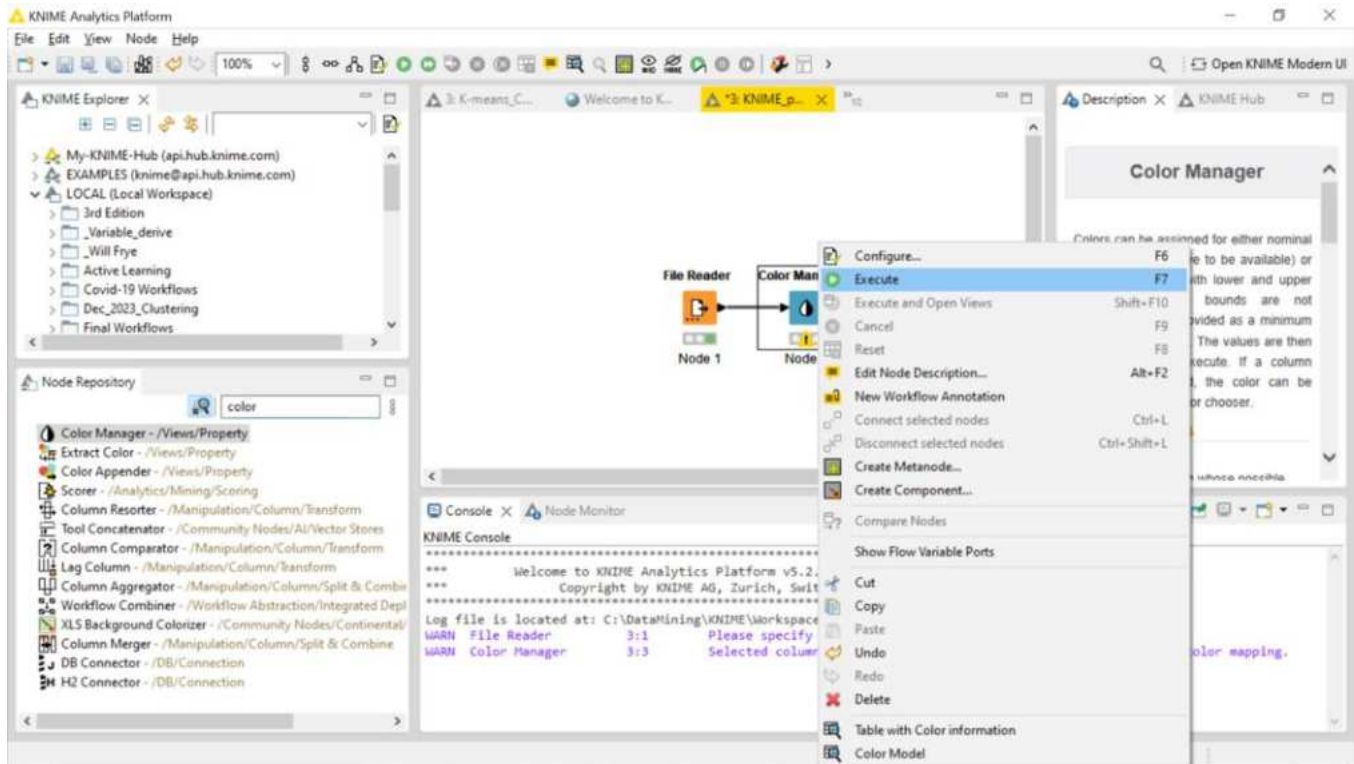


FIG. D.16 Right-clicking on the Color Manager node to execute the Color Manager.

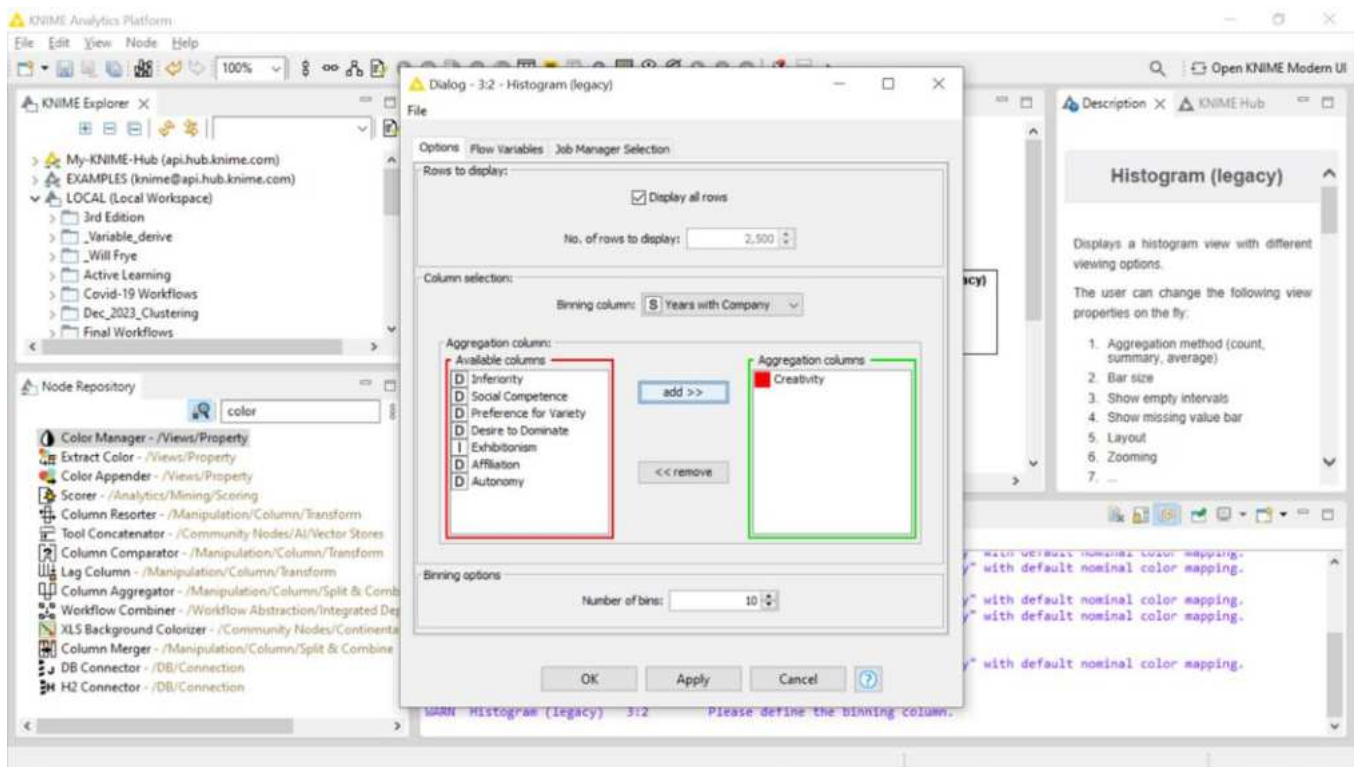


FIG. D.17 Selecting the Creativity variable to graph with the histogram.

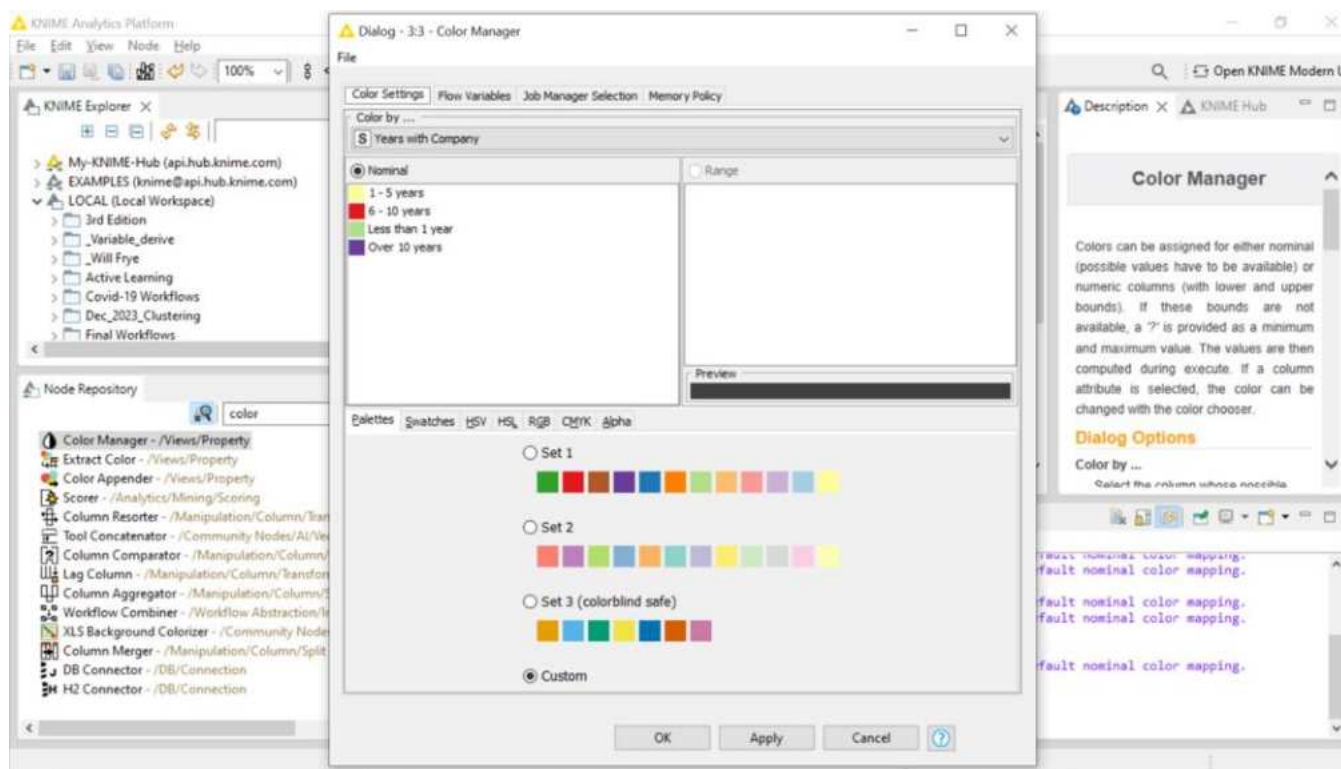


FIG. D.18 Color selection in the Color Manager node.

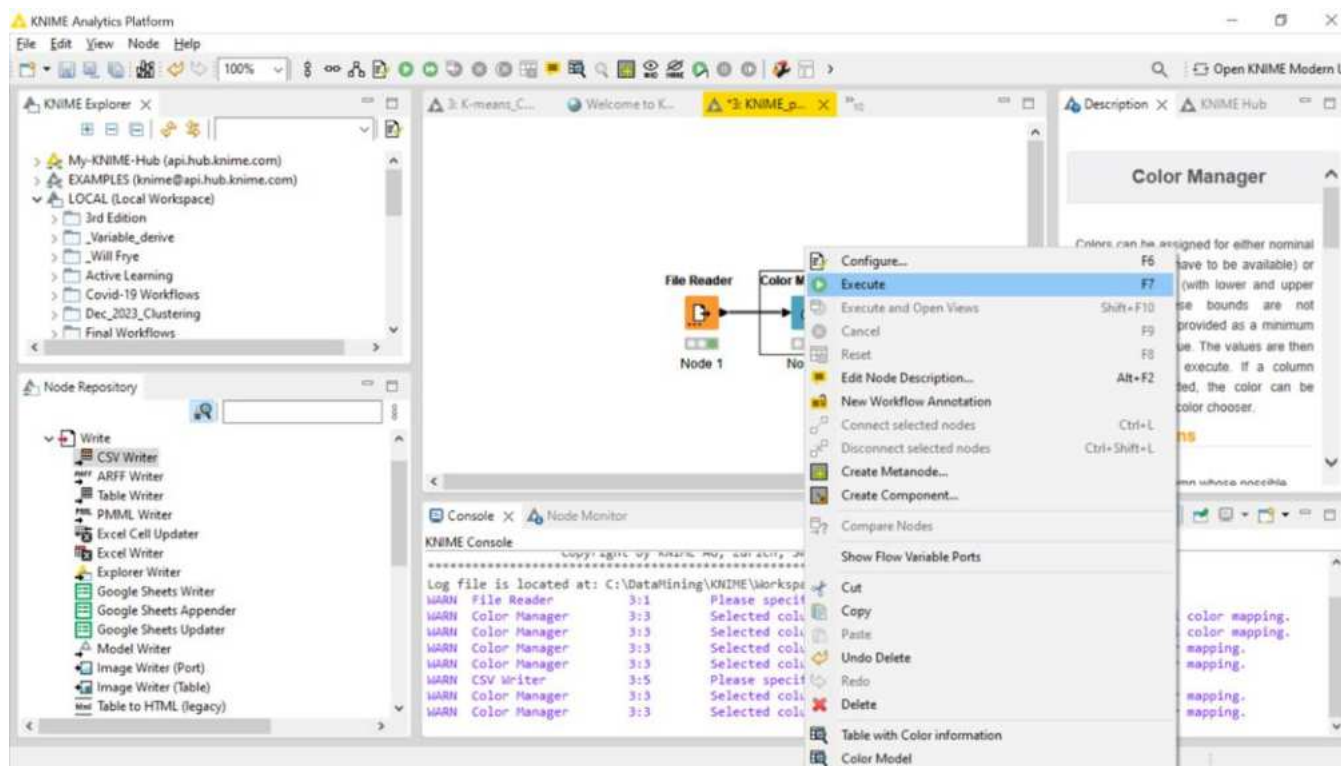


FIG. D.19 Right-click on the color management node, then click Execute.

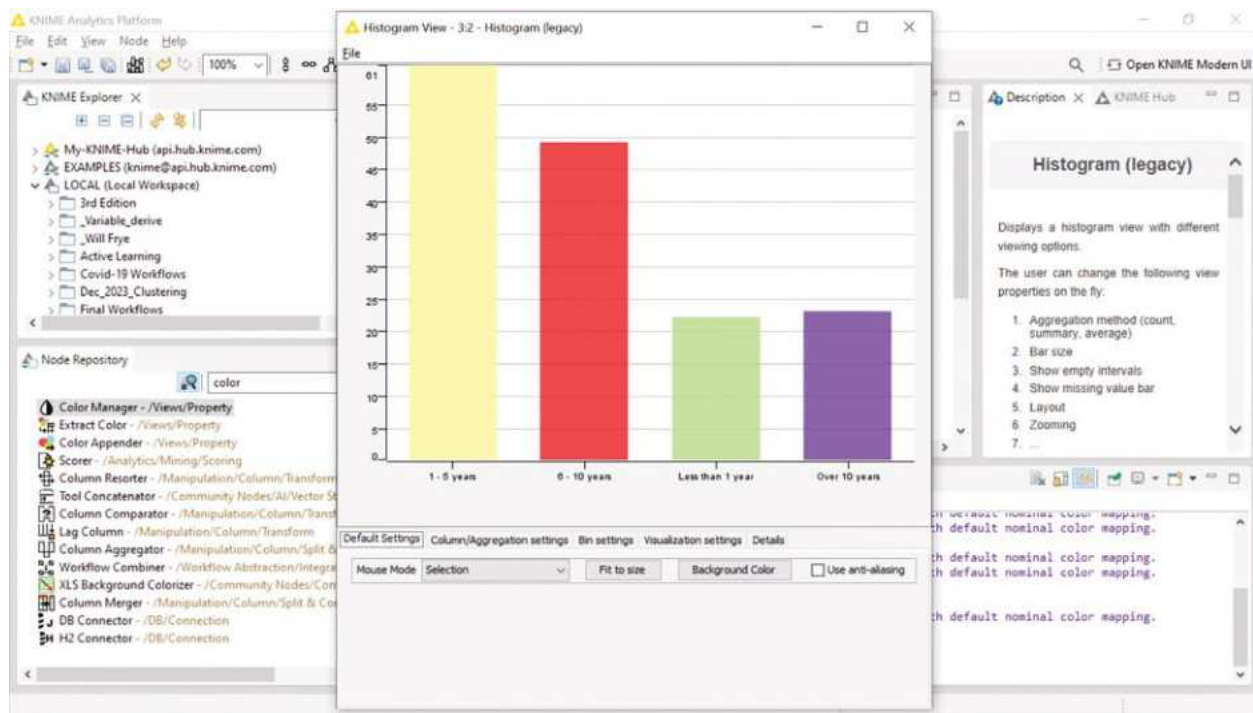


FIG. D.20 Results from the configured Histogram node.

22. Fig. D.20 shows the resultant Histogram with its pretty colors! Note that the Histogram shows that aggregate creativity peaks during the period of 1–5 years with the company and declines afterward.

Advice to the reader in working with KNIME: Save often to make sure you don't have to start over if the program crashes.

E

Feature selection using KNIME

Robert Nisbet

This tutorial will introduce you to one of the most valuable operations in predictive analytics — feature selection. The term “feature” is used often as a synonym for “variable.” Strictly speaking though, there is a distinction between the terms. Variable refers often to the raw value or code in a given column of a data set. On the other hand, Feature refers to transformed numbers of codes to “map” these raw values into “feature space,” according to some heuristic. The multidimensional decision surface of this feature space is much more expressive than with the raw data values.

For a background on feature selection in general, read Chapter 5 in the book. A brief review of the reason for feature selection is presented below.

Why select features?

Many modeling algorithms will select the important features in the model automatically; therefore why select features ahead of modeling? The reason is that algorithms work much better with fewer variables to wade through. Variables that are unimportant in defining the solution contribute only “noise” in the “signal” sought in the data set. The outflow of this principle has been described in many forms:

1. KISS (keep it simple, stupid)
2. Keeping matters “Short and sweet”

Occam’s razor – simple, but not simplistic

The articulation of this principle (named subsequently as “Occam’s razor”) was popularized by William of Occam, a 14th-century English clergyman and philosopher. He wasn’t the first philosopher who posited the principle, which can be traced back to Aristotle, but he popularized it in the Latin form:

Latin: “Entia non sunt multiplicanda praeter necessitatem”

English literal translation: Entities are not to be multiplied beyond necessity.

English general meaning: Don’t complicate the description of something beyond what is necessary to explain it.

Often, the modeling process follows this principle in that many modeling algorithms perform worse and worse as the number of predictors increases. This effect has been called the “Curse of Dimensionality.”

Analytical “dimensions” are represented by the variables in a model. The modeling algorithm “views” these variables as dimensions in a multidimensional decision surface. The algorithm must “traverse” this decision surface to “find” the point of lowest prediction error. But sometimes, the search gets “trapped” in a region on the decision surface of local but not global minimum error. The larger the dimensionality (number of predictor variables), the higher the probability that the error minimization processing of the algorithm will get trapped in a region of local minimum error.

Local minimum error

A decision surface is like a topographic map in three dimensions, except that it has many more dimensions.

If our decision surface had only two dimensions, it might look like Fig. E.1. The blue line traces the initial path of the minimum error search algorithm. After finding the candidate location of minimum error (which happens to be local minima), the algorithm must have some sort of perturbation check to “bounce” the search path over the local maximum error “hill” and continue its search for the global minimum error location on the surface. The downward-pointing arrow refers to a decreasing total prediction error.

Moving out of the local minimum

Different algorithms use different strategies to perturbate the search path to climb over small hills in the error

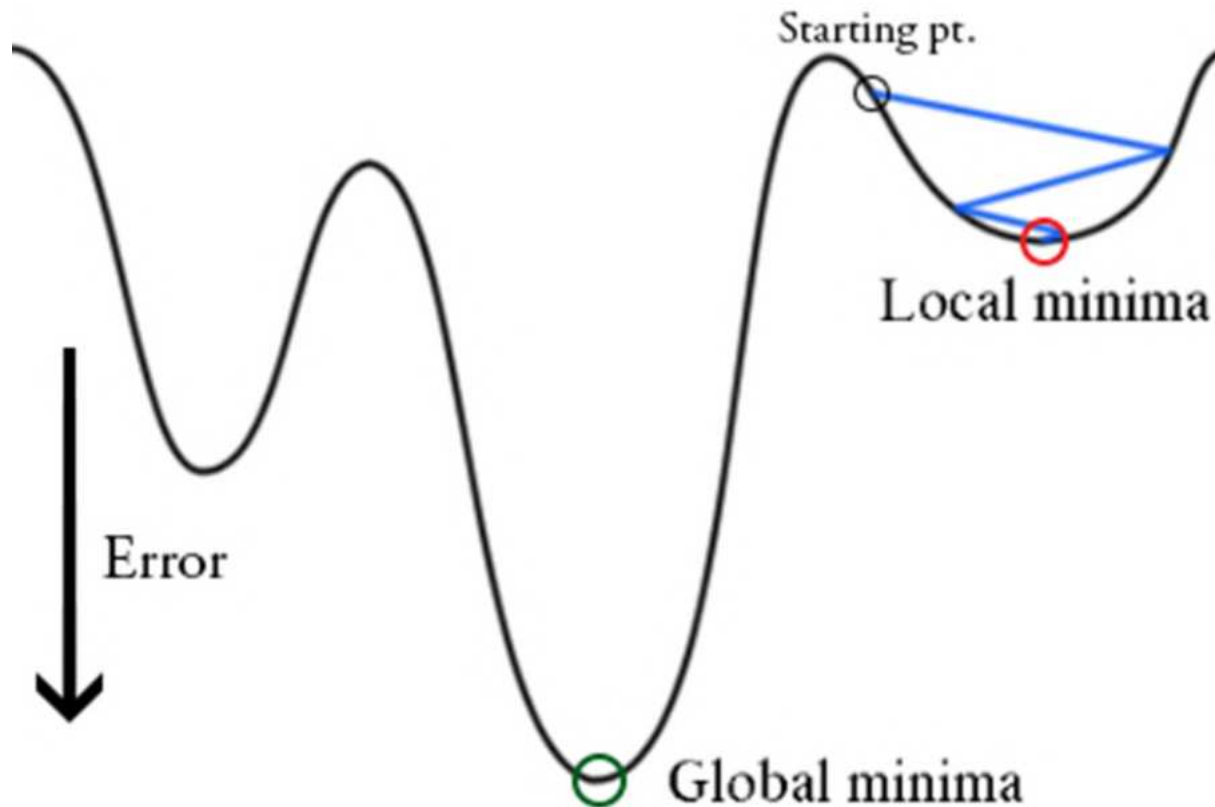


FIG. E.1 The relationship between local minimum error and global minimum error on a two-dimensional decision surface.

surface. Whatever strategy is used, the algorithm will work *much* better with fewer variables (dimensions) than with a greater number of variables. The greater the number of variables in a decision surface, the higher the probability that the search for the minimum error on a decision surface will become trapped in a local minimum error location, rather than finding the global minimum error location on the surface. This is one aspect of the “Curse of Dimensionality.”

For this reason, a primary strategy followed in data preparation operations of predictive analytics is to minimize the effects of the “Curse of Dimensionality” by reducing the number of variables submitted to the modeling algorithm.

Strategies for reduction of dimensionality in predictive analytics available in KNIME

There are a number of strategies used in KNIME for feature selection.

Deleting inappropriate variables. We will use this approach in the KNIME Column Filter node to delete inappropriate

variables: (1) the CUST_ID variable (identifiers, which cannot be used as predictor variables), and; (2) all date variables (dates cannot be used as predictor variables).

Low variance filtering. This approach uses the KNIME Low Variance Filter node, which screens out variables with less than a specified amount of variance. We will not use this approach in this tutorial.

Correlation coefficient filtering. This approach uses the KNIME Correlation Filter node, which screens out one member of variable pairs with a higher than a specified correlation coefficient. We will not use this approach in this tutorial.

Feature selection. This approach uses a constructed metanode that performs a forward feature selection operation contained in the workflow displayed in Fig. E.2.

We will use this very powerful KNIME metanode for Feature Selection. A metanode is a collection of nodes expressed in a workflow display by a single node icon. In Fig. E.2, the metanode is labeled “Feature Elimination,” located on the upper right of the display. The KNIME Feature Elimination metanode is so named because it selects features by eliminating the least important

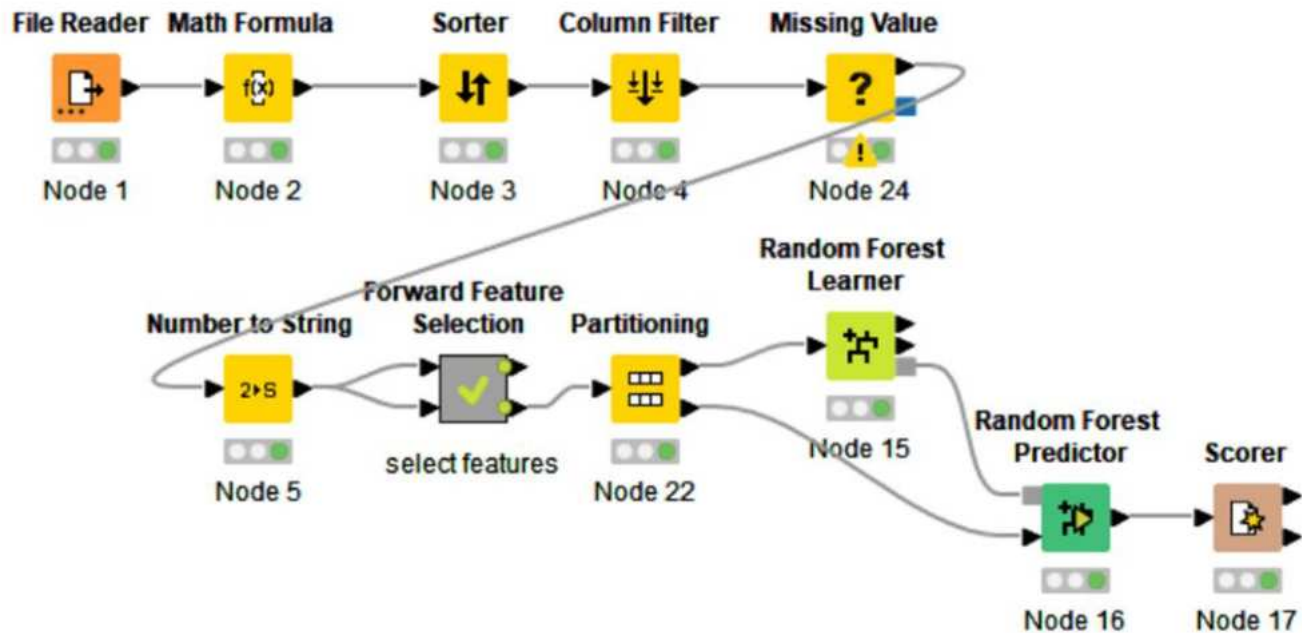


FIG. E.2 The Feature Selection workflow in KNIME.

predictors with a simple model of your choice, as specified within the metanode configuration. This function is similar to the way that Forward Step-Wise Linear Regression selects features with which to build the regression model.

Now we will build the workflow shown in Fig. E.2. Save your work often. Sometimes, KNIME will crash if you do some operation that the system does not expect, and you will lose all your work since the last time you saved the file.

1. Download the CH-10K_Balanced.txt file and the CH_10K_Data Dictionary.docx files from the book Web page.
2. Familiarize yourself with the meaning of the fields described in the data dictionary.
3. All node configuration directions listed below refer to the nodes shown in Fig. E.2.
4. Open KNIME to see the empty workflow screen shown in Fig. E.3.
 - a. Notice the locations of the top menu with the File tab in it, the Modeling workflow canvas, and the Node Repository.
 - b. Click the File tab in the top menu and select the <New> option highlighted in blue in the drop-down menu, as shown by the top right red arrow.
 - c. Notice the Node Repository, which we will use in the next operation.
 - d. Note: In the node configuration directions for this tutorial, strings or options to be chosen will be enclosed in caret symbols (< >) in this tutorial for

descriptive purposes only. Do NOT enter the caret symbols in the menus.

- e. Click on the <New KNIME Workflow> in the following screen, shown in Fig. E.4, for selecting an editor wizard.
5. In the KNIME Node Repository pane, enter "File Reader" into the search box, then double-click on the File Reader node in the list. This is the quickest way to find a node if you know one or more words in its name. KNIME does a fuzzy match on the string entered and displays a list of all nodes that are close matches to the string. You can hunt for the node in the hierarchy of the Node Repository, but you don't have to do that.
 - a. Double-click on the File Reader node to enter the node configuration screen (Fig. E.5). Also, you can drag-and-drop the node onto the workflow.
 - b. Click on the <Browse> box on the right of the display and navigate through the file list to the location where you saved the CH-10K_Balanced.txt file to load into the node.
 - c. Click on the Autodetect format box to detect the column delimiter (tab).

If the file loads properly, CUST_ID should be the first variable in the header (to the right of the ROW_ID, which is just the sequential row number).
 - d. After the file is loaded properly, right-click on the node and select <Execute>. This operation will

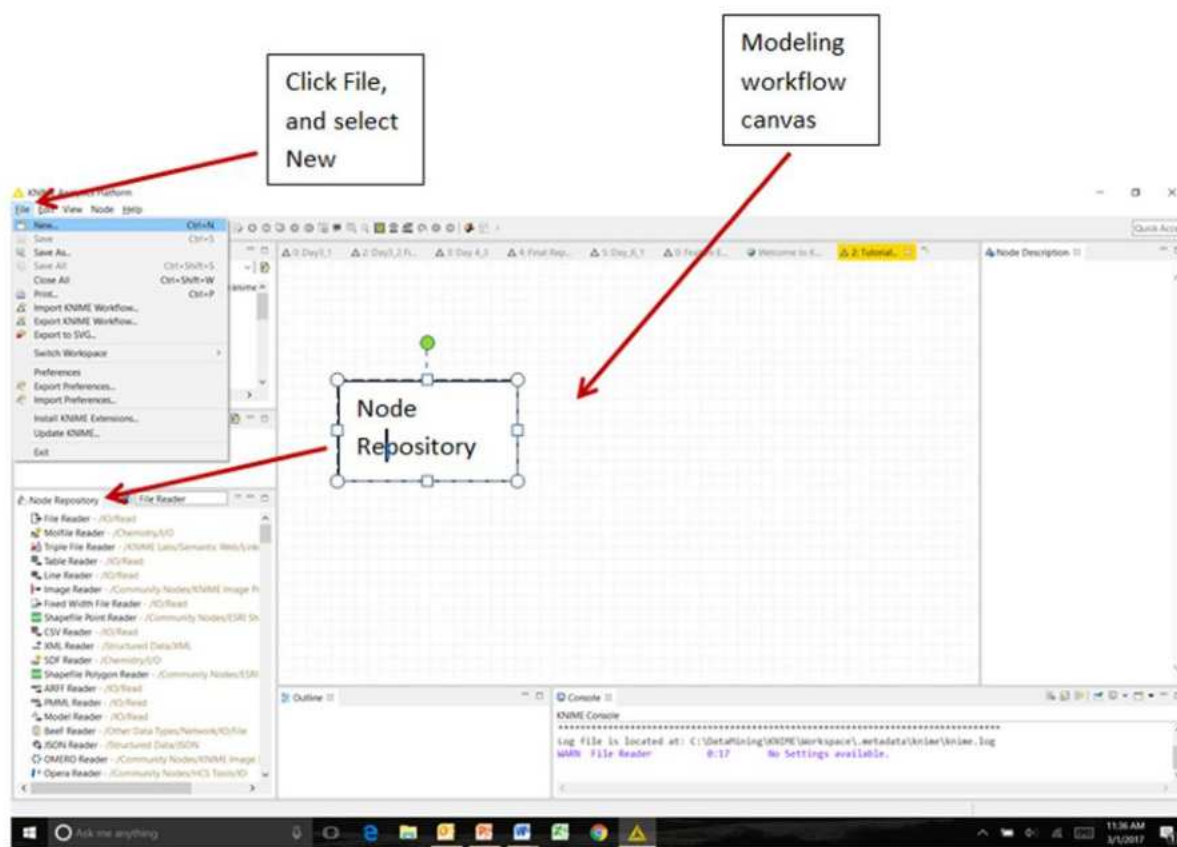


FIG. E.3 An empty workflow editor screen, showing labels of some of the major windows of the user interface.

- load the file into the node. The file shown before executing the node is just a preview of the file before it is loaded by the Execute Operation.
- e. Click OK to exit.
- f. To view the loaded data file, right-click on the File Reader node and select the <File Table> option.
6. Click on the File Reader to highlight it and connect a Math Formula node to it.
 - a. Find the Math Formula node in the Node Repository (according to the directions listed previously in step 4), and double-click on it. The new node will be connected automatically to the highlighted File Reader node on the workflow.
 - b. Double-click on the Math Formula node to configure it, as shown in Fig. E.6.
 - i. Enter the string <rand()> into the Expression box.
 - ii. Click the Append Column button and name the column <RANDOM>.
 - iii. Click OK and execute the node.

This operation will derive a new column, RANDOM, and enter a random number in each row that ranges from 0 to 1.
7. Highlight the Math Formula node and connect to it a Sorter node (after finding the node in the Node Repository according to the directions in step 4). Configure it as follows:
 - a. Select the RANDOM variable in the Sort by box (at the bottom of the variable list).
 - b. Your configured Sorter node should look like Fig. E.7.
 - c. Click OK and execute the node.

This operation will sort the records in the file on the RANDOM variable in ascending order (default). This must be done to ensure that the records are in random order before performing the row selection operation in the Partitioning node, used later in the workflow.
 - d. Right-click on the executed node to show the options screen (Fig. E.8) and select the

Select a wizard

This wizard creates a new KNIME workflow project.

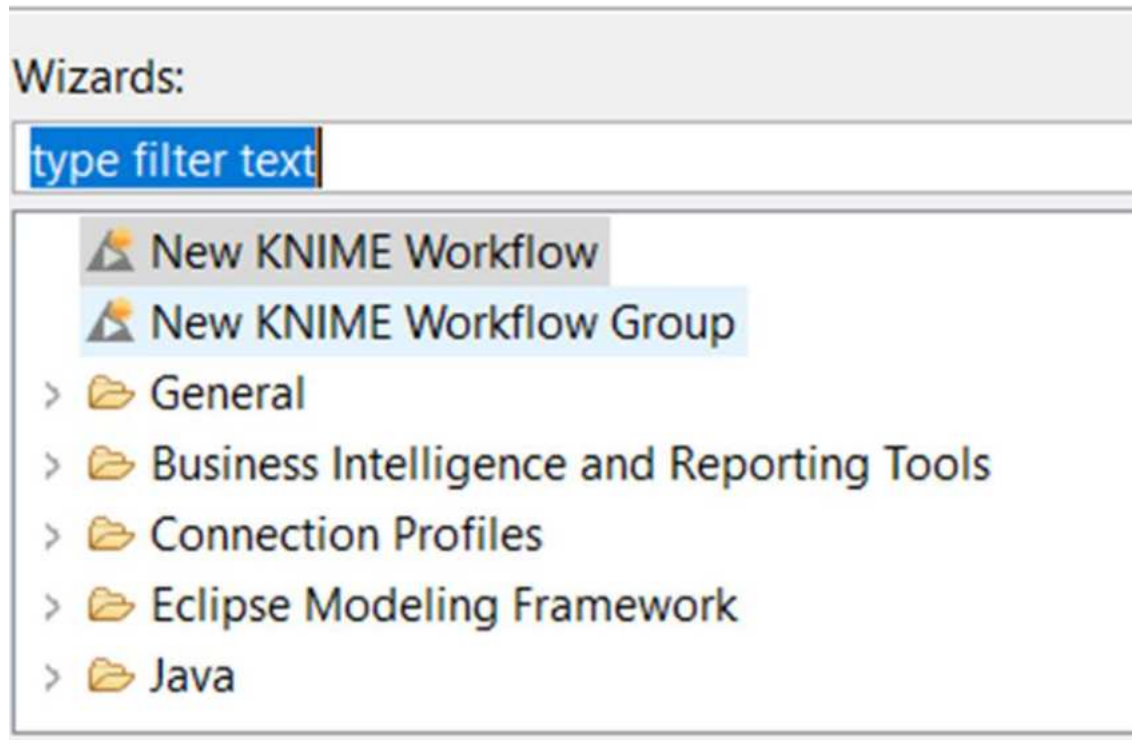


FIG. E.4 Wizard selection screen for building new KNIME elements.

- <Sorted Table> option to display the sorted data set.
- 8. Now we will delete some inappropriate variables for modeling.
 - a. Connect a Column Filter node to the Sorter node.
 - b. Right-click on the node to configure in the dialog screen shown in Fig. E.9 to include all variables that were included initially in the right-hand list, EXCEPT:
 - i. The CUST_ID variable (highlight the CUST_ID variable in the right-hand box and press the < symbol to remove it)
 - ii. Remove all variables that have _DT, _BEG, _END, or CLOSED in them (which have date strings in them in the format MM/DD/YYYY).
 - iii. Remove TEN_RAW and RANDOM variables also.
 - iv. When done, click OK and execute the node.
 - v. Right-click on the executed node and look at the <Filtered table> option to see in the output data set that only the right variables were passed. You will notice that the name of the output viewing option is different in each node type.
- 9. There are missing values in a few rows of some variables. We must decide how to handle them: fill them with some default value or delete the rows. In this tutorial, we will delete the rows with missing values. Because there are only a few rows with missing values, deleting rows with missing values is likely to have less of an effect on the model than imputing missing values with some constant or trying to build a submodel to infer the missing values.
 - a. Connect a Missing Values node to the Column Filter node.
 - b. Double-click on the Missing Values node to configure it.

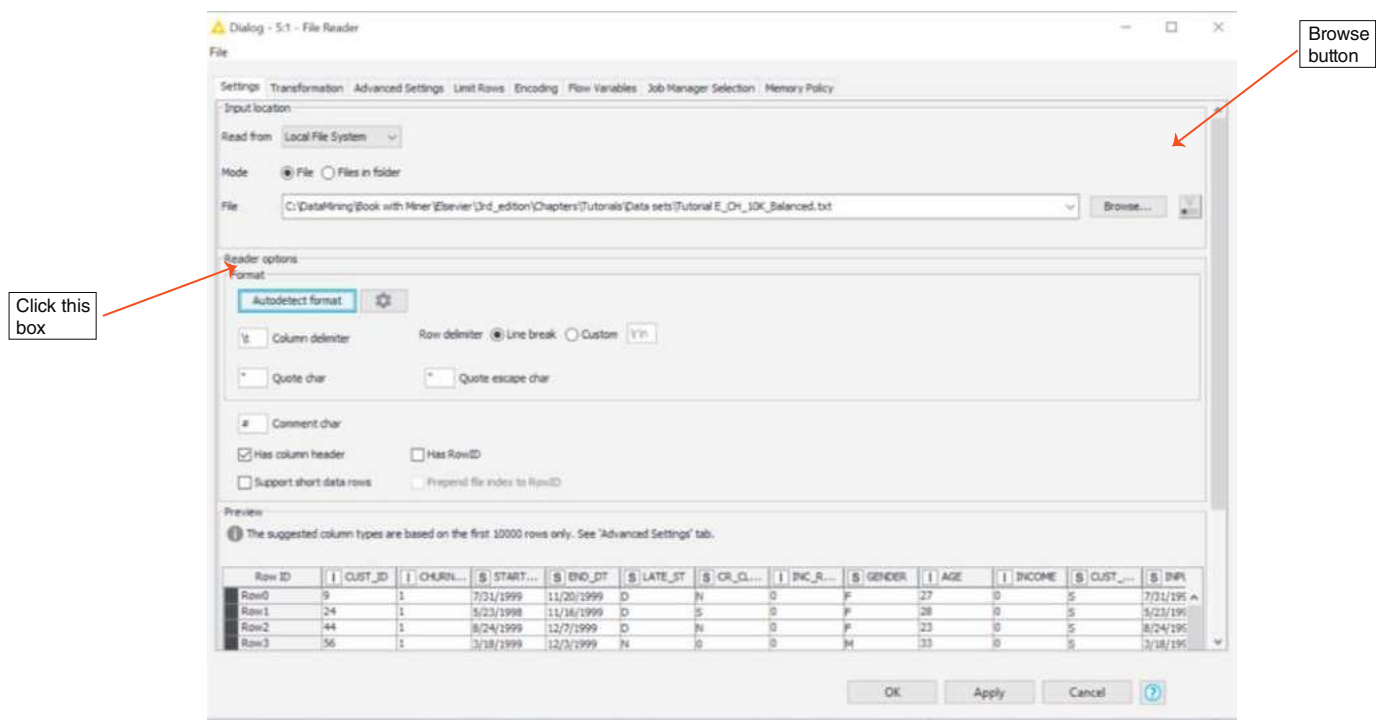


FIG. E.5 The File Reader configuration dialog screen.

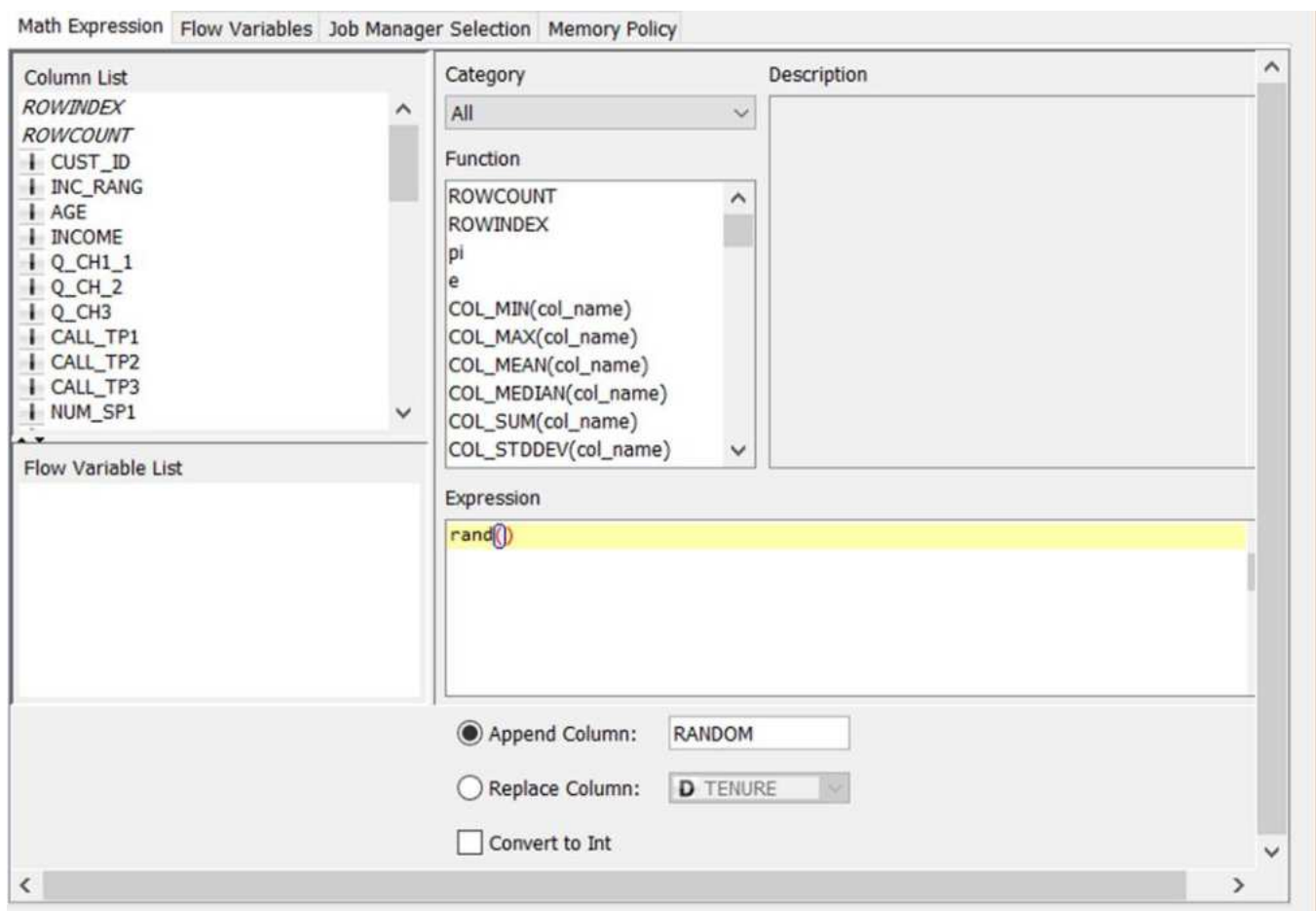


FIG. E.6 The Math Formula node configuration screen.

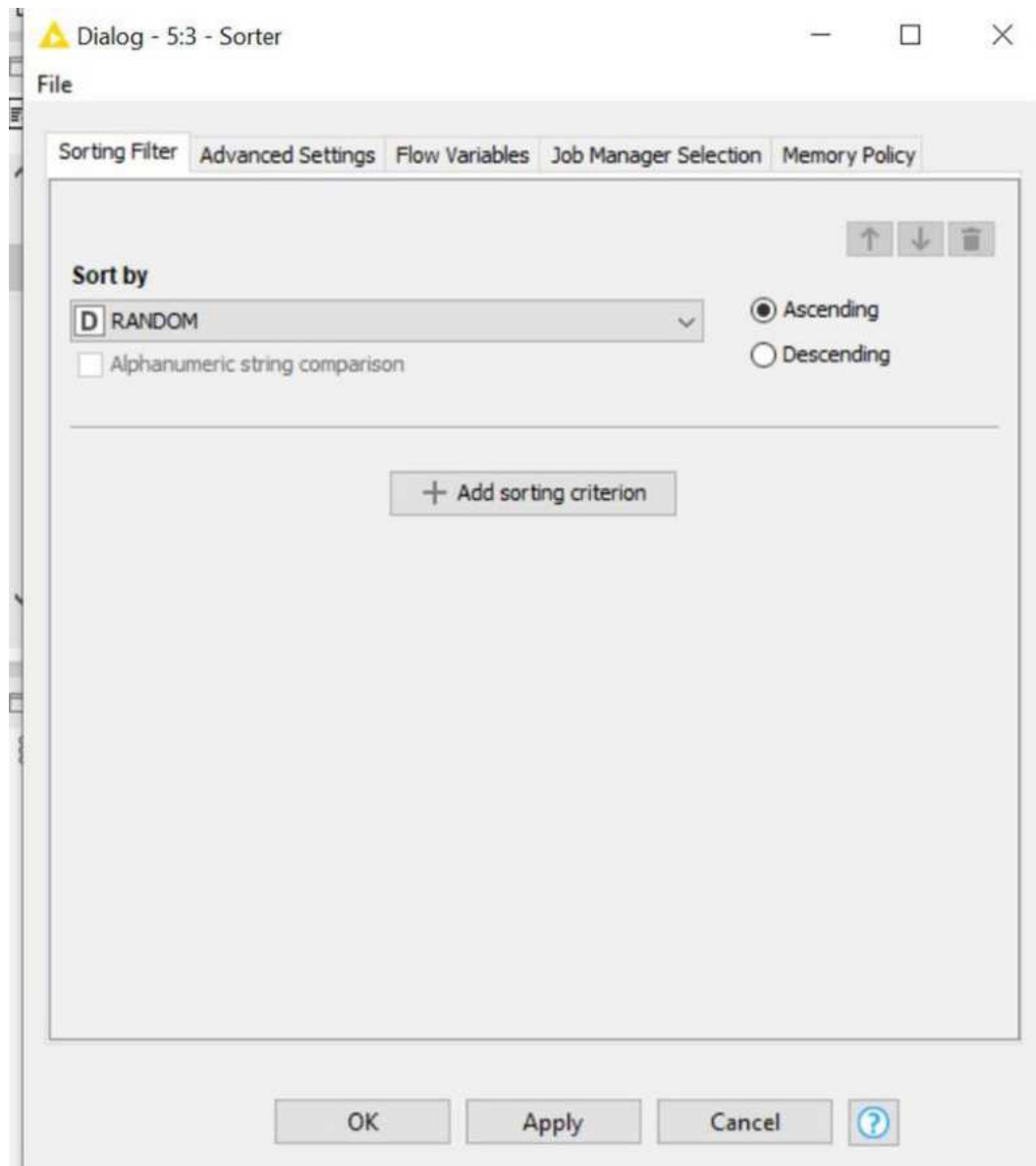


FIG. E.7 The Sorter configuration dialog screen.

- c. Set each of the data types (Integer, String, and Double) by clicking on the down arrow in the box next to them and selecting the <Remove Row*> option. (Fig. E.10).
10. In the next operation, we must change the CHURN_FL variable from an Integer data type to a string. The File Reader node assumed that the numbers of 1 and 0 in CHURN_FL were numeric, not characters of a string. CHURN_FL must be a string for the classification algorithm to use it as a target to predict.
 - a. Connect a Number to String node to the Column Filter node.
 - b. Double-click to configure it to include only the CHURN_FL variable in the right pane, as shown in Fig. E.11.
11. The next operation is where the subset of features are selected, using a KNIME feature selection loop. Looping is one of the most powerful and useful capabilities available for data engineering. Each iteration of the feature selection loop builds a model on a different subset of variables. At the end, the user selects which model to use based on which has the highest accuracy.
 - a. The component nodes of the Forward Feature Selection metanode are shown in Fig. E.12.

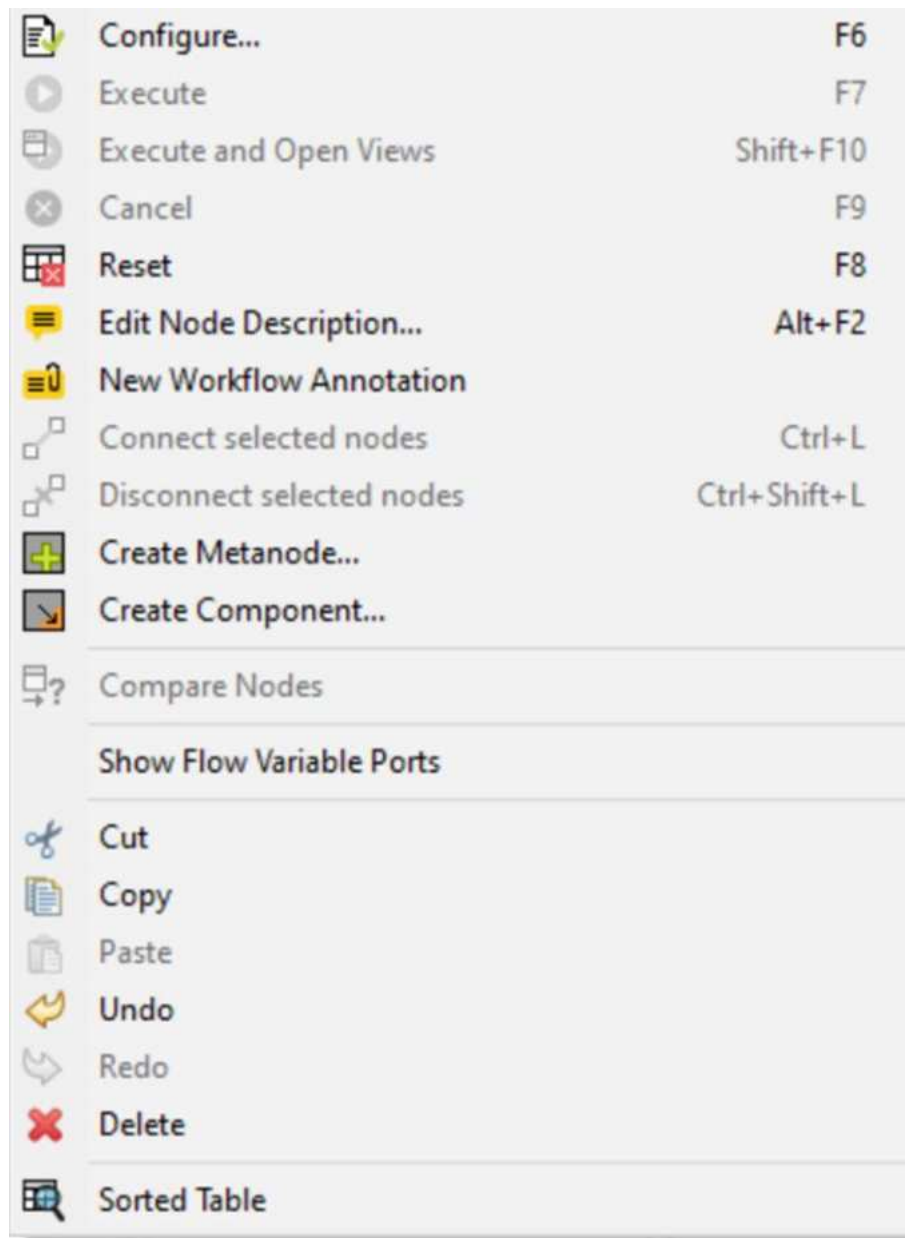


FIG. E.8 The option menu of the executed Sorter node.

- b. Configure the Feature Selection Loop Start node to:
 - i. Exclude the CHURN_FL variable.
 - ii. Use the <Forward Feature Selection> option in the Feature selection strategy box.
 - iii. Leave other default settings.
- c. Next, we must connect a Partitioning node to the Column Filter node. This node will divide the input data set into two pieces: a Training-Testing set and a Validation set. The modeling node in the Feature Elimination metanode will divide the input Training-Testing set into two halves for

Training and Testing operations prior to training the model.

- d. Connect a Partitioning node to the Column Filter node.
- e. Right-click on the Partitioning node and configure it in Fig. E.12 to set the Relative[%] of the first partition to 70 (70%). Configure the Partitioning node to use 70% of the rows for the first partition.
- f. Configure the Partitioning node to do stratified sampling by clicking on the Stratified sampling button and set the variable next to it as

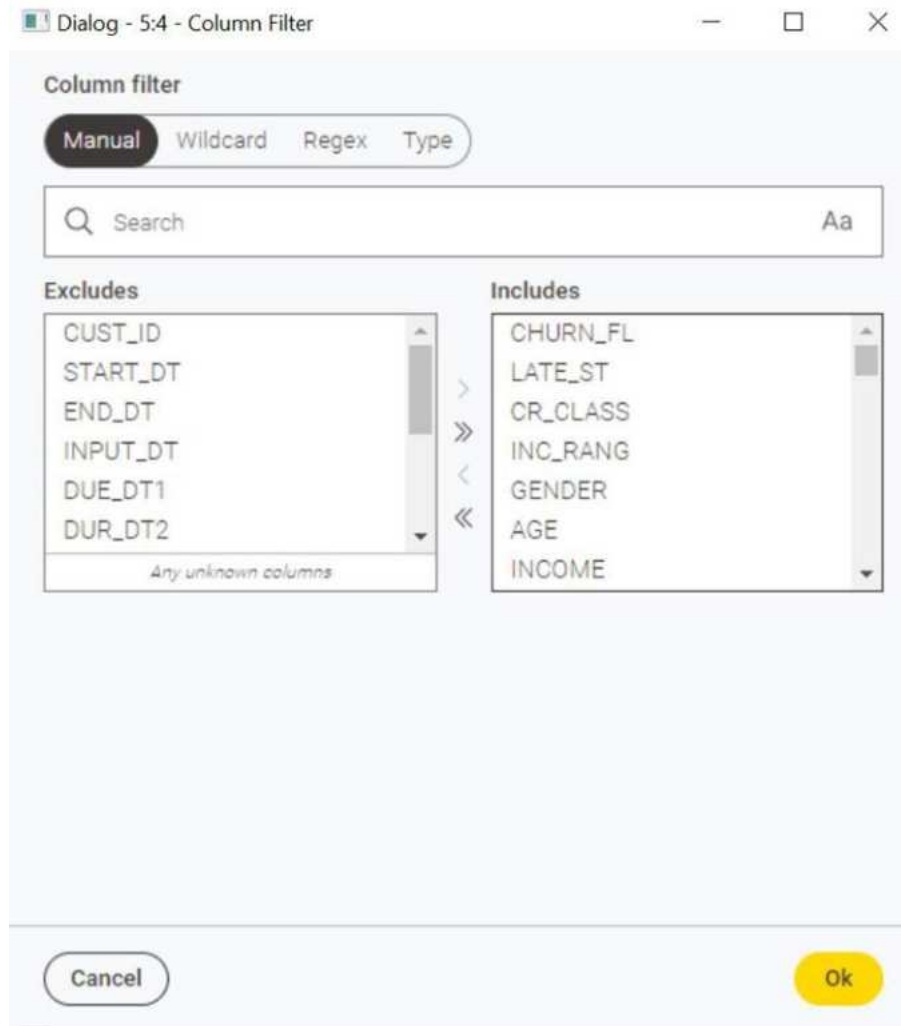


FIG. E.9 The Column Filter node configuration dialog screen.

- <CHURN_FL>. This setting will ensure that each partition has the same relative number of CHURN_FL = 1 rows and CHURN_FL = 0 rows.
- g. The configured Partitioning node is shown in [Fig. E.13](#).
 - h. Click OK to return to the modeling canvas.
 - i. Configure the Naïve Bayes Learner node to use CHURN_FL as the Classification Column (the target variable).
 - j. There is no need to configure the Naïve Bayes Predictor node.
 - k. Configure the First Column box of the Scorer node to use the <CHURN_FL> column and the Second Column box to use the Prediction (CHURN_FL) model output variable ([Fig. E.14](#)).
 - l. Configure the Feature Select Loop End node to use the <Accuracy> option in the Score box to evaluate the models built.
 - m. Ignore the Feature Selection Filter node for now.
 12. Now we are ready to execute the loop of nodes.
 - a. Right-click on the Feature Selection Loop End node and select the <Execute> option.
 - b. When the status light of the Feature Selection Loop End node turns green, we are ready to select a model to use.
 13. Double-click on the Feature Selection Filter node to see the model results shown in [Fig. 15](#).
 - a. [Fig. E.15](#) shows the model accuracy in the left pane that corresponds to the number of variables in the right pane used to build it. The modeler much highlight the model accuracy for the

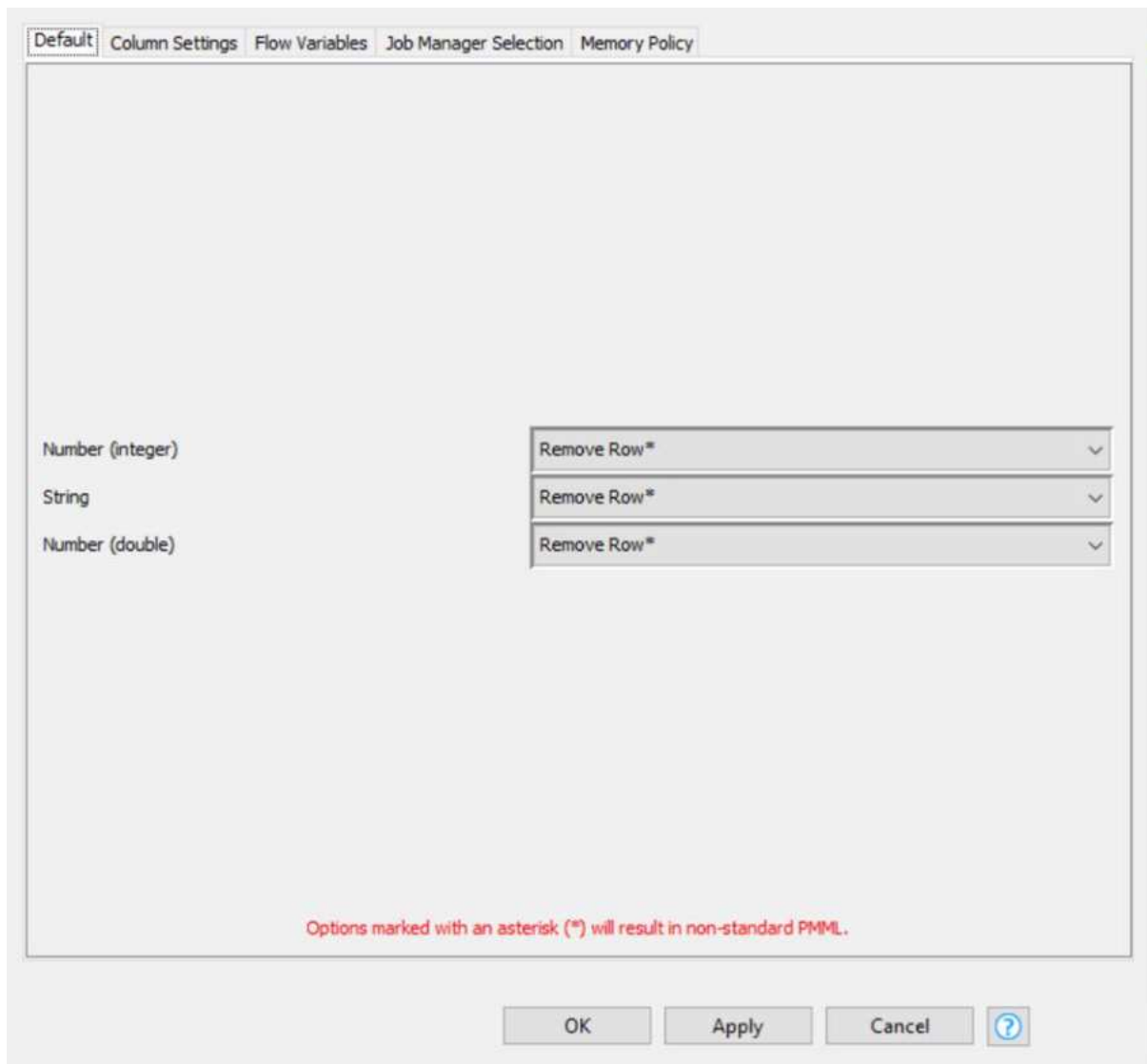


FIG. E.10 Setting the missing value options.

- model that should be used to filter the variables to be used to train the final model.
- b. The goal is to pick the model with the fewest number of predictor variables associated with the model of acceptable accuracy. In this case, the model with 8 predictors has almost the same accuracy as the model with 24 predictors. Following the rule above, the best model is the one with 8 predictors.
- c. Click on the first row of the Accuracy screen to highlight it, as shown in Fig. E.15.
- d. Click OK to return to the metanode diagram.
- e. Right-click on the Feature Selection Filter model and select the <Execute> option.
- f. Right-click on the executed node and select the Filter table report to see that the data set has the number of variables in it for the model selected. Your number of variables may be slightly different than shown in the tutorial because of the randomness of the Partitioning operation in generating the two data subsets for modeling.
- g. Exit the table screen to return to the metanode diagram.
- h. Exit the metanode screen by clicking on the X in the upper right of the screen.
14. Now we can build the final model with the Random Forest algorithm.
 - a. Connect Partitioning node to the lower output port of the metanode and configure it as before.
 - b. Configure the Random Forest Learner to use the CHURN_FL variable as the target column.

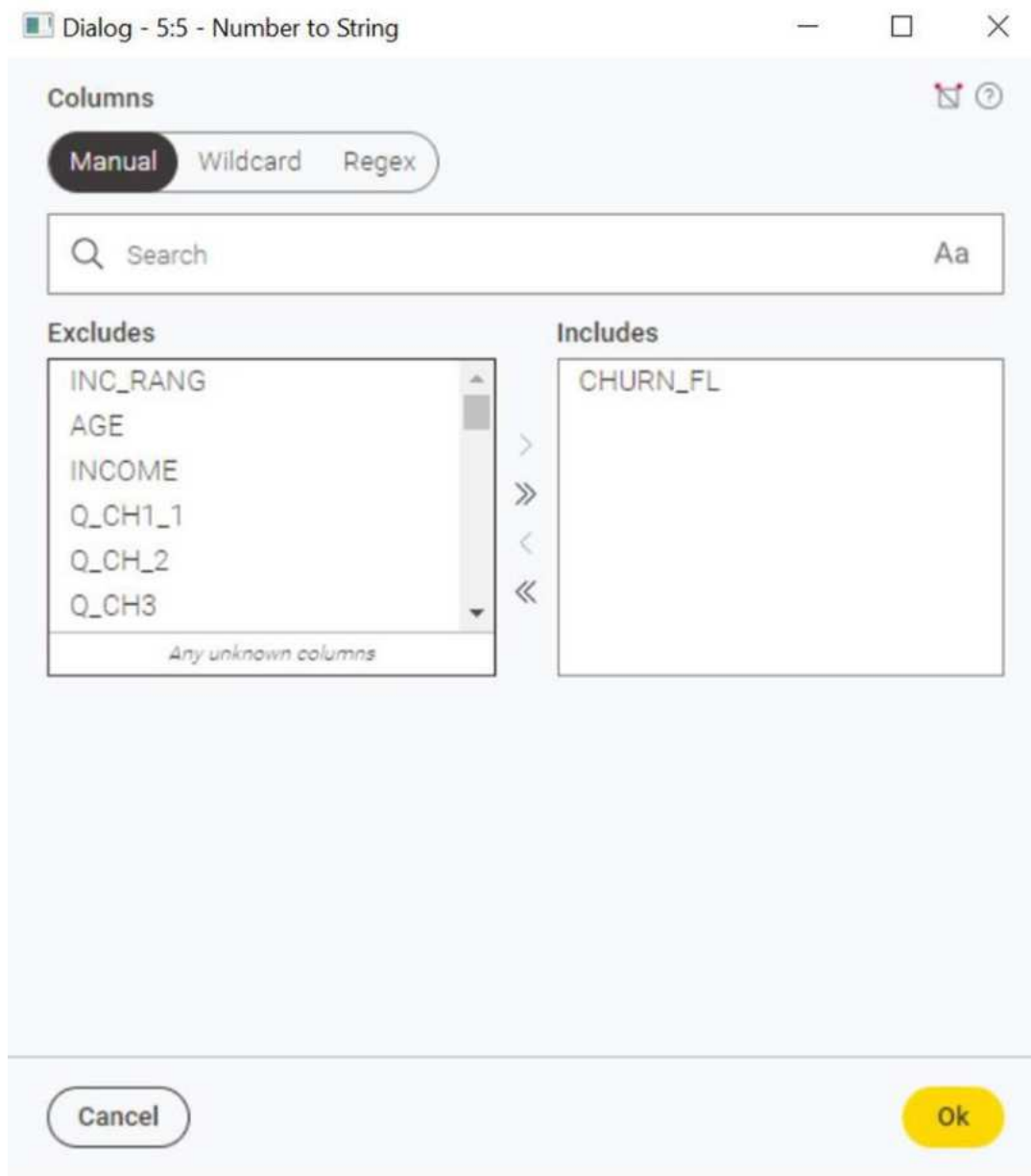


FIG. E.11 The configuration screen of the Number to String node, including only the CHURN_FL variable.

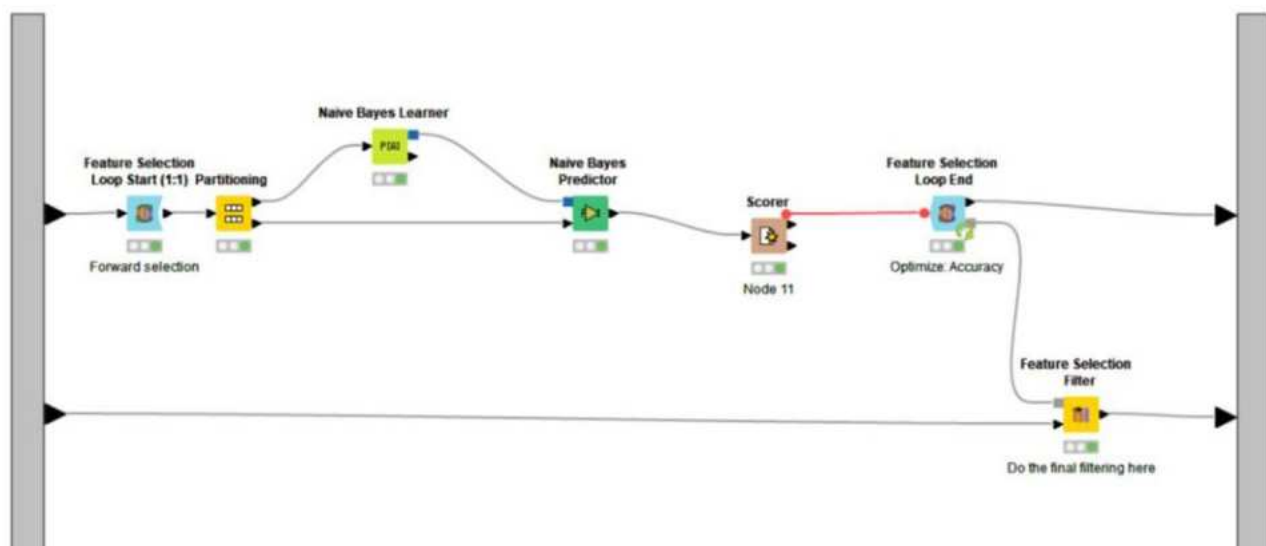


FIG. E.12 The set of KNIME nodes in the Forward Feature Selection metanode.

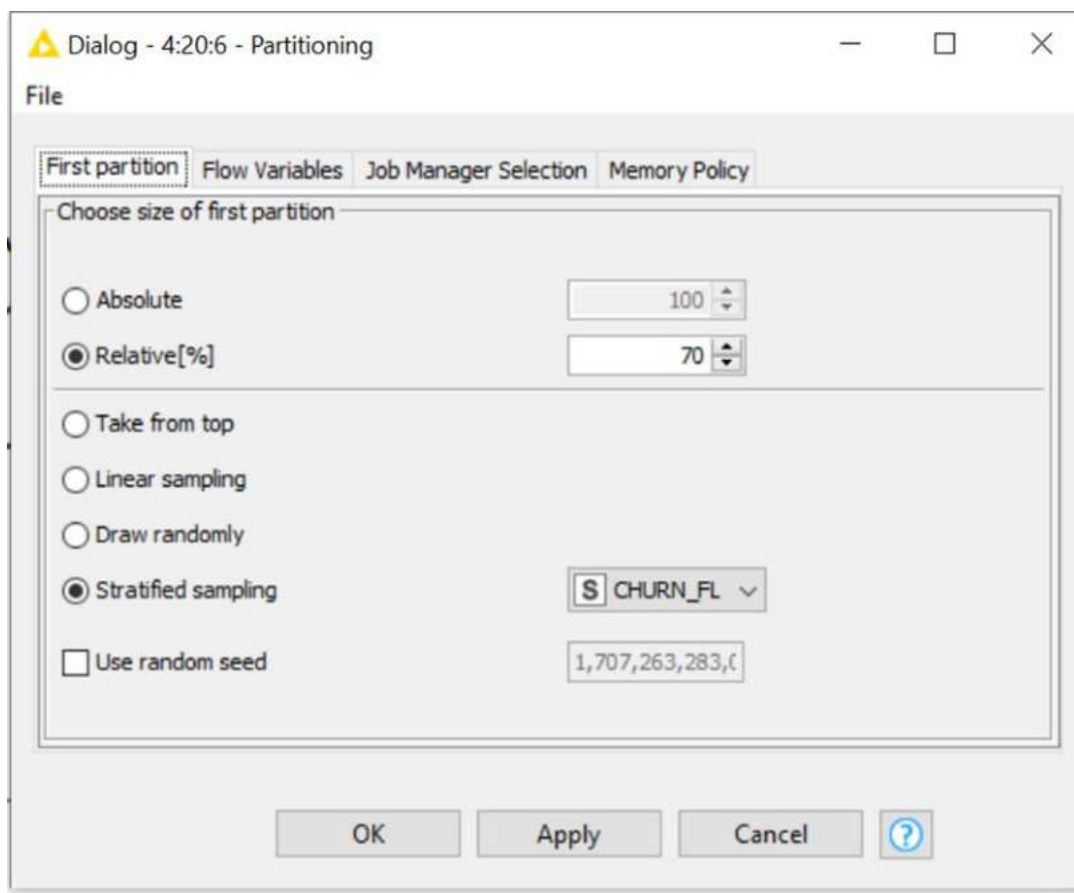


FIG. E.13 The properly configured Partitioning node.

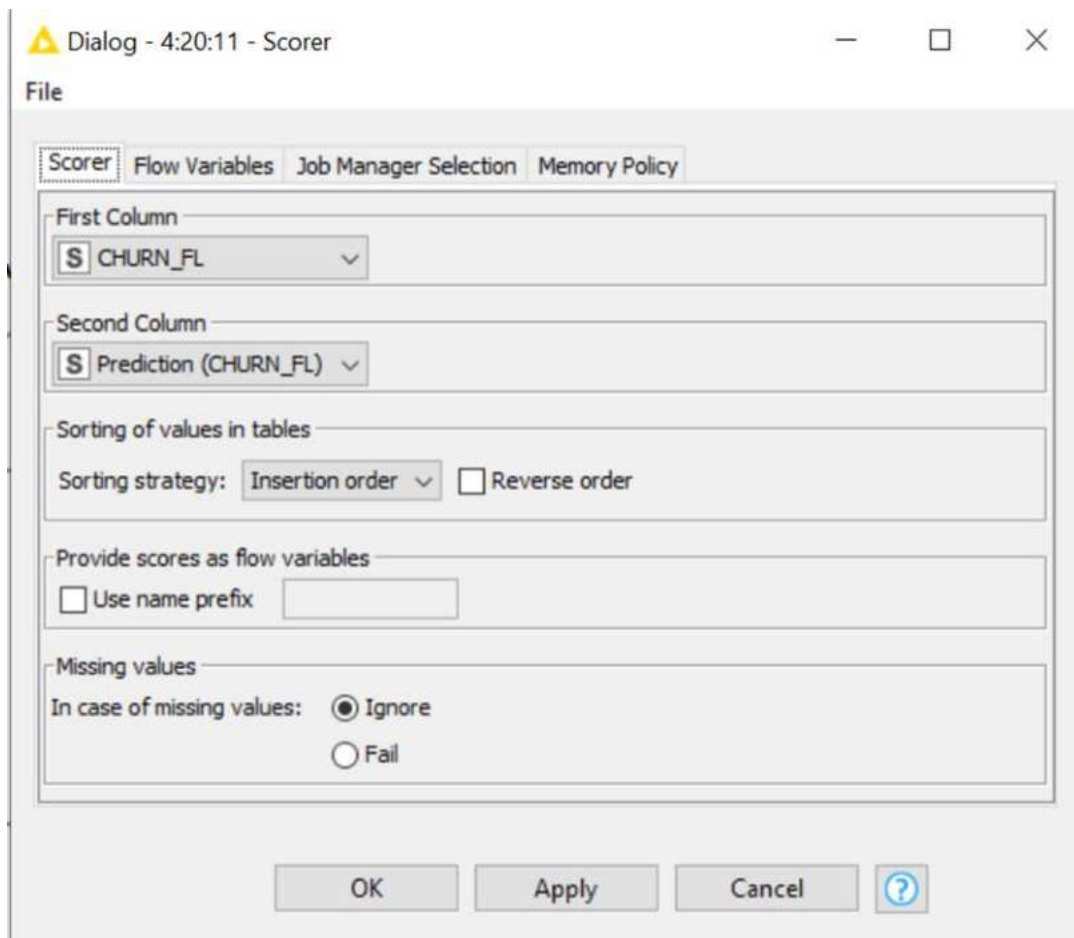


FIG. E.14 The configured Scorer node.

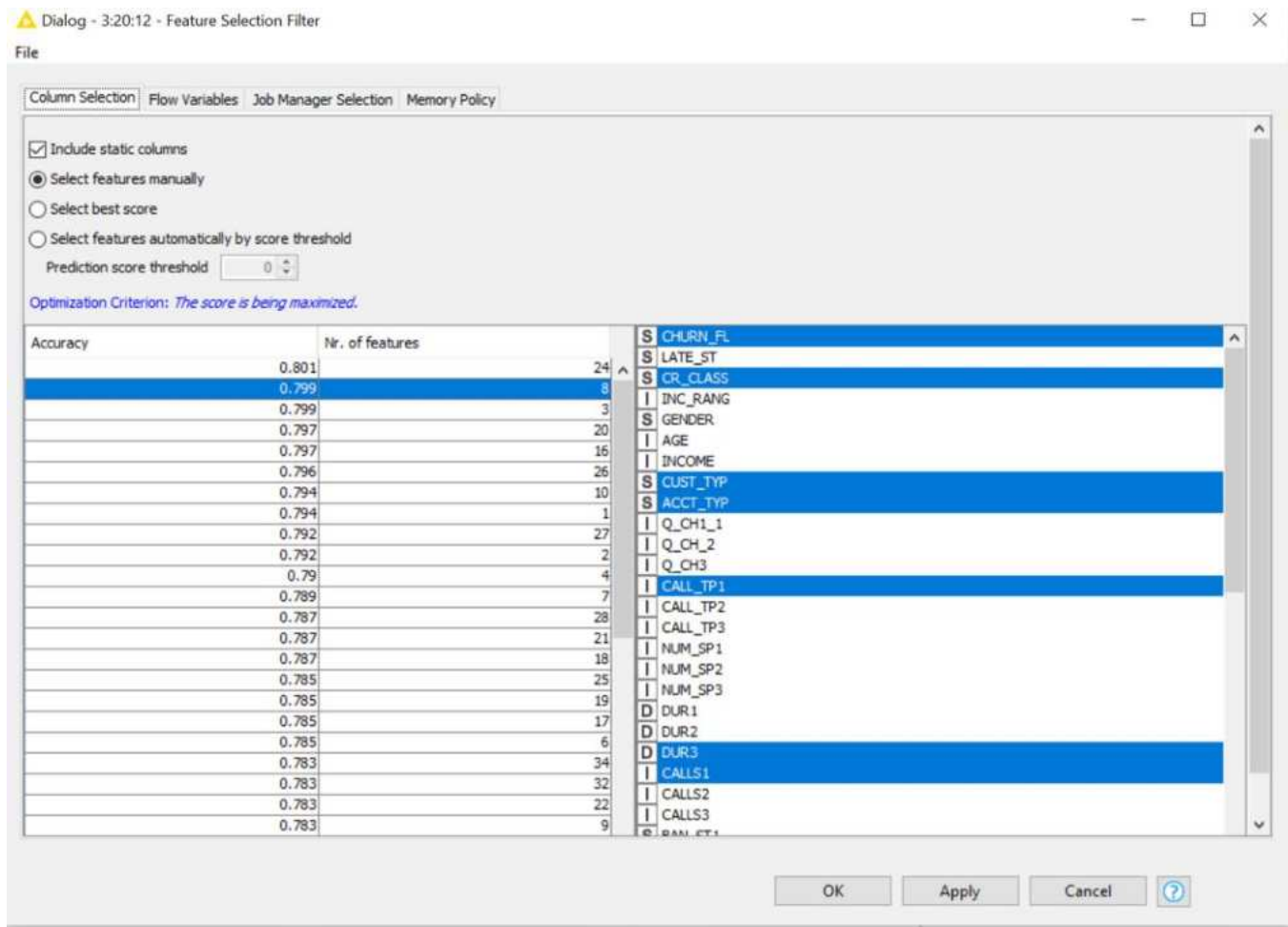


FIG. E.15 The Feature Selection Filter screen.

Row ID	I TruePo...	I FalsePo...	I TrueNe...	I FalseN...	D Recall	D Precision	D Sensitivity	D Specificity	D F-meas...	D Accuracy	D Cohen'...
0	250	100	178	30	0.893	0.714	0.893	0.64	0.794	?	?
1	178	30	250	100	0.64	0.856	0.64	0.893	0.733	?	?
Overall	?	?	?	?	?	?	?	?	?	0.767	0.534

FIG. E.16 The accuracy table of the Random Forest model.

- c. There is no need to configure the Random Forest Predictor node.
 - d. Configure the Scorer node as before, using <CHURN_FL> for the First Column and the <Prediction (CHURN_FL)> for the Second Column.
 - e. Click OK to return to the modeling canvas.
 - f. Right-click on the Scorer node and select the <Execute> option to execute the Scorer node and all upstream nodes in sequence.
 - g. Right-click on the executed Scorer node and select the <Accuracy statistics> table to view the model accuracy metrics.
15. **Save your work!**
 16. Evaluate the trained model.
 - a. Right-click on the Scorer node and select the <Accuracy statistics> option.
 - b. Fig. E.16 shows the overall accuracy of the model is 0.767 or 76.7%.

F

Medical/business tutorial using STATISTICA Data Miner 13 determining possible predictors for days with hospice service for patients with dementia

Linda A. Miner, James Ross, and Karen James

Medicare has a set of guidelines for hospices for admitting patients with dementia conditions to their care. The ideal duration of the service is 6 months (180 days) or less, but prognostication is difficult for the non-cancer patient. The following tutorial models the length of stay of ≤ 180 days in the hospice.

Data were gathered for 449 cases over a period of 6 years in a large hospice on patients with dementia, many of whom had Alzheimer's disease. The following tutorial provides the steps followed to identify a set of predictor variables that would classify patients into two groups (≤ 180 days and > 180 days duration).

Open the data set, Nursing Home Data - Full. First, in the classic menu, the 449 cases were separated randomly into two groups, 50/50, using a random selection. This was done by opening a data mining workspace, inserting the data set, selecting classification and discrimination under the node browser, and finally, the first option, splitting data into training and testing sets. Fig. F.1 shows where to find the workspace and all procedures.

Fig. F.2 shows how to connect the data.

The screen will look like Fig. F.3.

Next, under the node browser, go to Data, then to Manage, and finally to Define Training Testing Sample, as in Fig. F.4.

With the data highlighted, double-click on Define Training Testing Sample and the node will go into the workspace and connect to the data (Fig. F.5).

Right-click on the node to edit the parameters, and to match the configuration in Fig. F.6, click on the X of the validation data and change the numbers to 50/50.

After clicking OK, right-click on the node and select Run so that the two data sets could be formed (Fig. F.7).

Fig. F.8 shows how to click on the tiny document on the node to find the new data set that has both training and testing.

Variable 26 is the new variable that shows training and testing. Use data and then sorting to sort the data into training and testing data (Fig. F.9).

Save the two sets into separate files.

The first group was called the training data set and the second was the testing or holdout data set. Training data were analyzed in an exploratory manner, seeking the variables that seemed most predictive. Then these were applied to the testing set. This tutorial involves only the training data set (attached). Open the training data. Fig. F.10 shows the dependent variables and length of stay (variables 20 and 21).

The dependent variable was the length of stay with the hospice and was used in two forms: a discrete variable of ≤ 180 days versus > 180 days and the service until death, or the actual number of days with the service until death.

The independent variables were gender, BMI, PPS, FAST, Coronary, COPD, Delirium, Pressure, Aspiration, Pneumonia, UTI, Septicemia, Fever Recurrent, Use of Foley, Location at Referral, Marital Status, and Age at Admission. The variable Location at Referral (Variable 15) was not the domiciles of the patients but rather, where they were located at the time they were referred. For example, someone might have been hospitalized, even though living at home. The location at referral

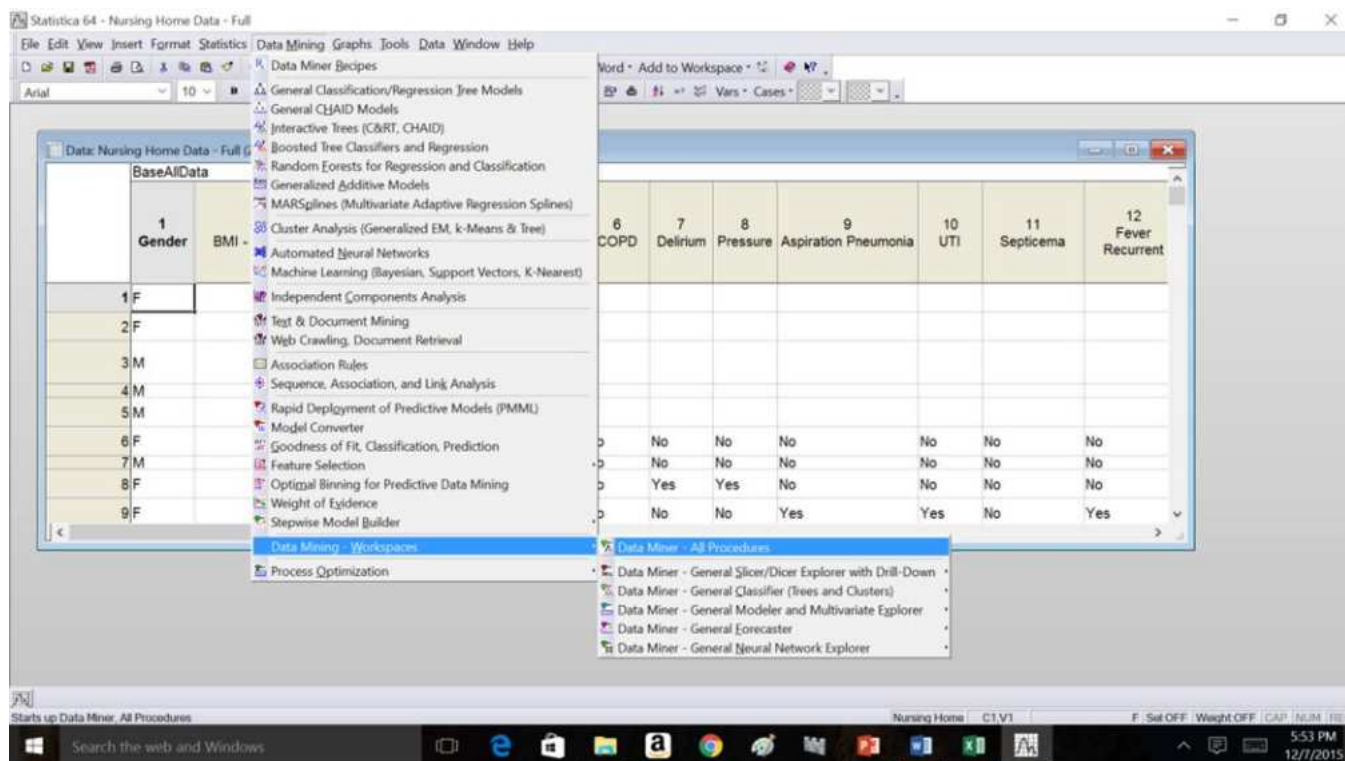


FIG. F.1 Workspace and all procedures.

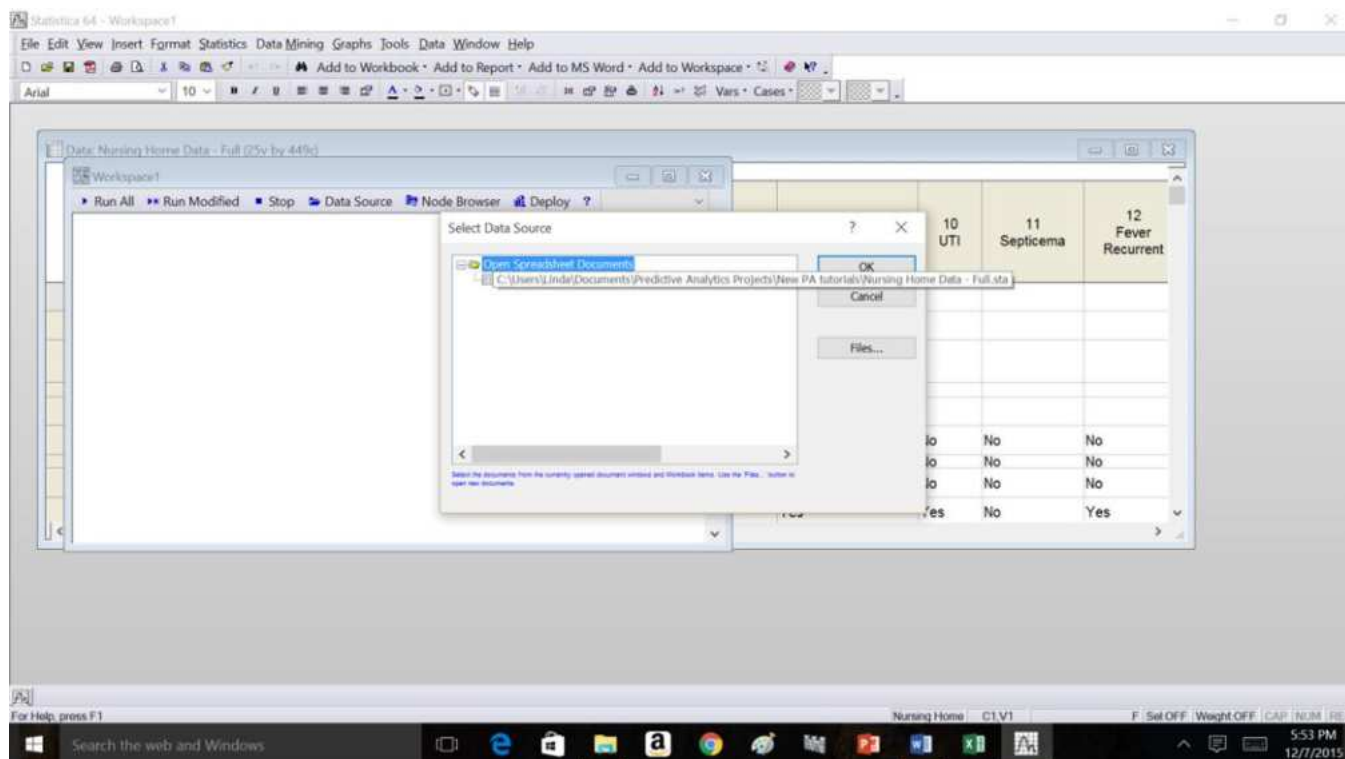


FIG. F.2 Click on the data to highlight and click OK.

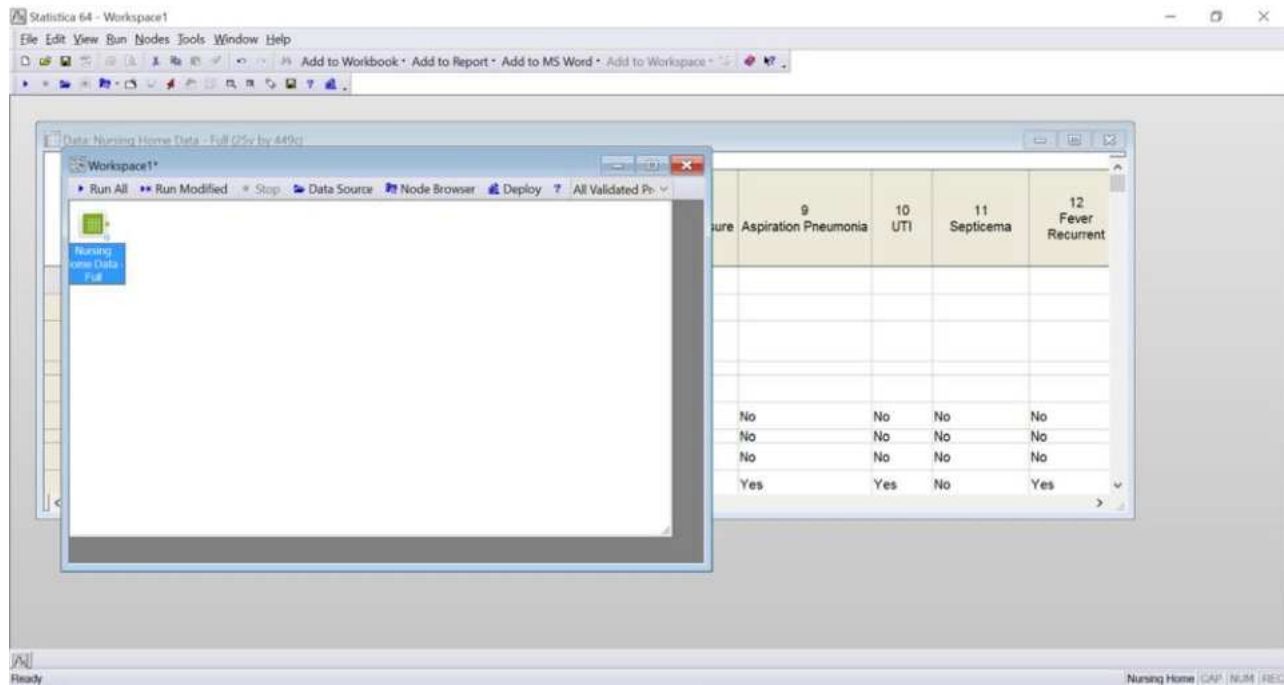


FIG. F.3 Data in the data mining workspace.

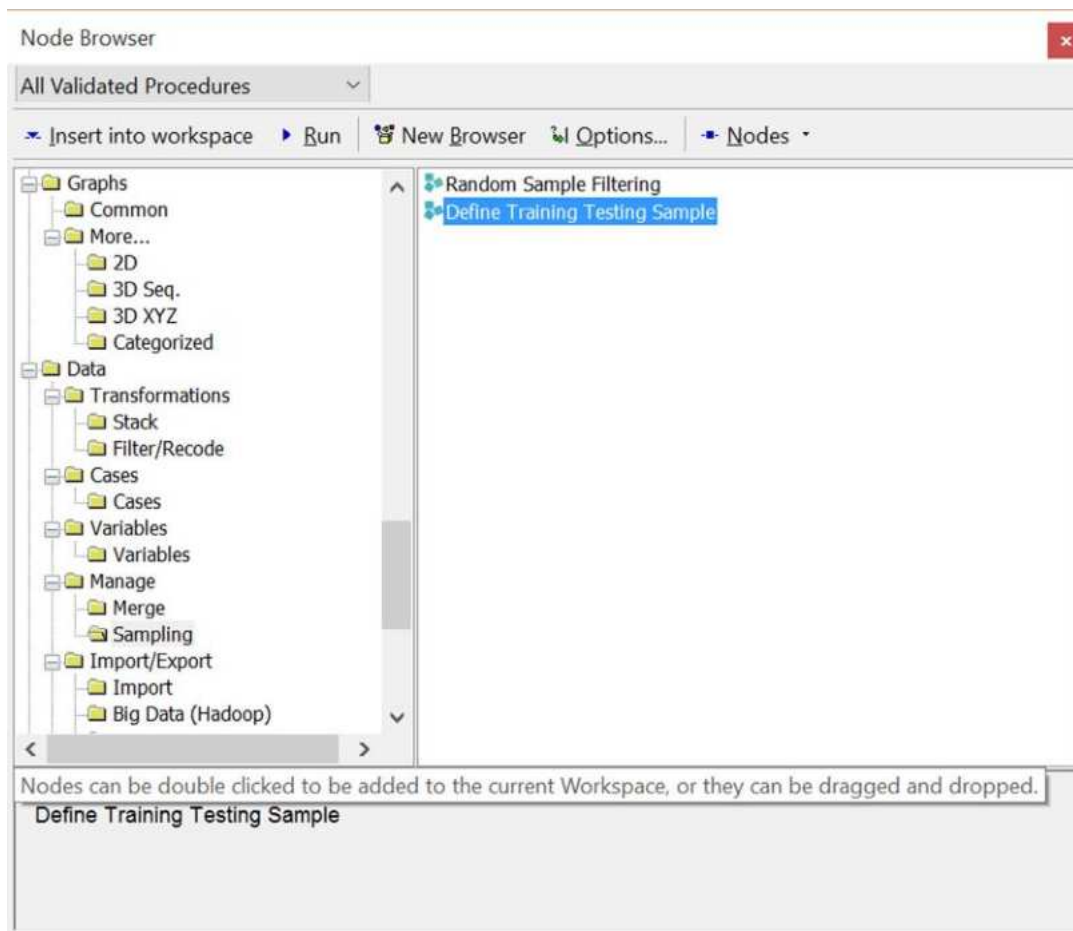


FIG. F.4 Finding Define Training Testing Sample.

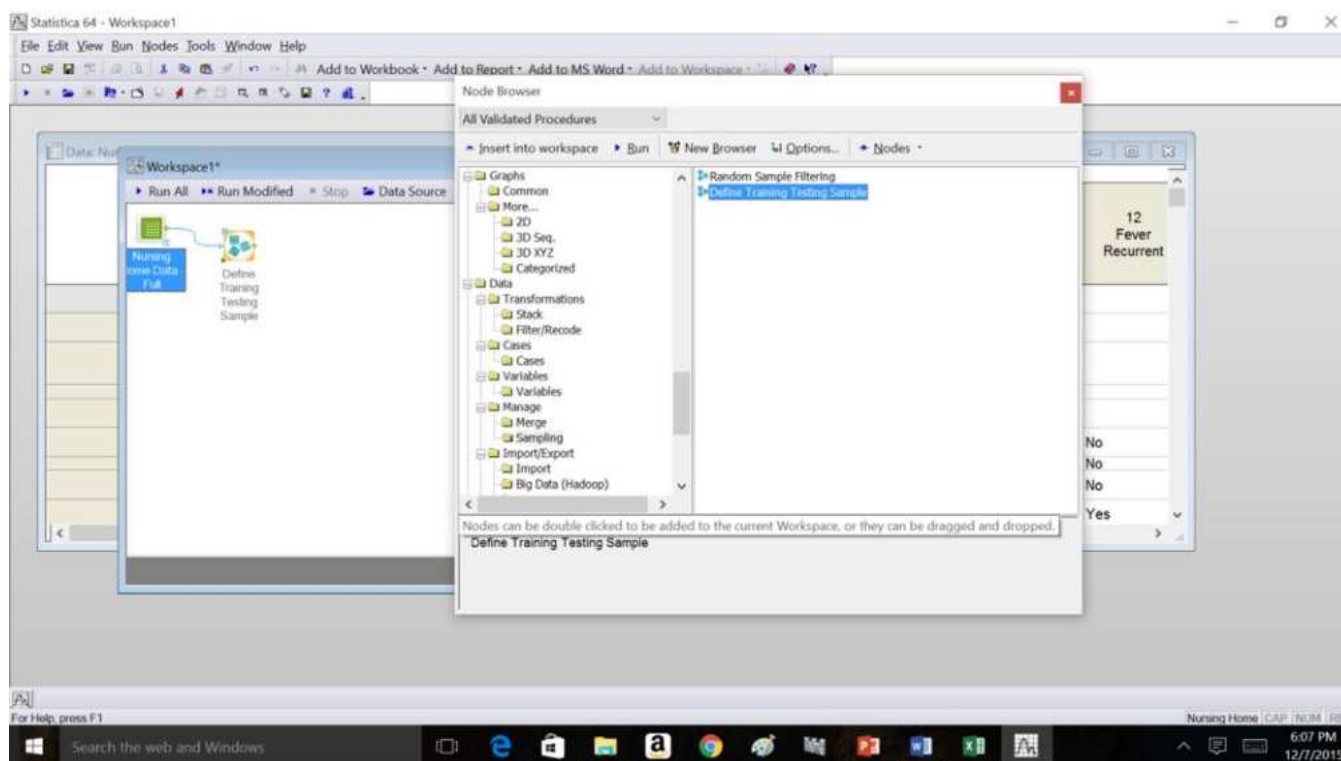


FIG. F.5 Connecting data to Define Training Testing Sample node.

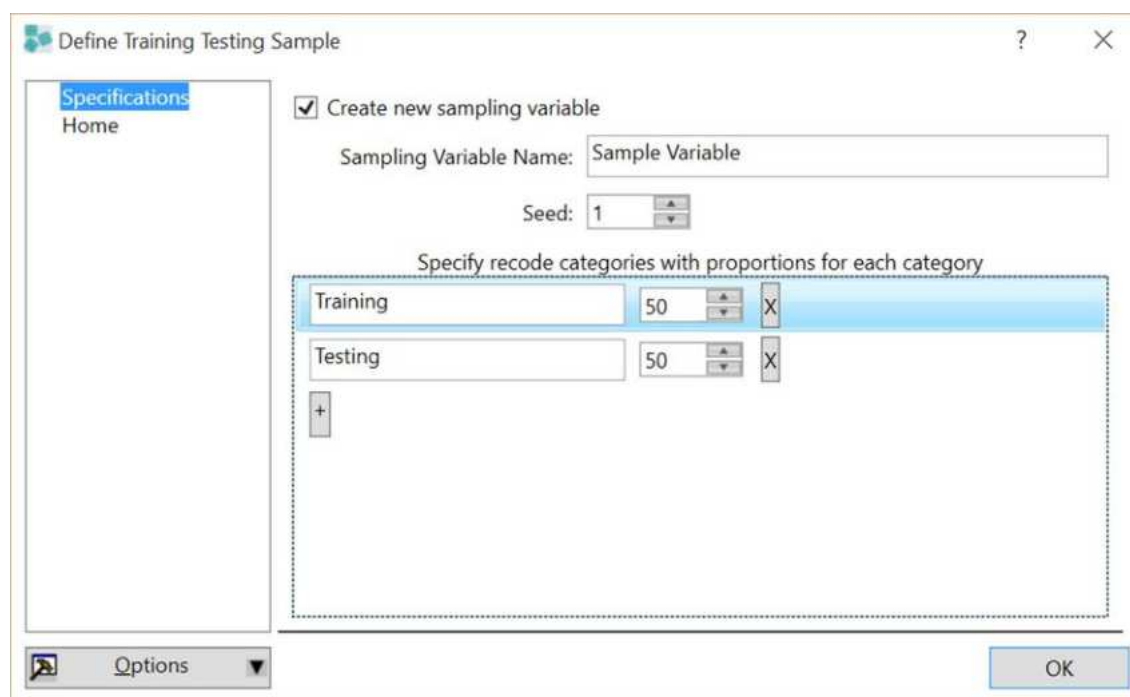


FIG. F.6 Edit the parameters to just Training and Testing.

would then be listed as hospital. BMI and PPS were both continuous variables, as was Age. The rest were categorical variables. The variables included those mandated by Medicare for providing services to Alzheimer's patients.

Working first with the training data, a stepwise multiple regression was done using the number of days (variable 20) as the dependent variable (continuous variable) and the independent variables. Multiple regression was done, rather than feature selection, because it was a more

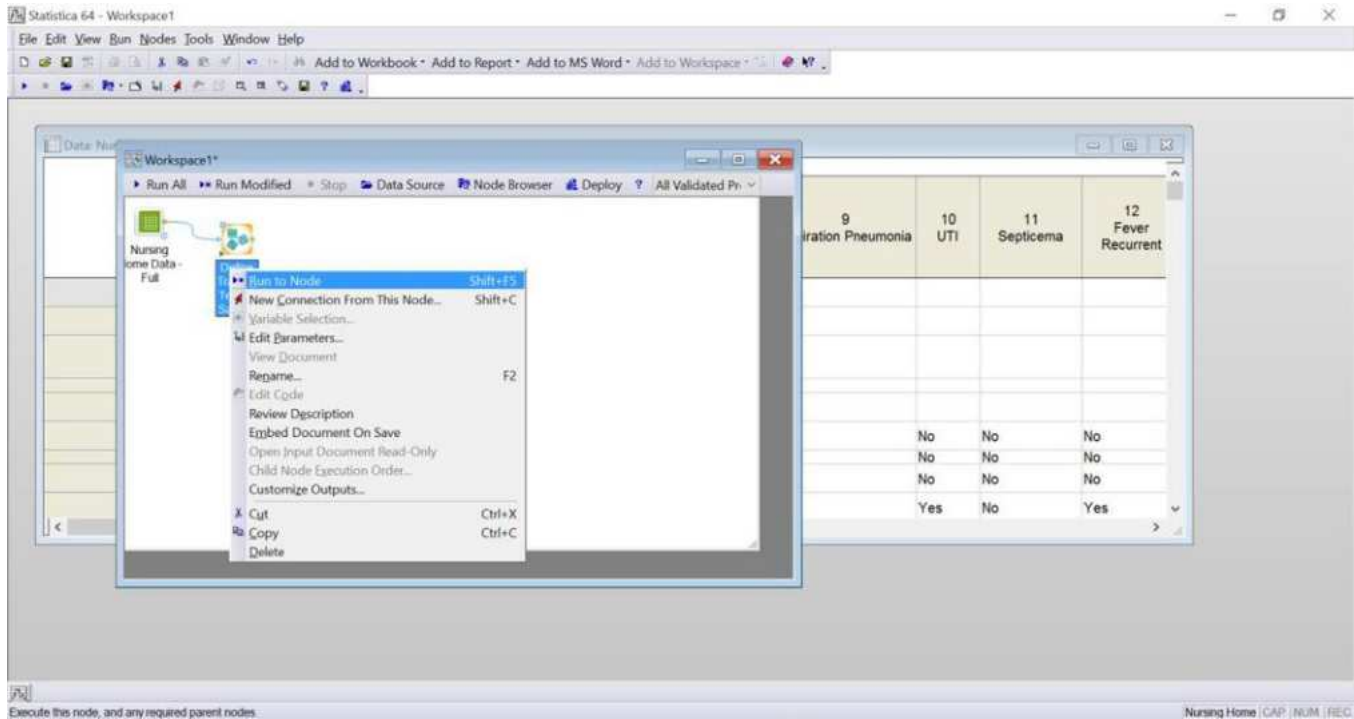


FIG. F.7 How to run the node.

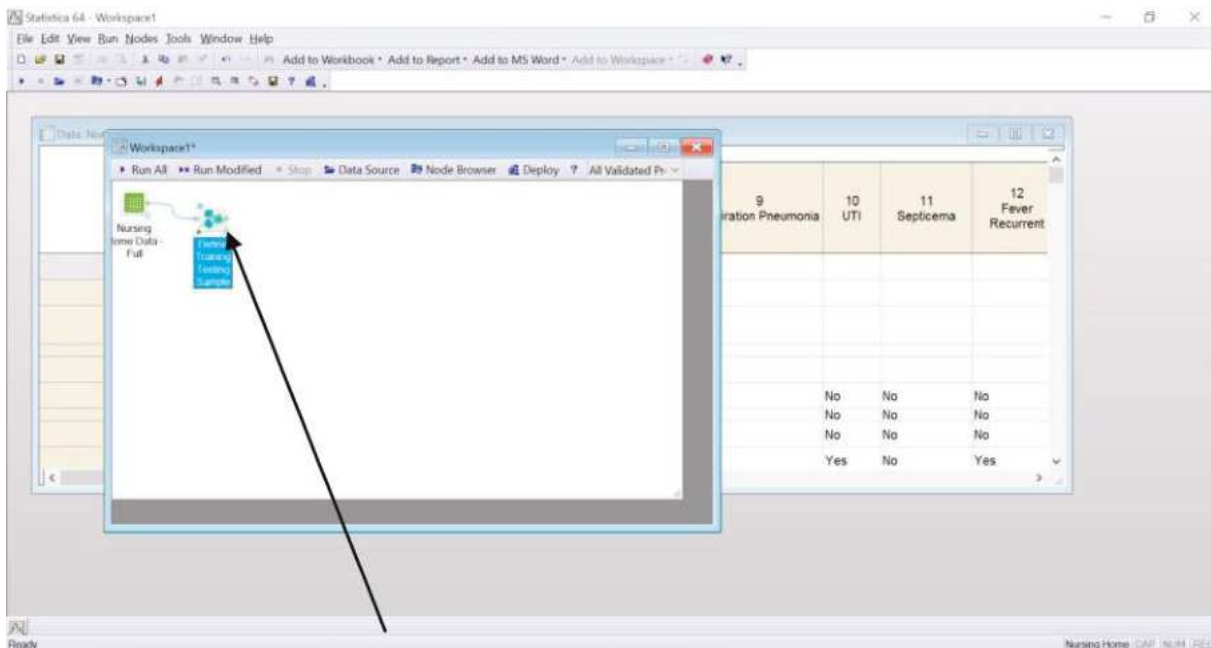


FIG. F.8 Finding the document with the new data set and new variable for training and testing.

powerful procedure than Feature Selection for a continuous variable, although the latter was used later on. Given the relatively small data set, Feature Selection would give us a beginning answer as to which variables might be important for premodeling the data. Variables may be seen in Fig. F.11. The appropriate variables box was unchecked.

The program was told to continue with the current selection in Fig. F.12.

Under the Advanced tab, Advanced options was selected and then OK was clicked (Fig. F.13).

The following emerged (Fig. F.14):

The summary is shown in Table F.1.

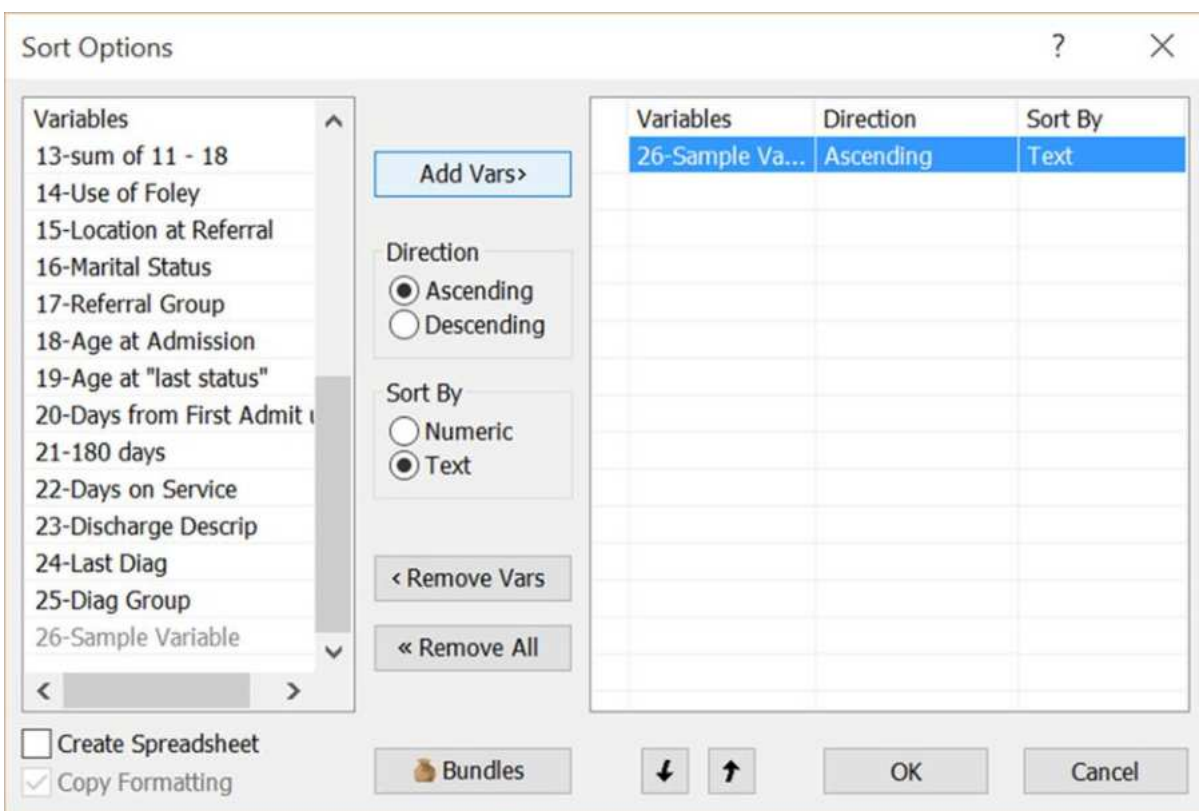


FIG. F.9 Sorting the data – under Data and Sort, variable 26 should be added via Add Vars.

Statistica 64 - Training Data

File Edit View Insert Format Statistics Data Mining Graphs Tools Data Window Help

Arial

10

<

FIG. F.10 The two variables that indicate length of stay – variables 20 and 21.

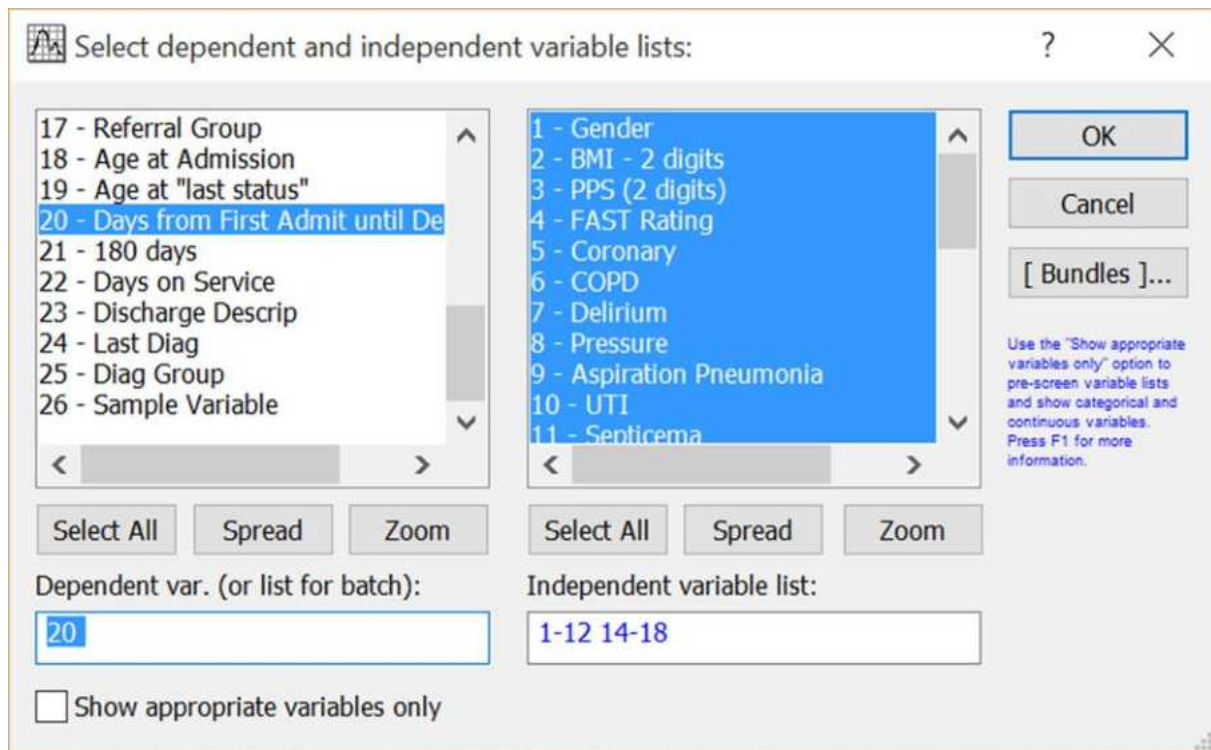


FIG. F.11 Variables that were selected for the analysis.

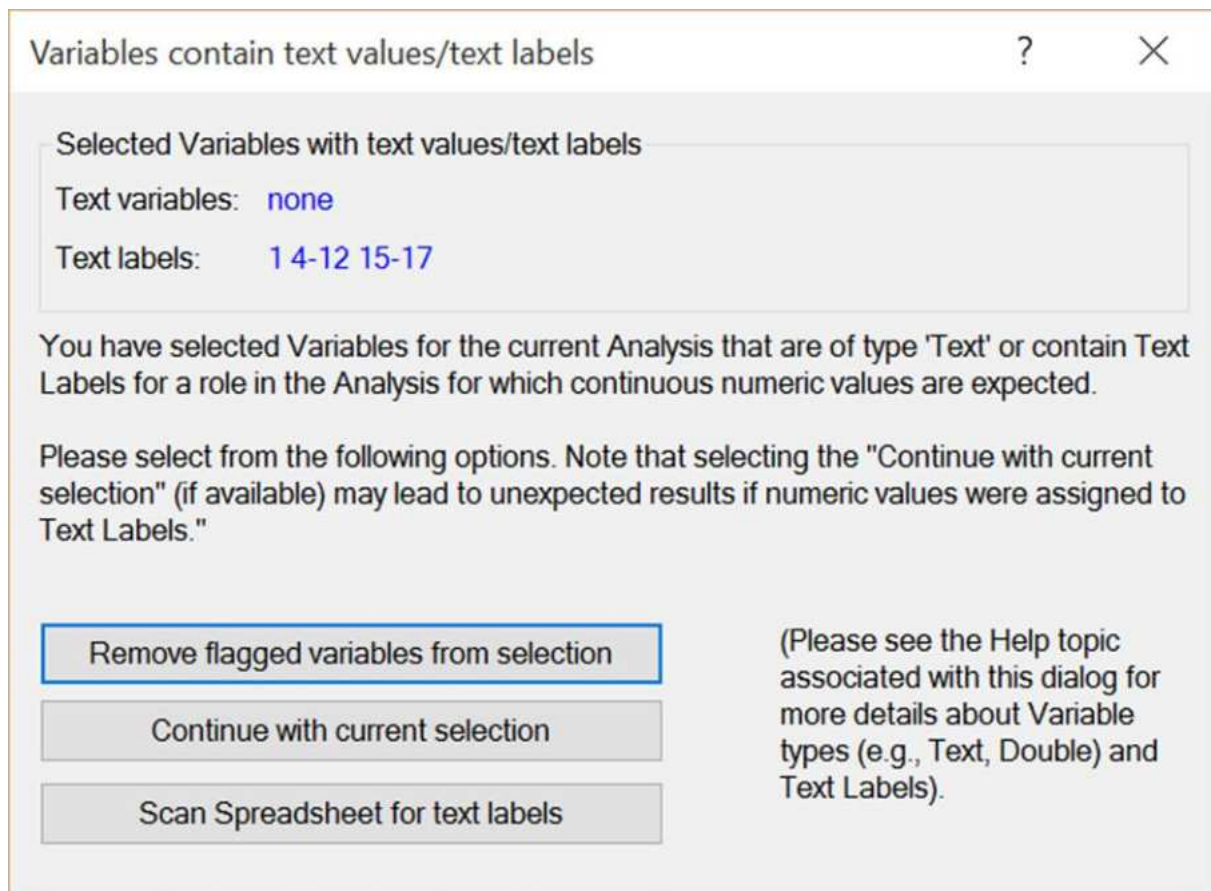


FIG. F.12 Continue with current selection.

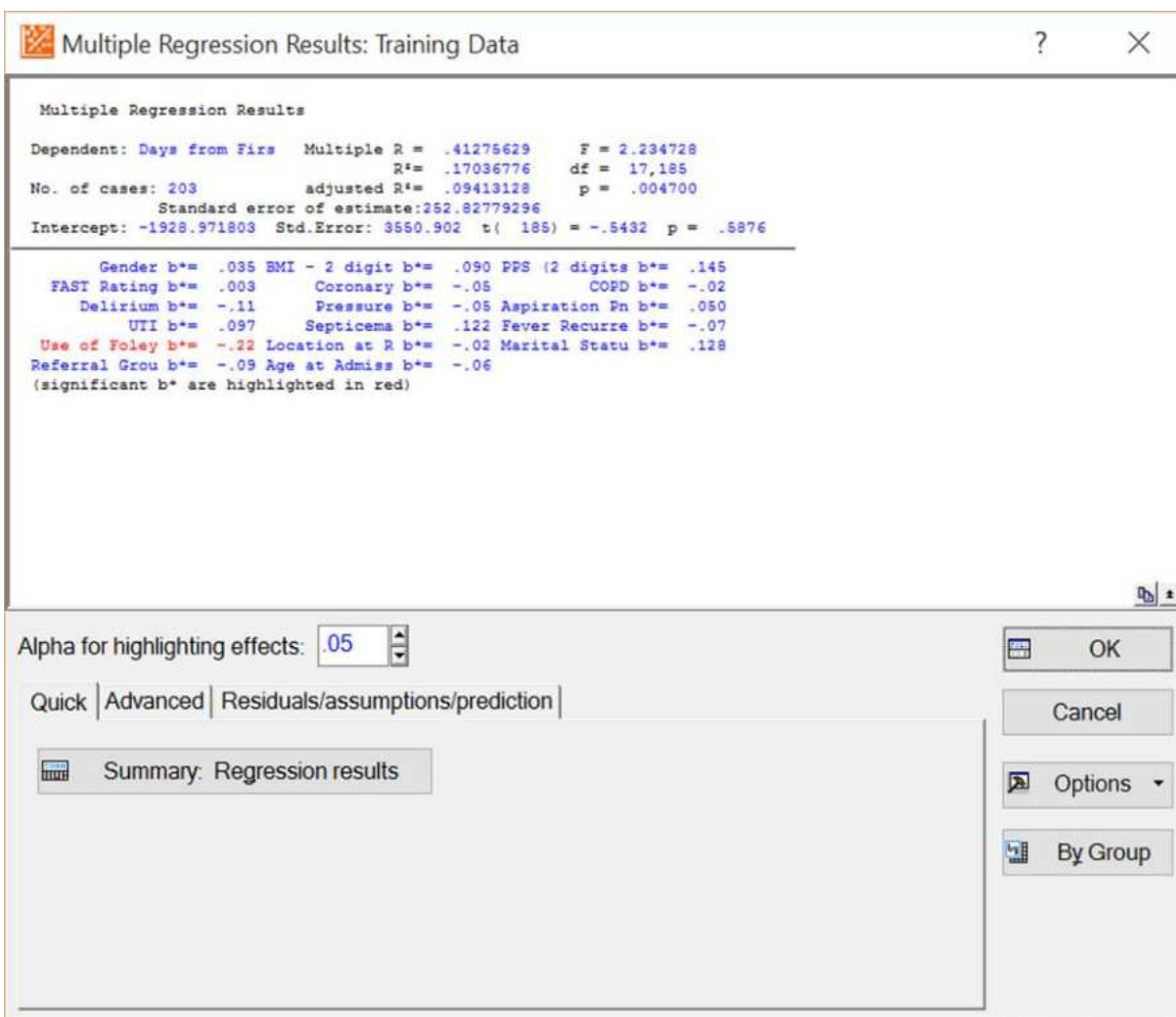
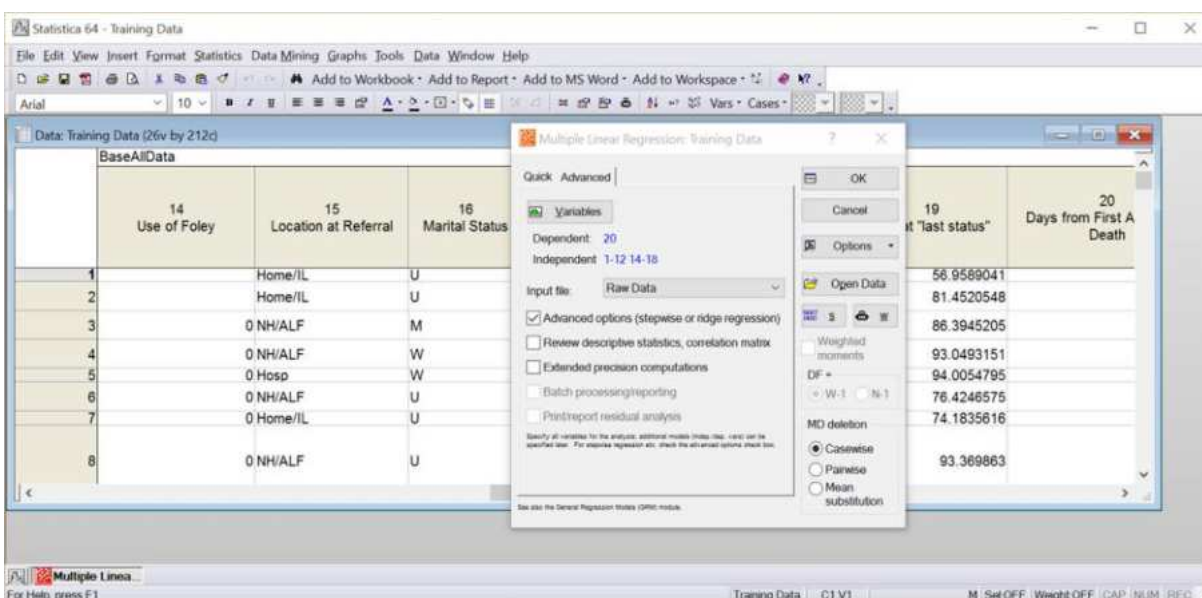


FIG. F.14 Multiple Regression: Use of Foley was highlighted as significant.

TABLE F.1 Summary of results.

Regression Summary for Dependent Variable: Days from First Admit Until Death (Training Data) R=.41275629 R ² =.17036776 Adjusted R ² =.09413128 F(17,185)=2.2347 p<.00470 std.Error of estimate:252.83						
N=203	b*	Std.Err. of b*	b	Std.Err. of b	t(185)	p-value
Intercept			-1928.97	3550.901	-0.54323	0.587622
Gender	0.035319	0.068889	1.64	3.198	0.51270	0.608773
BMI 2 digits	0.090038	0.073454	5.25	4.285	1.22577	0.221843
PPS (2 digits)	0.145195	0.079361	4.42	2.417	1.82955	0.068927
FAST Rating	0.003164	0.076632	0.53	12.764	0.04129	0.967106
Coronary	-0.045612	0.070172	-26.13	40.193	-0.65000	0.516497
COPD	-0.015950	0.069573	-13.60	59.307	-0.22926	0.818924
Delirium	-0.112140	0.070840	-66.49	41.999	-1.58300	0.115130
Pressure	-0.054794	0.075601	-36.17	49.899	-0.72477	0.469510
Aspiration Pneumonia	0.050429	0.073435	34.61	50.404	0.68672	0.493119
UTI	0.096734	0.073546	52.96	40.268	1.31529	0.190040
Septicemia	0.121894	0.072304	136.96	81.241	1.68585	0.093511
Fever Recurrent	-0.069383	0.074790	-75.10	80.948	-0.92771	0.354765
Use of Foley	-0.219293	0.078698	-148.97	53.462	-2.78651	0.005883
Location at Referral	-0.021191	0.073089	-8.05	27.783	-0.28993	0.772197
Marital Status	0.128429	0.070482	36.55	20.056	1.82215	0.070048
Referral Group	-0.086254	0.072612	-9.29	7.819	-1.18787	0.236408
Age at Admission	-0.057840	0.071139	-2.28	2.808	-0.81305	0.417236

The variable highlighted in red, Use of Foley, was the one deemed probably most important to the model and most certainly would be selected for further data mining analysis. Because we mixed continuous and categorical variables, we did a feature selection just to check this result. We opened a data mining workspace (under the Data Mining tab), copied in the data (highlighted and entered), and selected the variables of interest. Fig. F.15 shows that in the Node Browser, under Data Mining, Tools and then Feature Selection need to be clicked.

With the data highlighted, double-click Feature Selection, which will insert the node into the workspace and connect to the data as in Fig. F.16. Be sure to highlight the data before double-clicking.

We right-clicked on the Feature Selection node to Edit the Parameters (Fig. F.17).

Select the variables in the first screen. Fig. F.18 shows the variables selected.

Continue with current selection. We also used the defaults under the results tab.

Click OK. Right-click on the node and run the node as in Fig. F.19.

Click on the reporting documents to find the box in Fig. F.20.

We can see the importance plots for both target variables. The use of the Foley is the top for both targets. However, it is important to note that this procedure is not a hypothesis test and should not be viewed as such. It is only a pattern-seeking procedure.

To make sure we could open the output later, we embedded the document on Save as in Fig. 21.

First, look at the important variables for the continuous dependent variable (Table F.2) and then the important variables for the categorical variable (Table F.3).

Use of Foley was most important for both targets. Then other variables differed some. The business department of the nursing home was most concerned with trying to predict less than or greater than 180 days, because that was how payments were decided by the government. Therefore we decided to concentrate on the discrete variable as the dependent variable for the subsequent analyses.

Next, a Data Mining Recipe was done using 180 days (the discrete variable) as the dependent variable and all

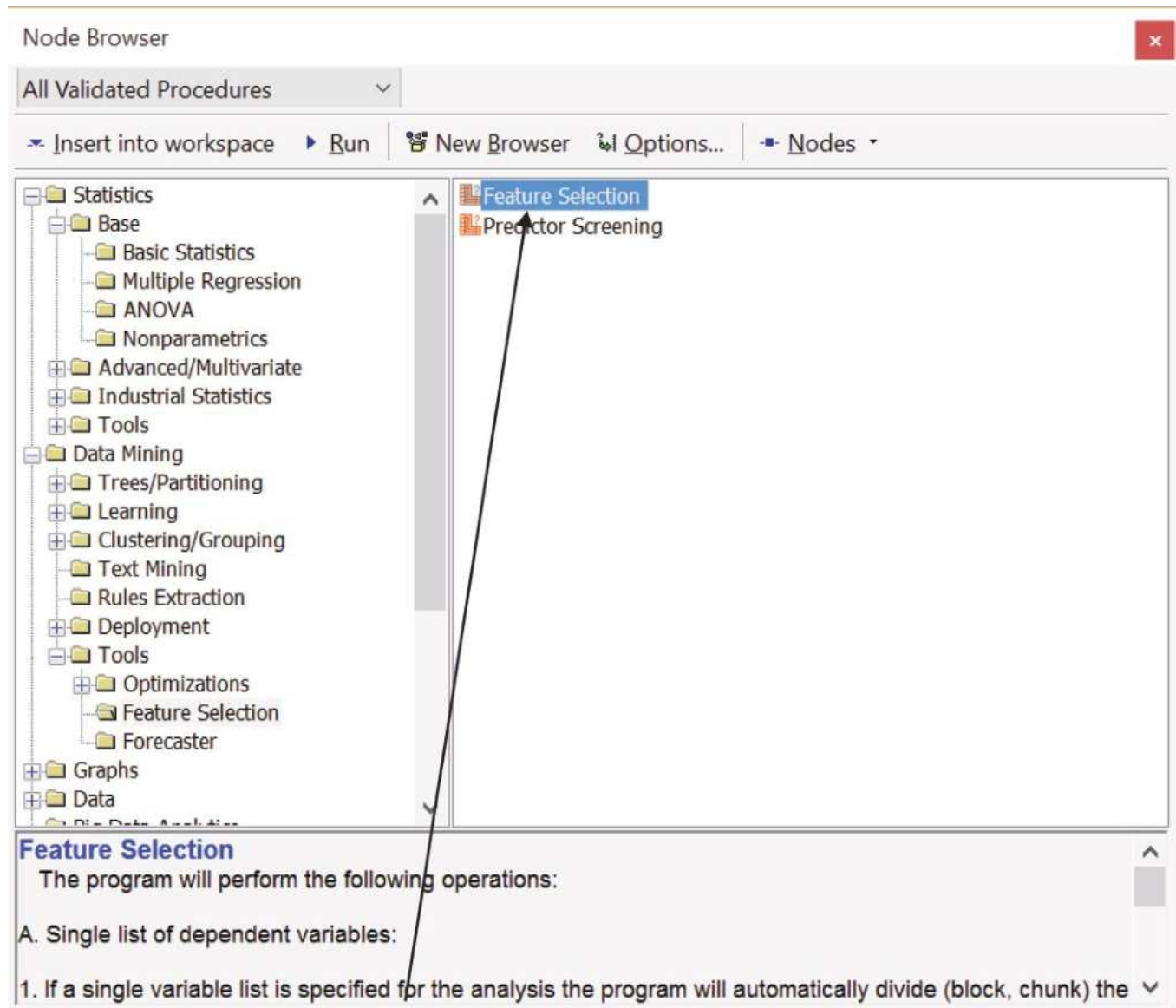


FIG. F.15 Finding Feature Selection.

the variables (1–12, 14–26, and 18) as above for the independent variables. Fig. F.22 shows where to find the Data Mining Recipe in the ribbon view.

“New” was selected in Fig. F.23.

Fig. F.24 shows that we connect the data by clicking on “Open/Connect Data File” and selecting our open data file in this window. Click OK.

Next, we need to select the variables, as in Fig. F.25.

Click OK. The error screen in Fig. F.26 emerged, because the program was perceiving FAST Rating as categorical. Just click Continue with current selection.

In Fig. F.27, we can see what the project looks like at this point. Note the red Xs. We will want to have the program select a testing sample for us, so click “Configure all steps” and we can get to that part.

Click on Data for analysis, select Testing Sample, and then specify to have the program select a random sample of 20% of the cases, as in Fig. F.28.

Click OK. Under Target variable, first select Important Variables so that we can again get a feature selection. In Fig. F.29, we have selected a Fast predictor screening. We can compare the lists at the end if we like.

Next, click on Model building and select the two that are not checked in the default so that we can do all the models possible. See Fig. F.30.

Uncheck Configure all steps and select Run to completion, as in Fig. F.31.

Allow the program to run. These data had missing values, so the warning in Fig. F.32 came up. Just click OK.

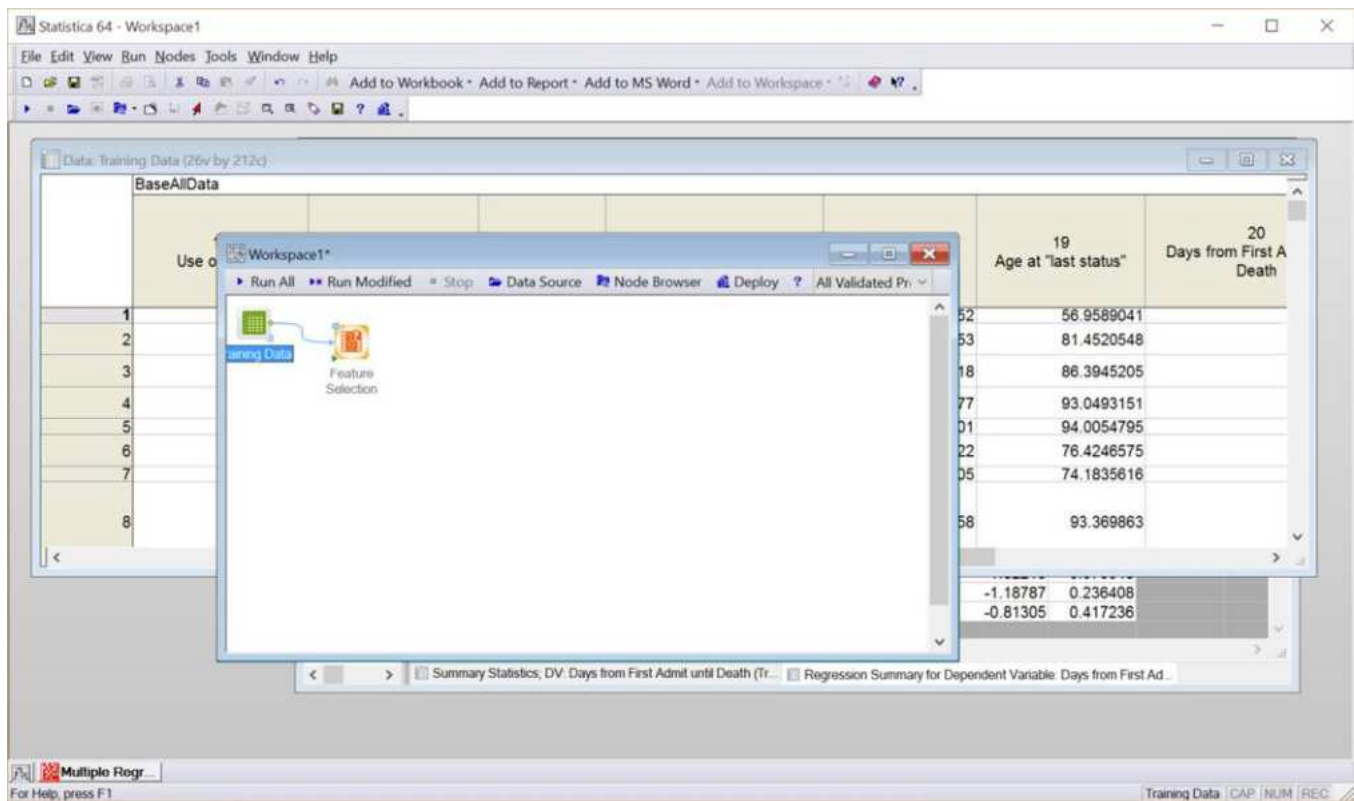


FIG. F.16 Workspace with Feature Selection.

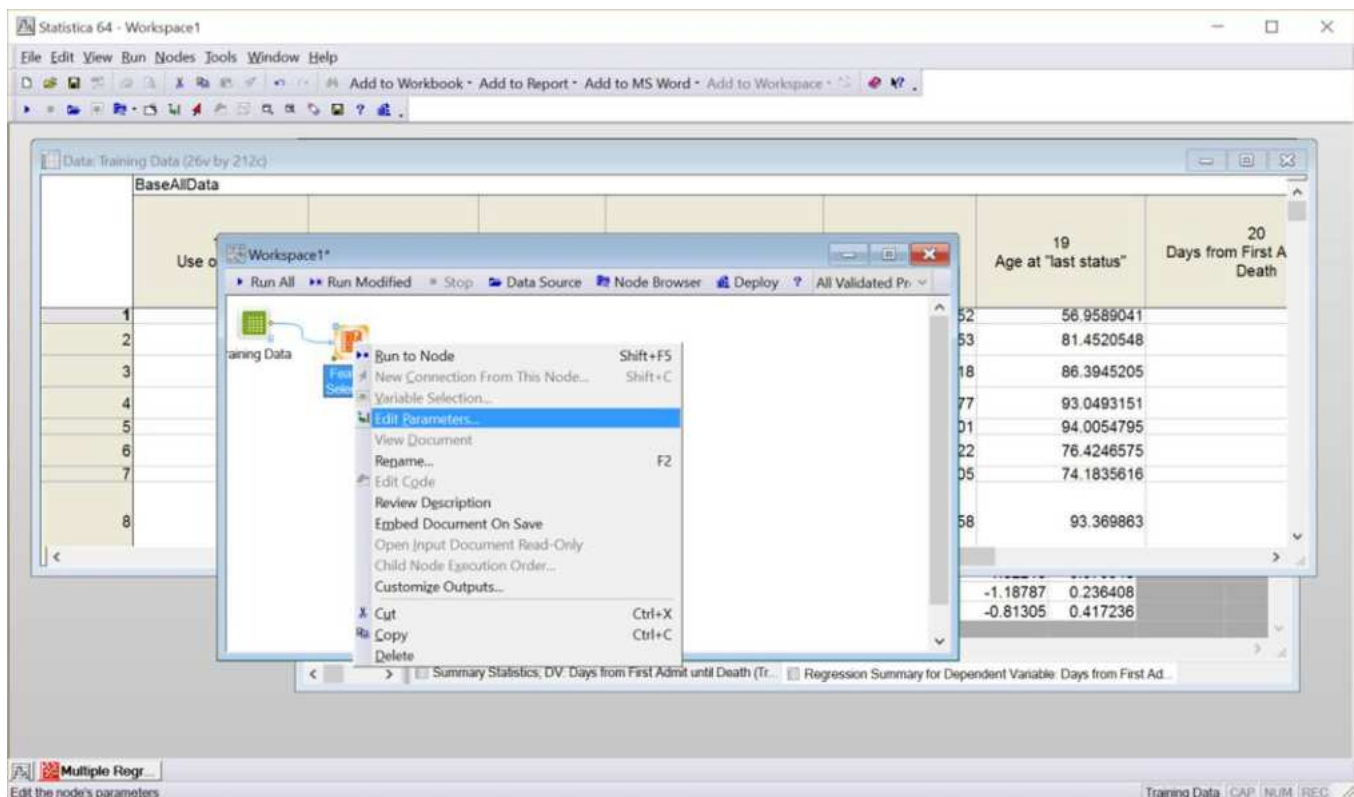


FIG. F.17 Edit Parameters.

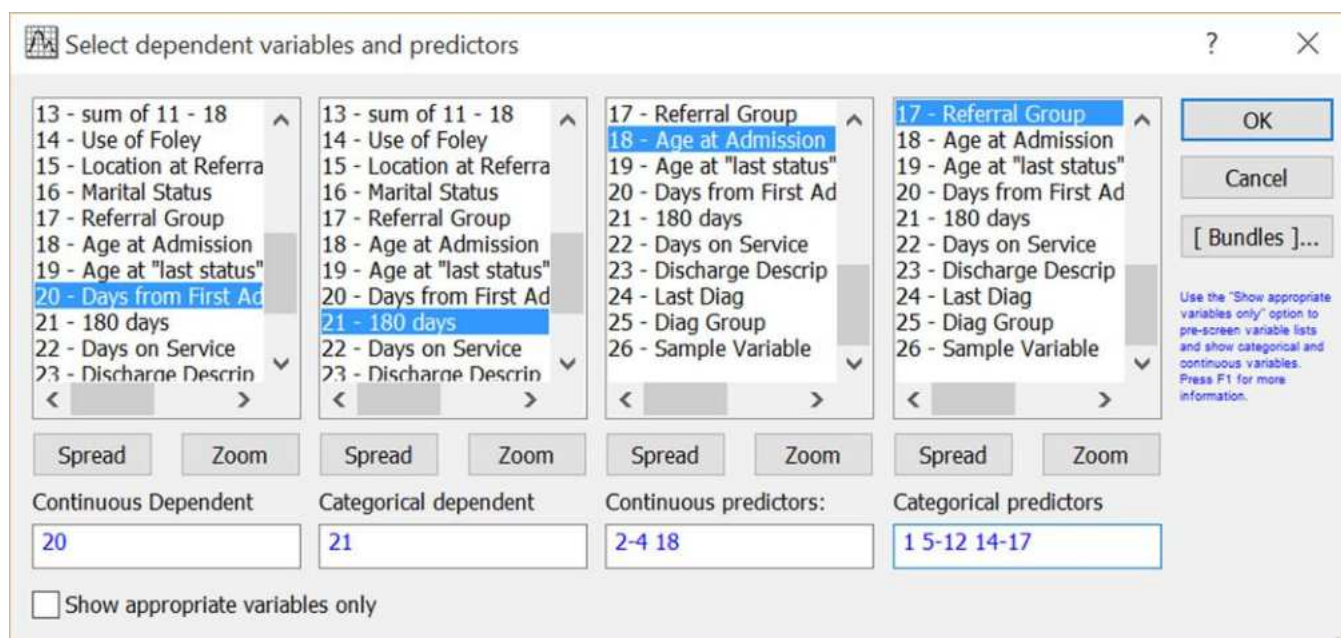


FIG. F.18 Variables selected.

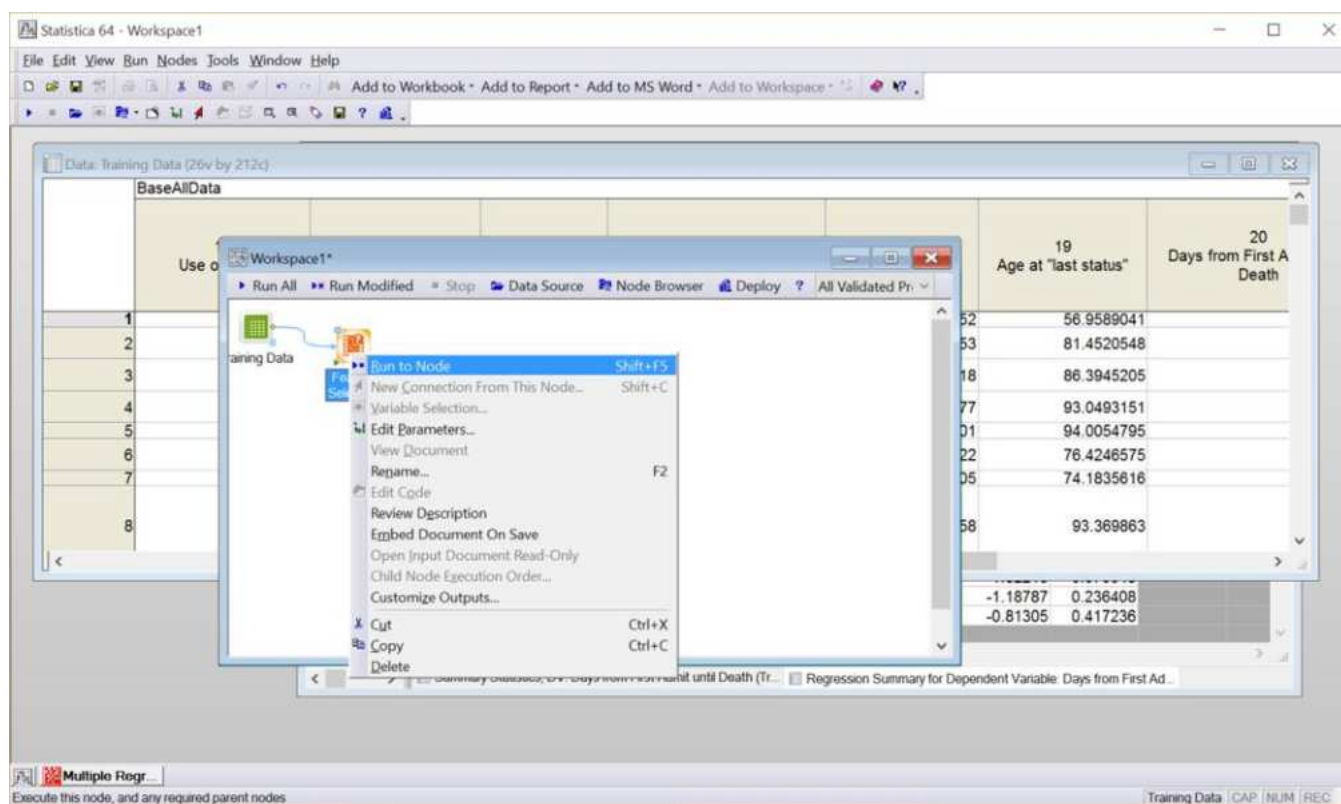


FIG. F.19 Running the node.

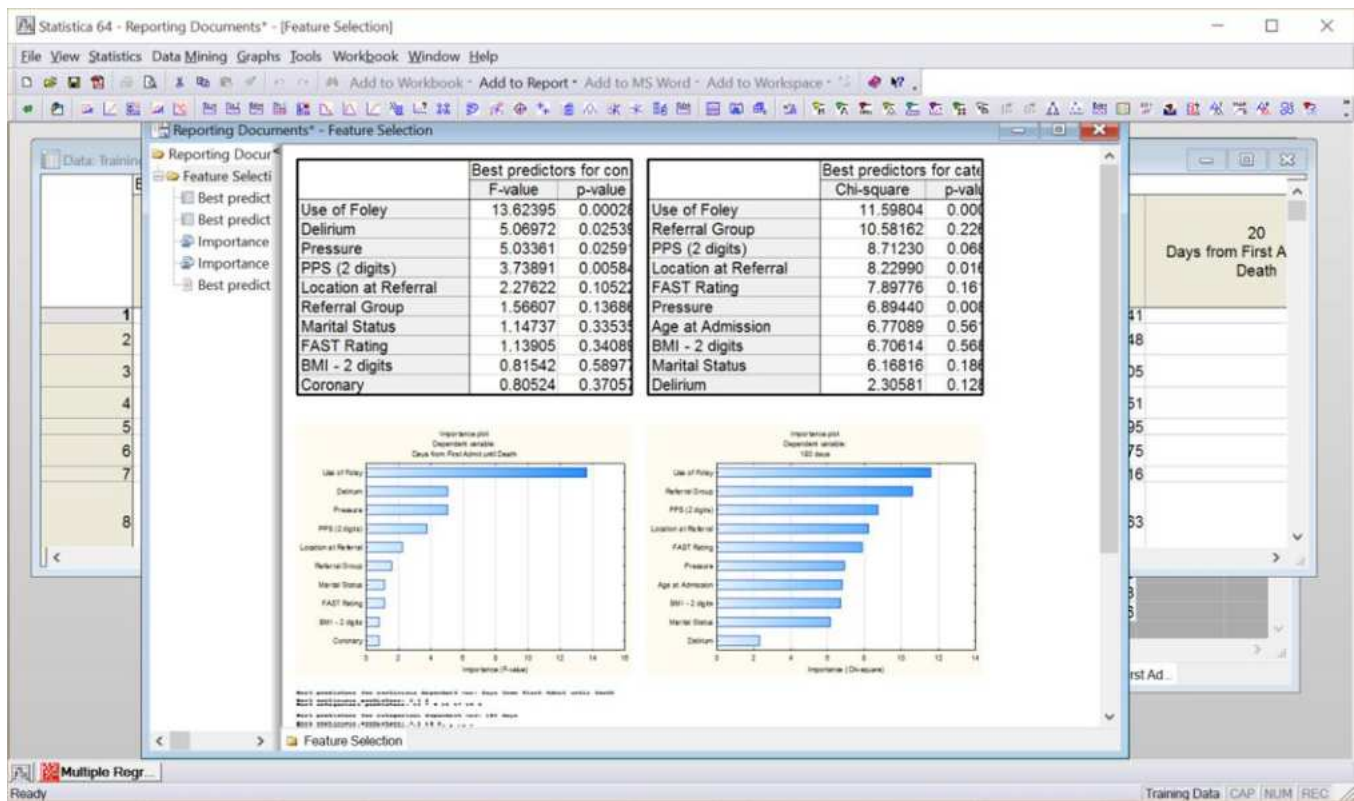


FIG. F20 Results in Reporting Documents.

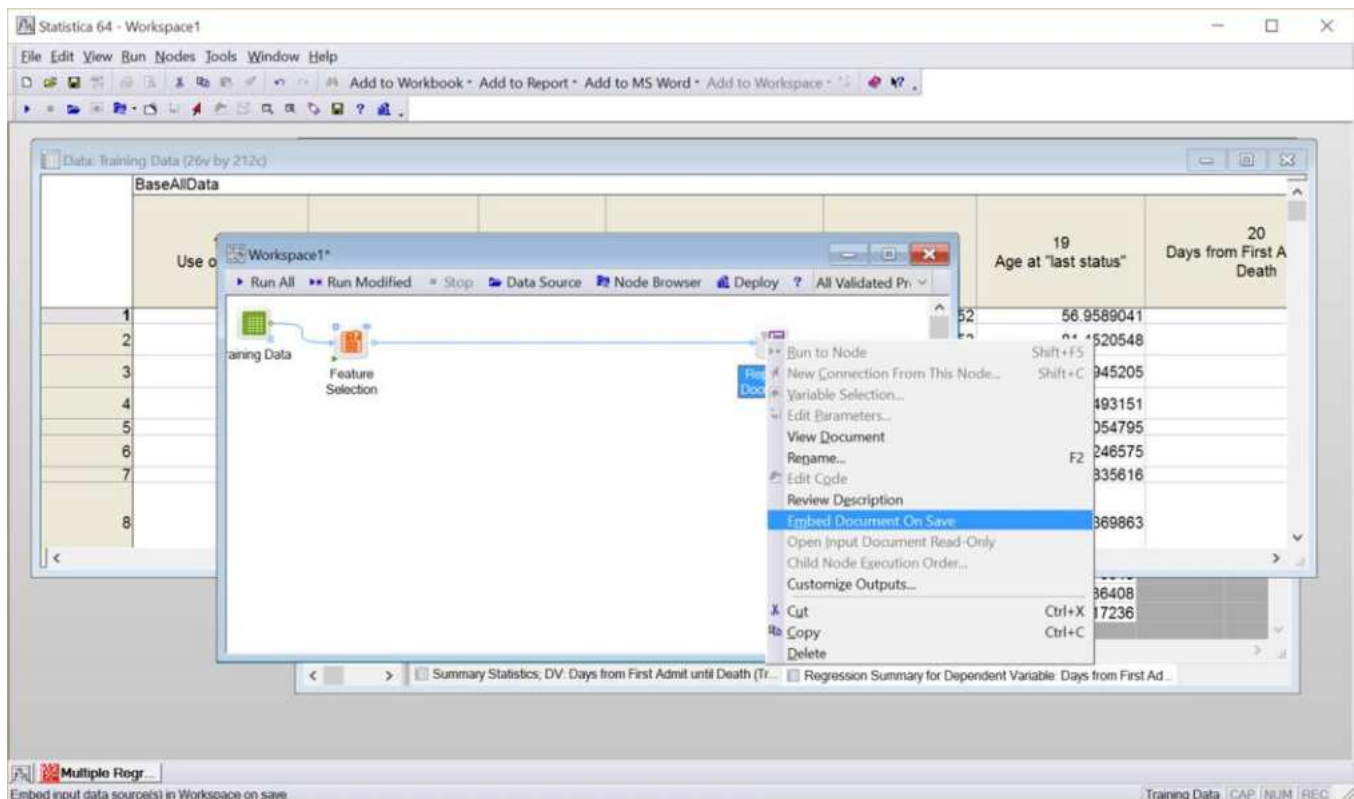


FIG. F21 Embed Document on Save will allow you to save the workbook and see the output documents later.

TABLE F.2 Important variables for continuous target.

Best predictors for continuous dependent var:		
	F-value	p-value
Use of Foley	13.62395	0.000285
Delirium	5.06972	0.025392
Pressure	5.03361	0.025914
PPS (2 digits)	3.73891	0.005846
Location at Referral	2.27622	0.105224
Referral Group	1.56607	0.136863
Marital Status	1.14737	0.335356
FAST Rating	1.13905	0.340899
BMI - 2 digits	0.81542	0.589777
Coronary	0.80524	0.370570

TABLE F.3 Important variables for categorical target.

Best predictors for categorical dependent var:		
	Chi-square	p-value
Use of Foley	11.59804	0.000660
Referral Group	10.58162	0.226550
PPS (2 digits)	8.71230	0.068707
Location at Referral	8.22990	0.016327
Fast Rating	7.89776	0.161961
Pressure	6.89440	0.008647
Age at Admission	6.77089	0.561542
BMI-2 digits	6.70614	0.568644
Marital Status	6.16816	0.186937
Delirium	2.30581	0.128891

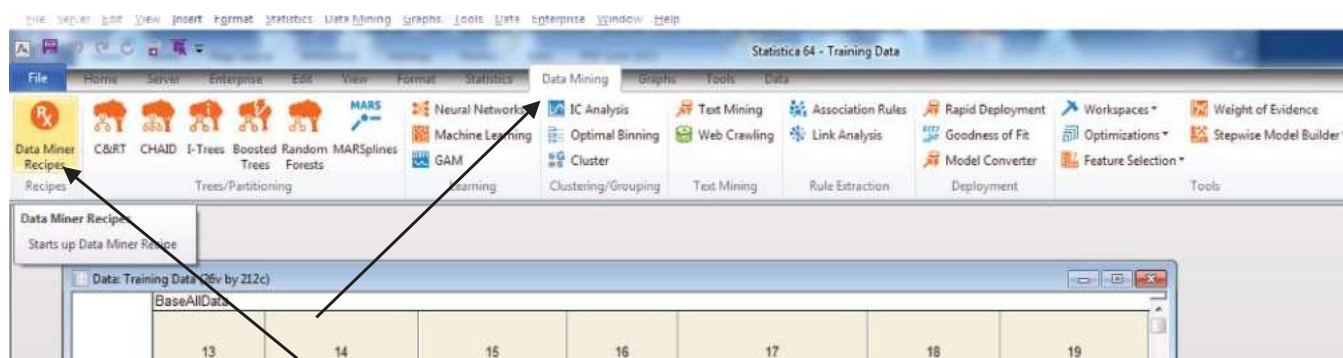


FIG. F.22 Location of Data Mining Recipe.

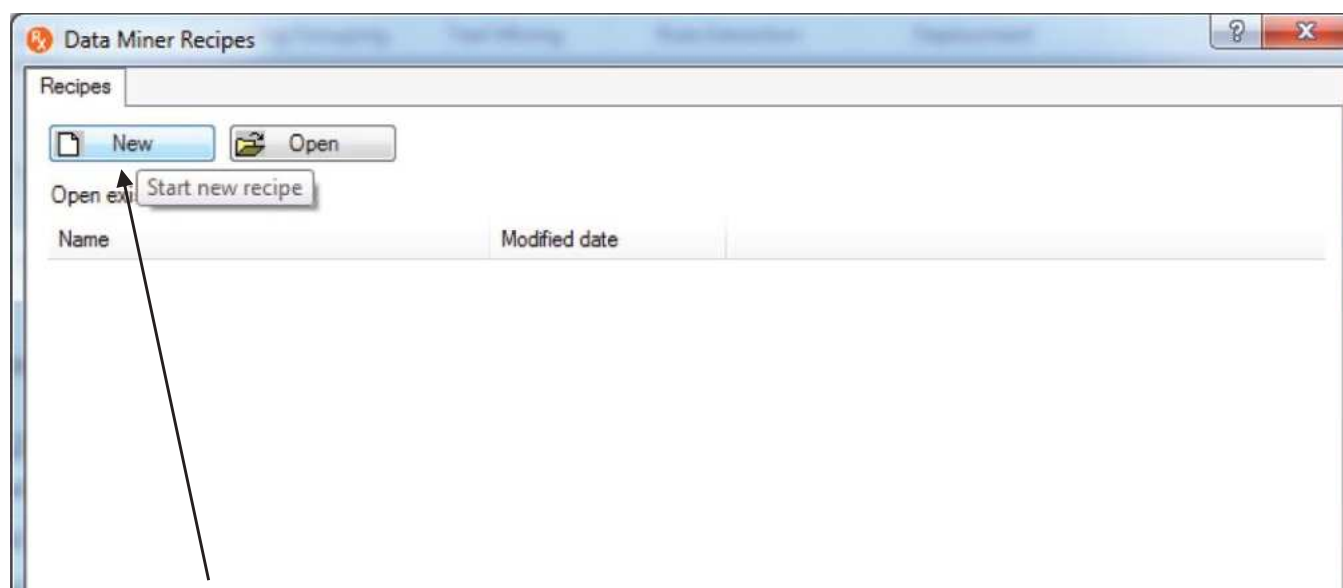


FIG. F.23 Select New.

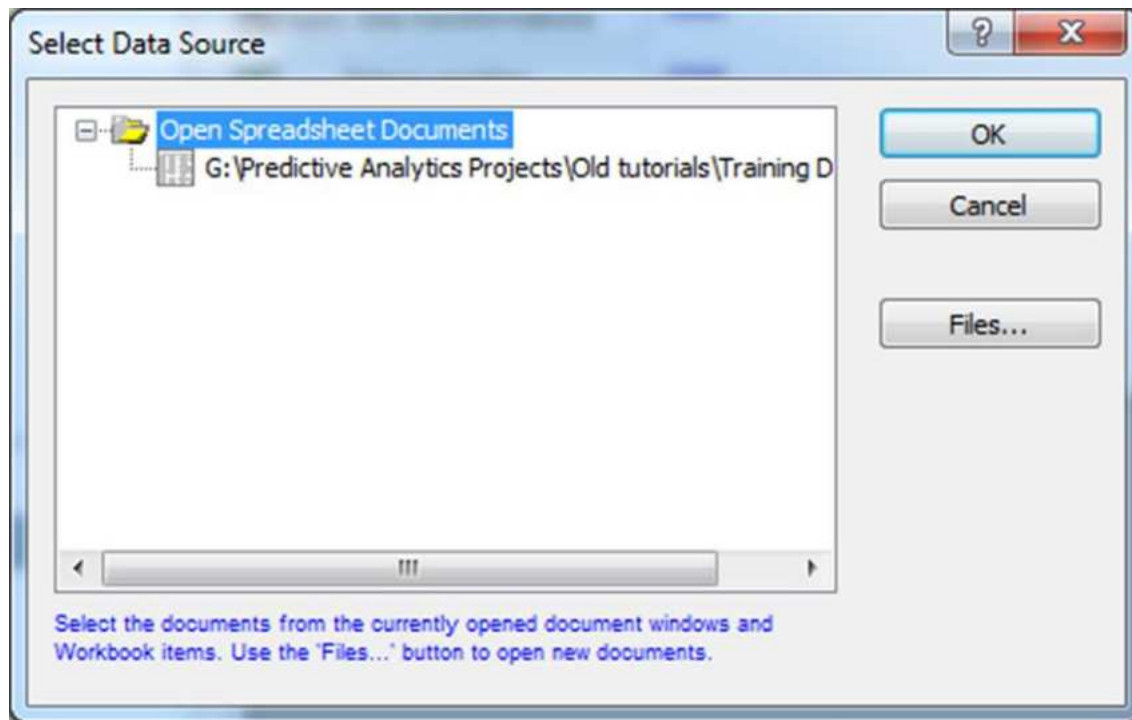


FIG. F24 Select the open file and then click OK.

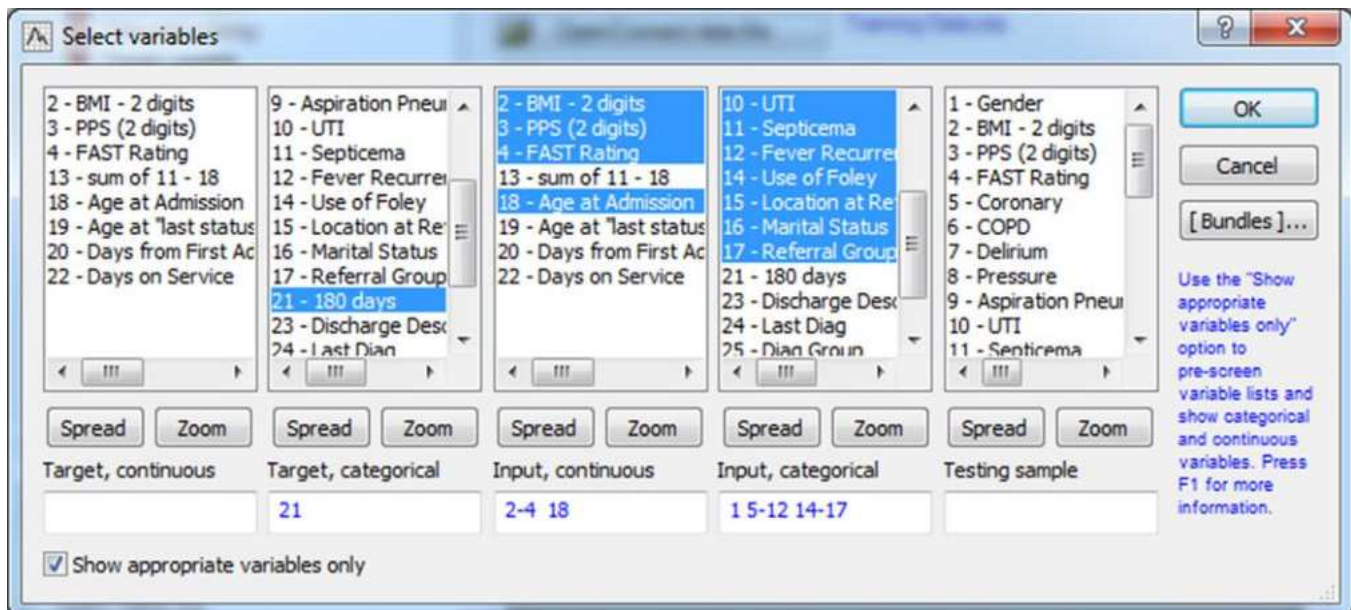


FIG. F25 Dependent variable is 21; continuous predictors are 2 to 4 and 18; categorical predictors are 1, 5 to 12, and 14 to 17.

We could have imputed the data at the beginning. The program refuses to run. So if we wish to continue, we have to do something about the missing data. Fortunately, there are ways to handle missing data within the program. Fig. F33 shows where.

Fig. F34 shows that we clicked on automatic imputations.

This program uses k-nearest neighbors as the missing data algorithm. Again, select Run to completion and let the program run.

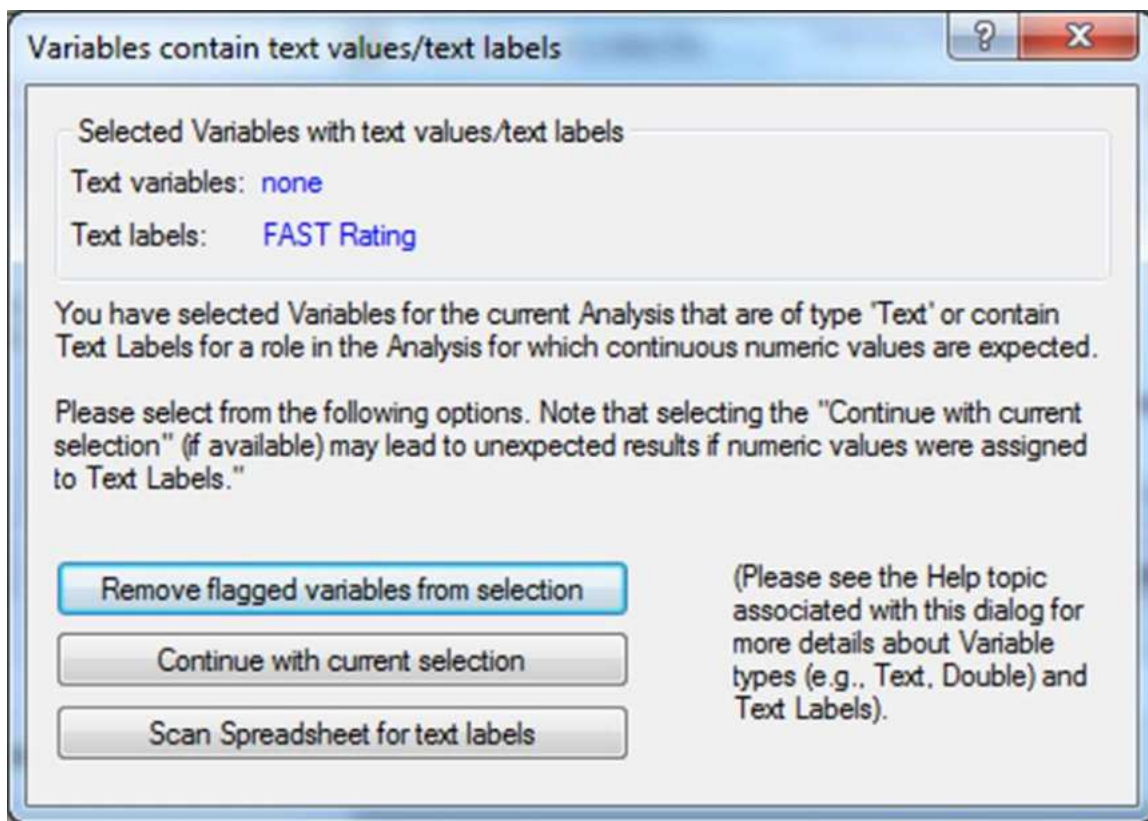


FIG. 26 Error screen that displays if the automatic function in the variables misinterprets a variable. We are treating FAST Rating as continuous, so click Continue with current selection.

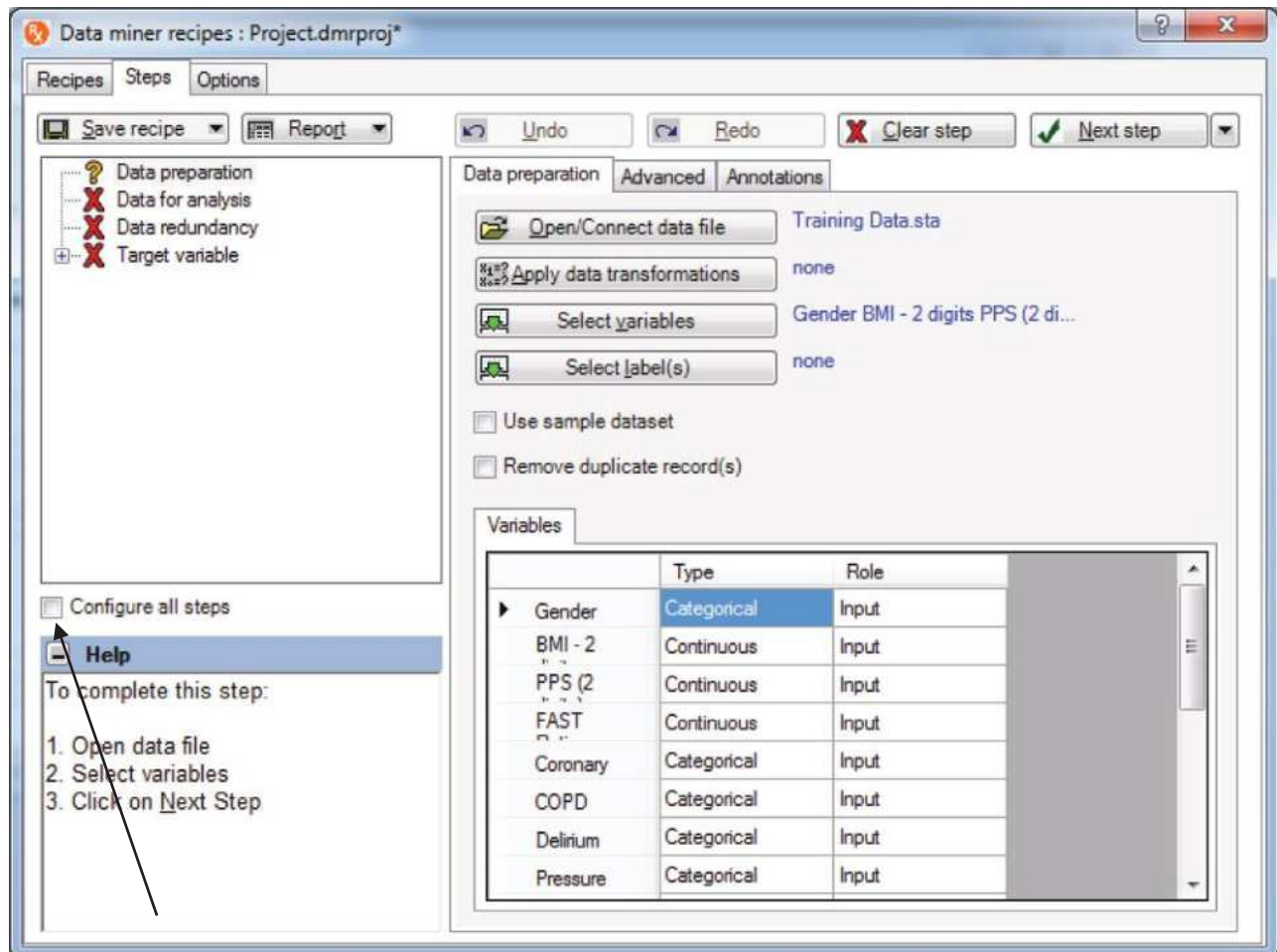


FIG. F.27 We will now check Configure all steps to make the red Xs blue.

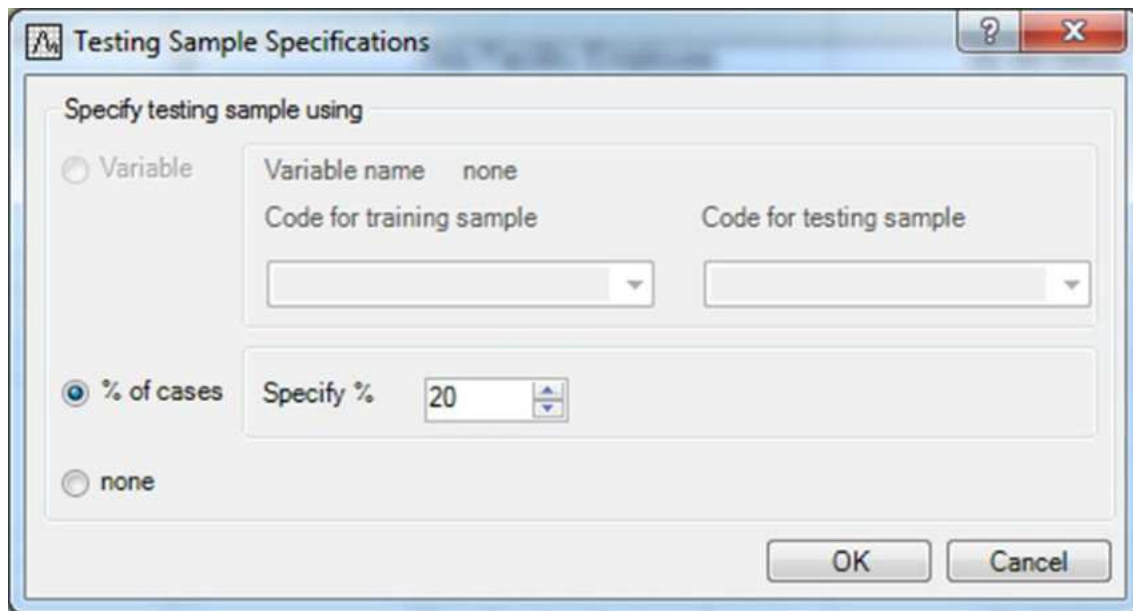


FIG. F.28 Specifying a sample of 20% selected at random.

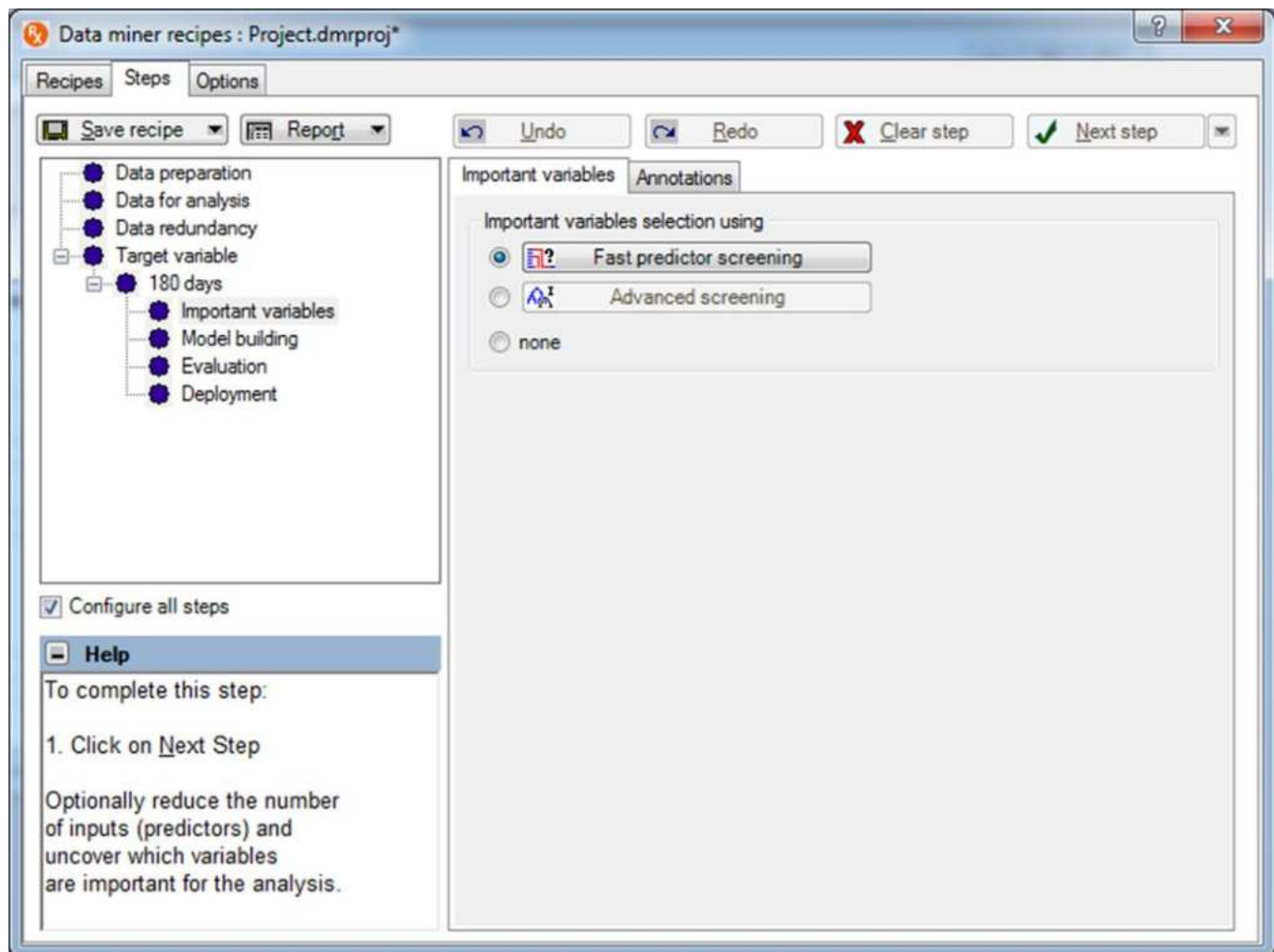


FIG. F.29 Select Fast predictor screening.

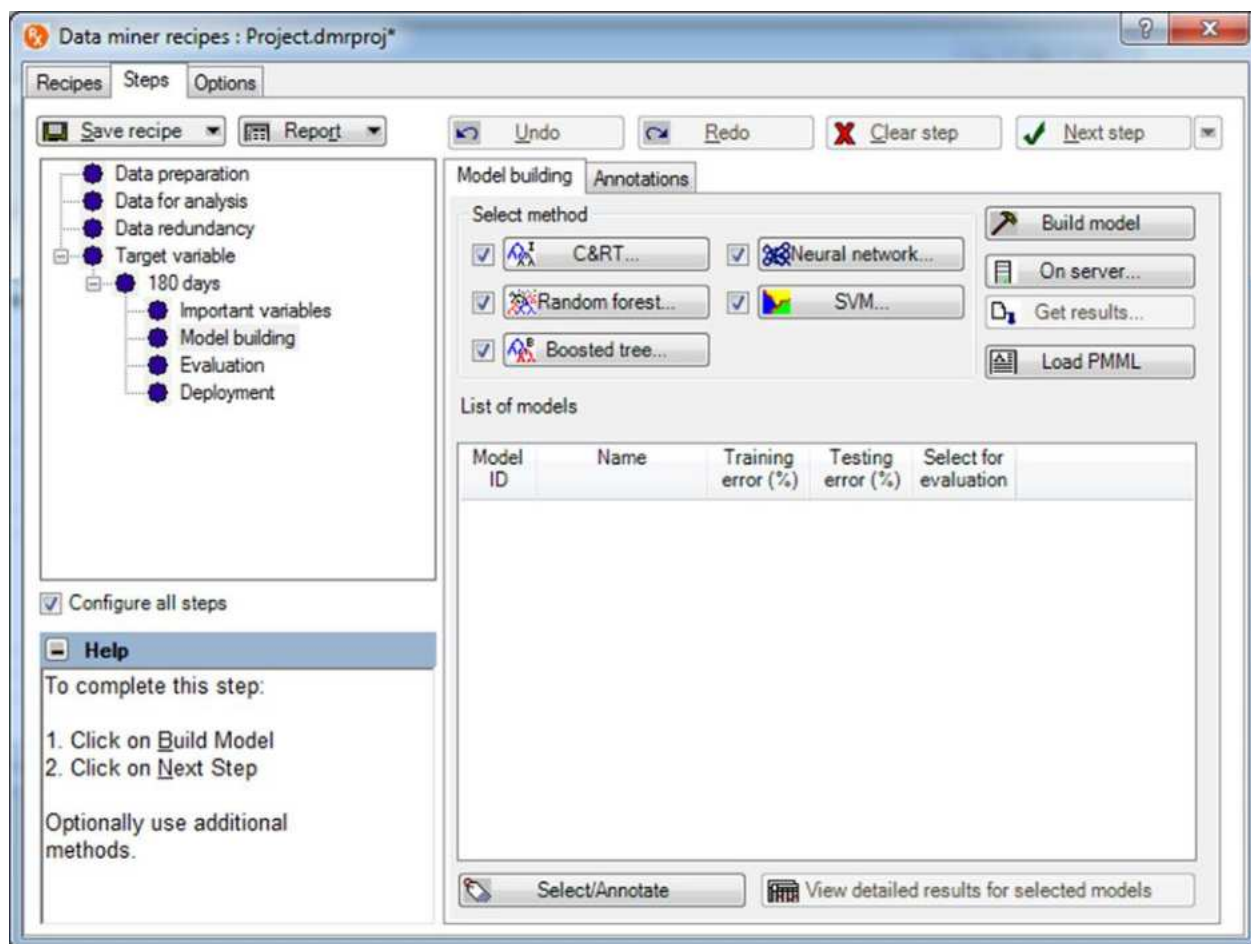


FIG. F.30 Click on all models. Note that Random forest and SVM have been checked.

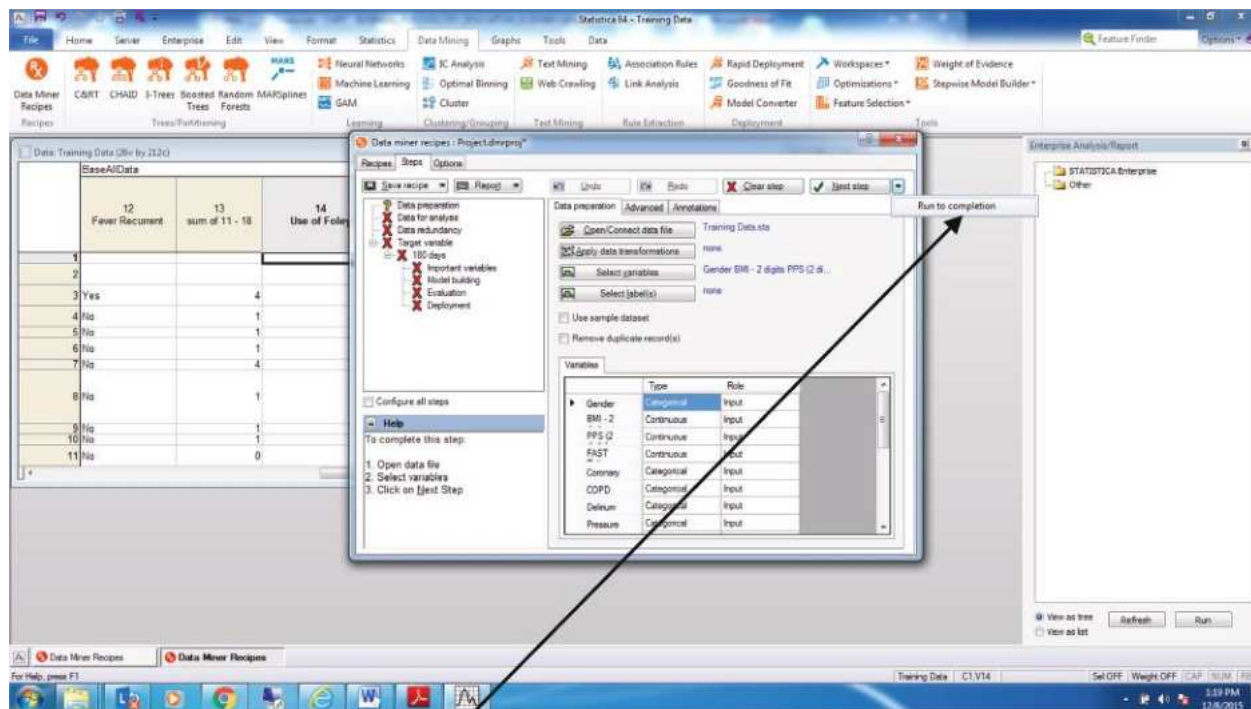


FIG. F.31 Run to completion.

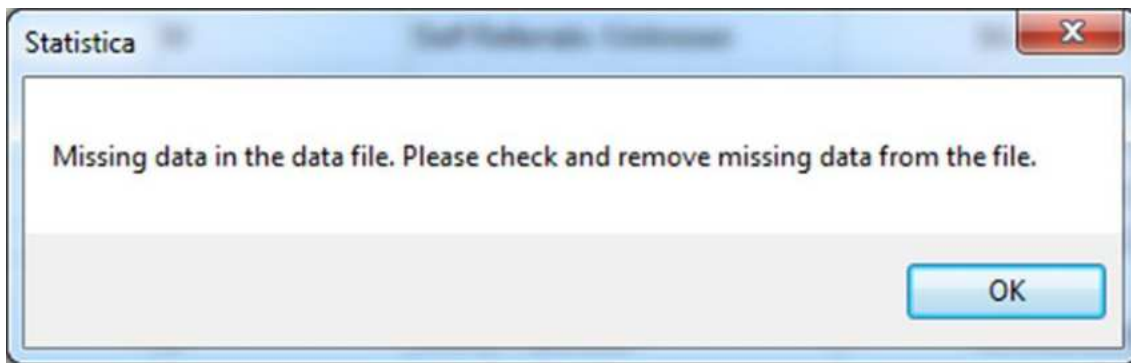


FIG. F.32 Friendly warning that we have missing data.

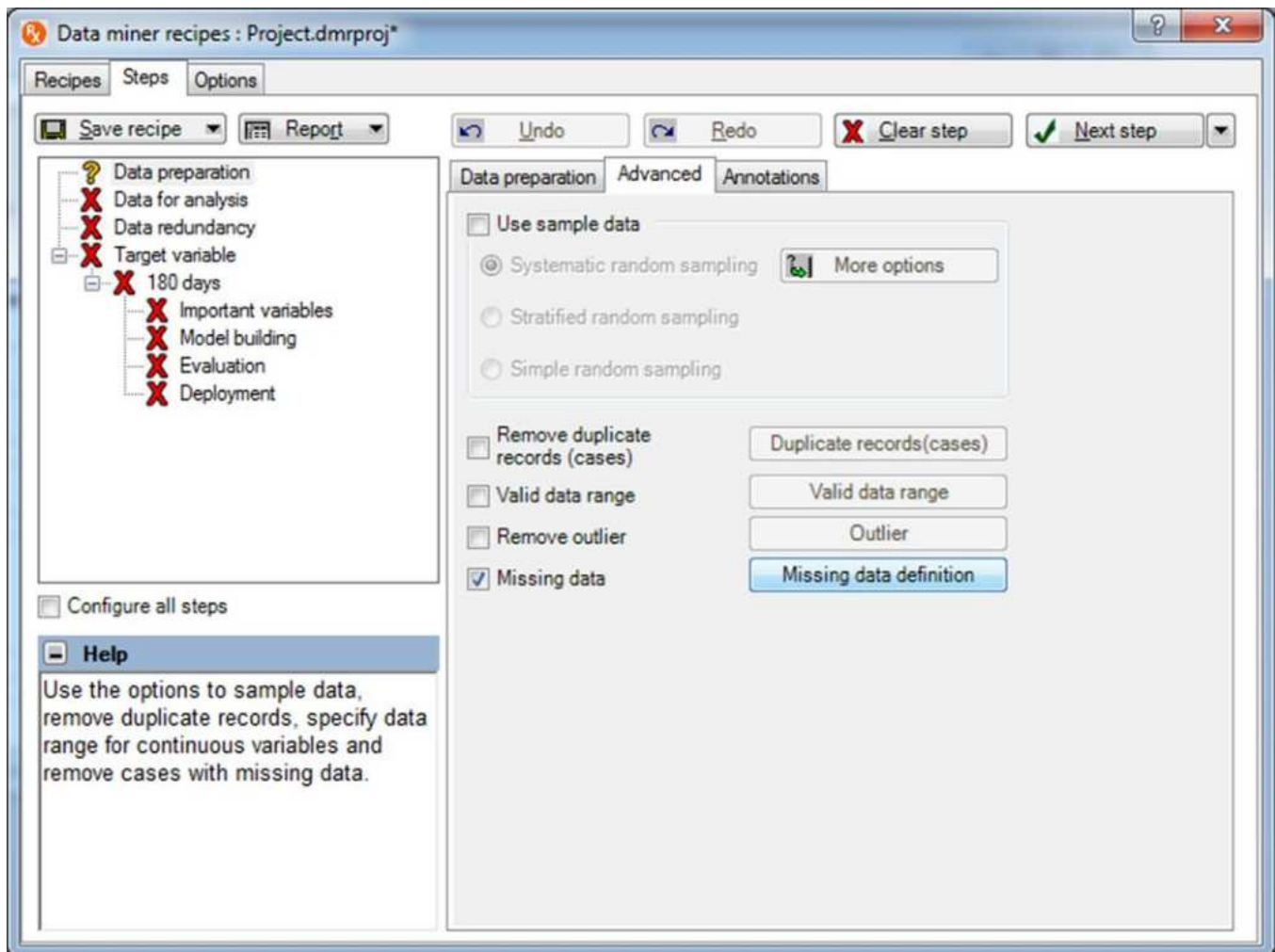


FIG. F.33 Missing data definition under the Advanced tab to deal with missing data.

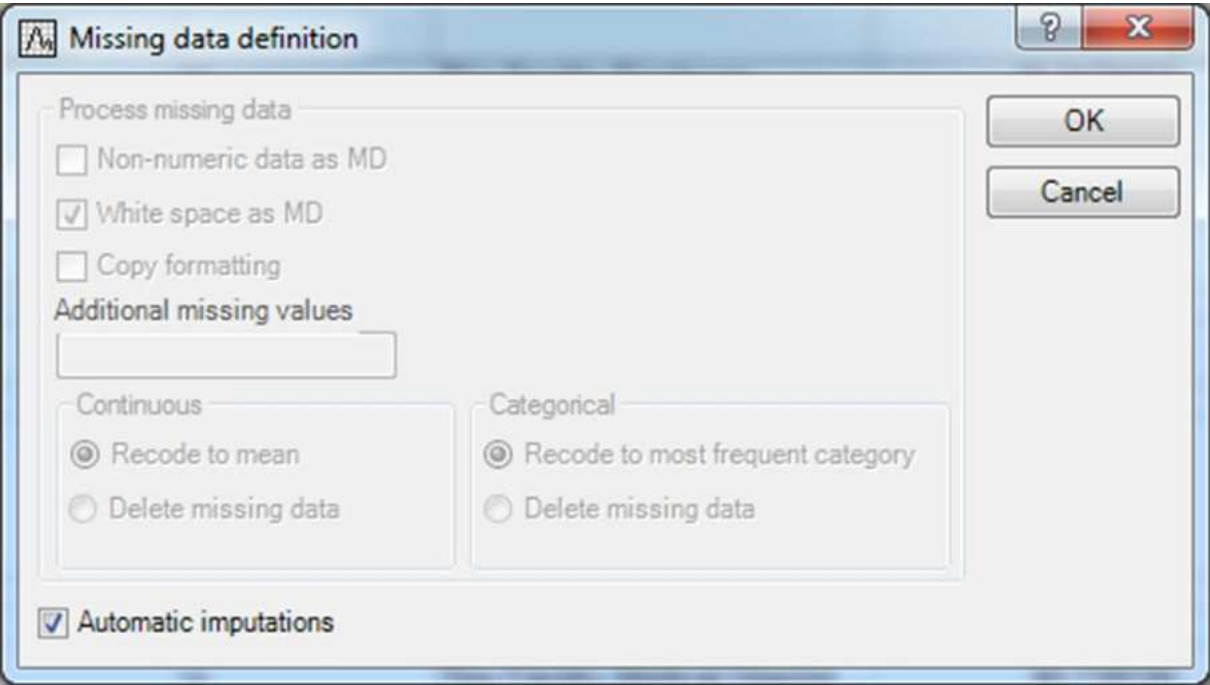


FIG. F.34 Automatic imputations.

TABLE F.4 All models show considerable error levels.

	1	2	3
Model selected for deployment	4		
Model Evaluation Summary	ID	Name	Error rate(%) (Testing Sample)
	4	SVM	30.43
	5	Neural network	32.61
	2	Random forest	39.13
	3	Boosted trees	43.48
	1	C&RT	47.83
Table	Step options		
	Date and time	12/8/2015 1:28:43 PM	

In the Reporting Documents, clicking on Evaluation of Models produces Table F.4.

We liked C&RT because it was so easy to see relationships in trees, and it was our “pet” algorithm. However, as the data mining experiment showed, C&RT in this prediction seems really quite horrible in terms of error. It is doubtful that we would do any better if we tried interactively to run C&RT, and it is doubtful that the program would predict accurately for our holdout data set.

We clicked on the important variables to see what was there – Fig. F.35.

In the Referral Group, most of the feature selection factors were still key predictors. Only the first two factors had p-values less than 0.05. We wondered what would happen if we were to reduce the variables to only two for predictors. We ran the Data Mining Recipe once again but this time, used only Referral Group, Use of Foley, FAST, and PPS. We also selected automatic imputation so that we wouldn’t be stopped by missing data.

In Table F.5, we see that C&RT did a bit better than it had, and SVM was worse.

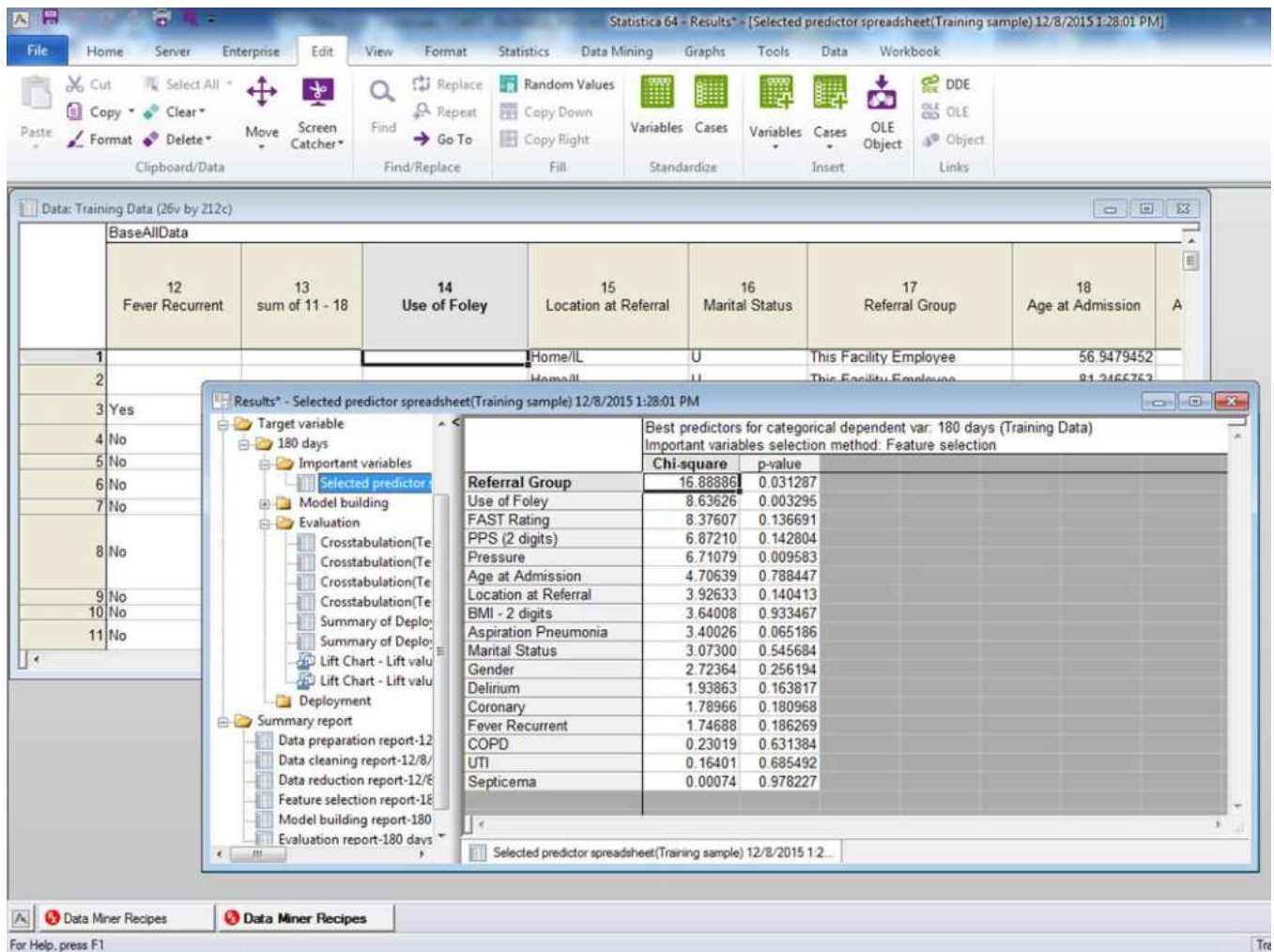


FIG. F.35 List of Important Variables.

TABLE F.5 None of the models had an error rate of less than 32.5%.

	1	2	3
Model selected for deployment	3		
Model Evaluation Summary	ID	Name	Error rate(%) (Testing Sample)
	3	Boosted trees	32.5
	5	Neural network	32.5
	1	C&RT	37.5
	2	Random Forest	40
	4	SVM	42.5
Table	Step options		
	Date and time	12/8/2015 1:44:53 PM	

TABLE F.6 Summary for Boosted trees.

Summary Frequency Table (Prediction) Table: 180 days(2) × Boosted trees Prediction(2)				
	180 days	3 Boosted trees Prediction>180	3 Boosted trees Prediction ≤180	Row Totals
Count	>180	3	8	11
Column Percent		37.50%	25.00%	
Row Percent		27.27%	72.73%	
Total Percent		7.50%	20.00%	27.50%
Count	≤180	5	24	29
Column Percent		62.50%	75.00%	
Row Percent		17.24%	82.76%	
Total Percent		12.50%	60.00%	72.50%
Count	All Grps	8	32	40
Total percent		20.00%	80.00%	

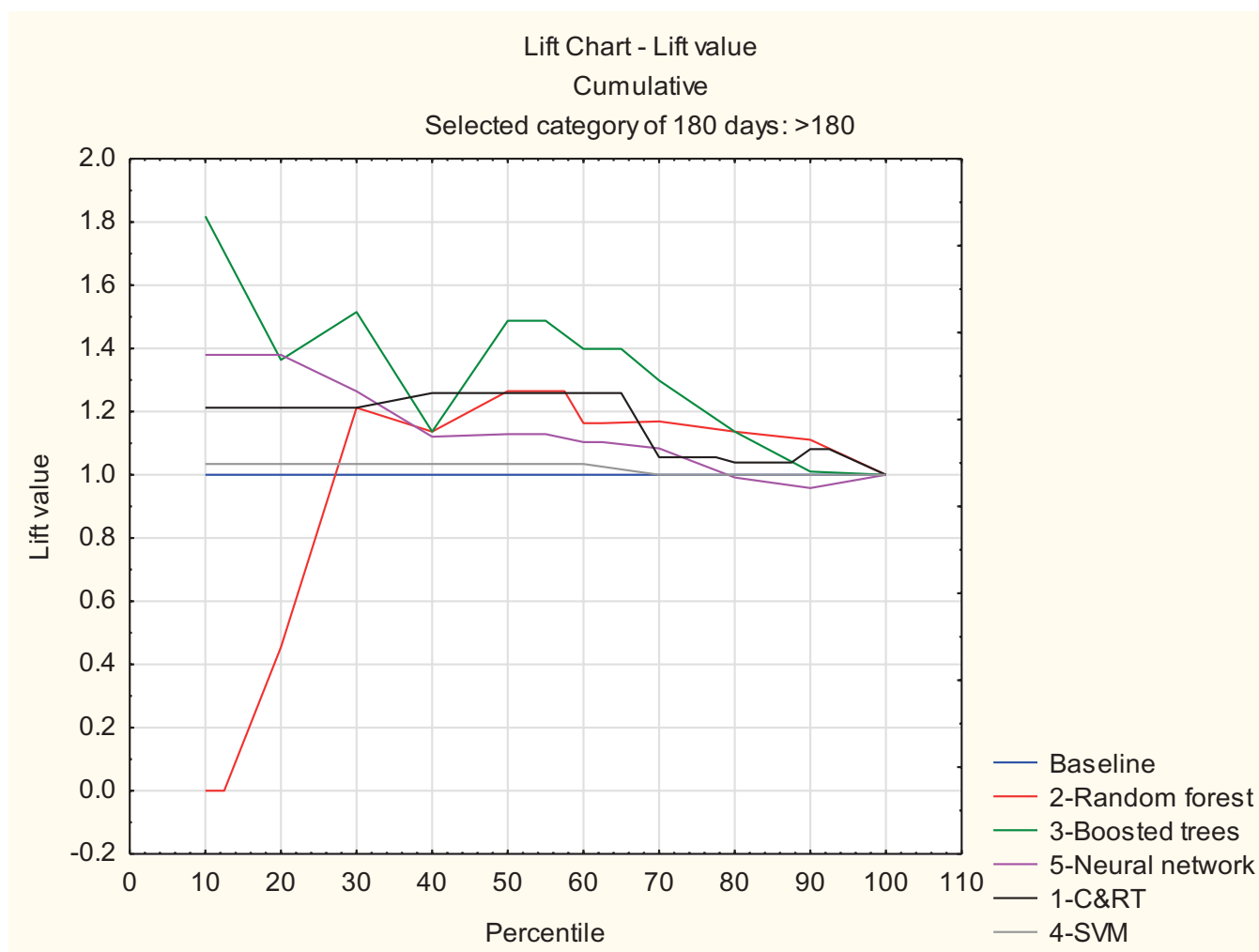


FIG. F.36 Lift chart. It seems that SVM was the best model for greater than 180 days.

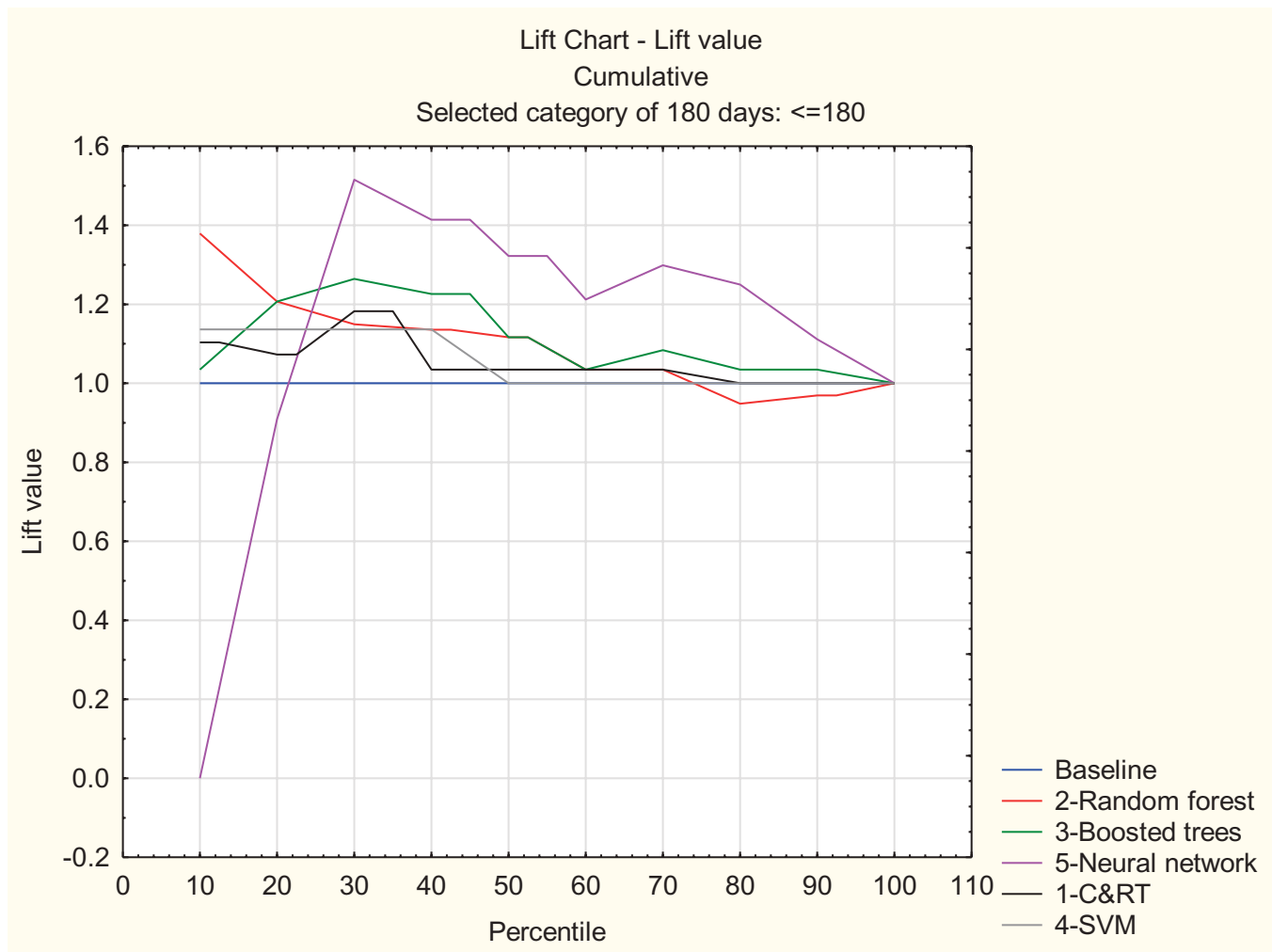


FIG. F.37 Lift chart for 180 days or less.

Boosted trees might be a good model to use interactively with the four variables.

Here is the cross-tabulation table (Table F.6) for the boosted trees from the results documents.

Predicting greater than 180 days produced the lift chart in Fig. F.36.

For 180 days or less, neural networks seemed best and then boosted trees next, as in Fig. F.37.

It was tempting to examine the trees for the C&RT, our pet algorithm. Fig. F.38 shows the training group trees.

This kind of graph is so enticing because it graphically shows what can lead to what.

Fig. F.39 shows the graphical summary of Boosted trees.

Surely this kind of graph is not as intuitive as the first one. But if Boosted trees predicts better than C&RT, then Boosted trees should be the model that is used interactively to ultimately find the best model for the data.

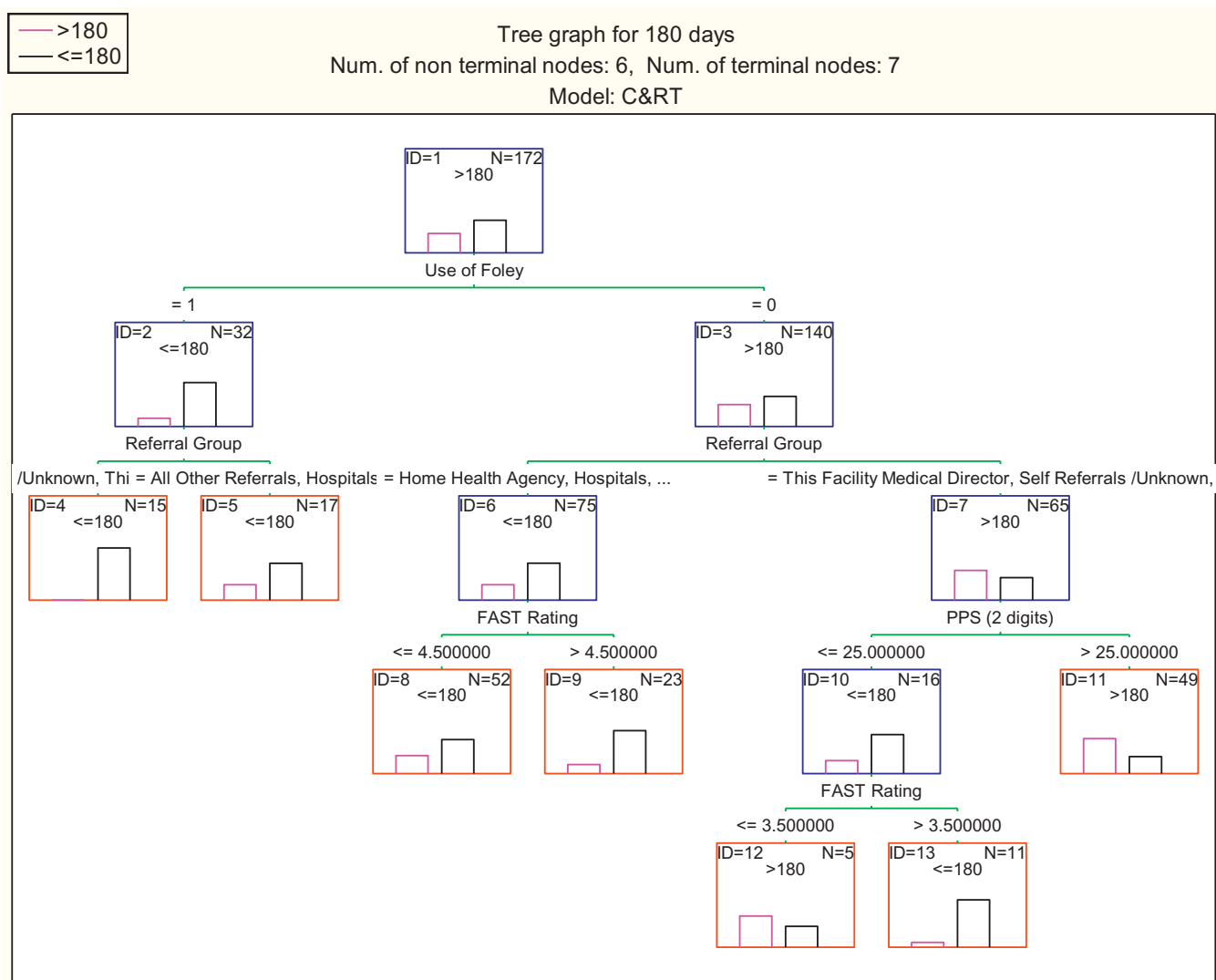


FIG. F.38 C&RT decision trees for the training data.

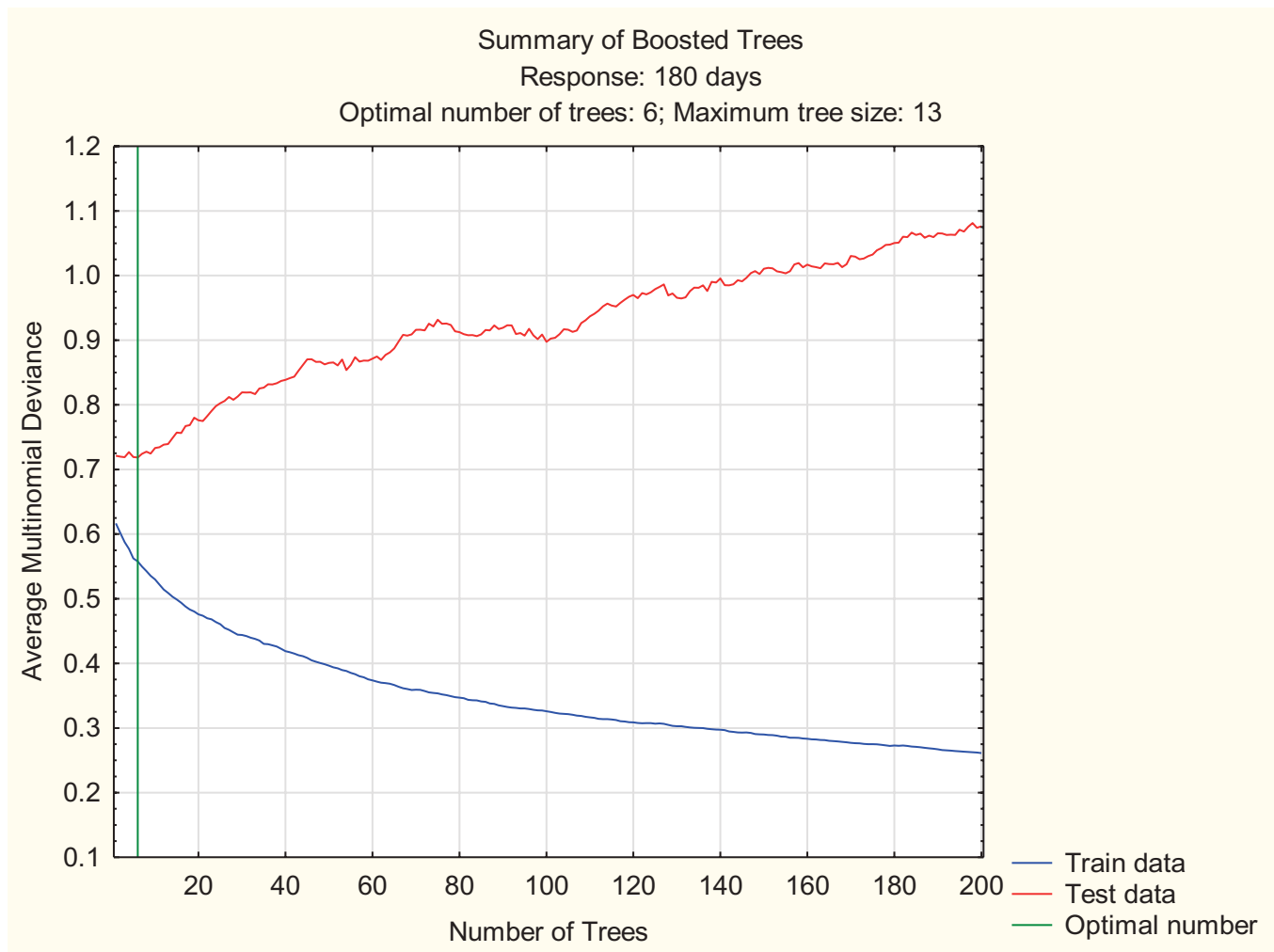


FIG. F39 Boosted trees.

G

Modeling hospice residence stay with KNIME

Linda A. Miner and Robert Nisbet

Introduction

The KNIME Analytics Platform was introduced in Tutorial D. That tutorial presented step-by-step instructions with screenshots that explain how to perform each operation. Tutorial E (and subsequent KNIME tutorials like this one – Tutorial G) assumes that the user has completed Tutorial D and has some basic skills in the use of KNIME.

This tutorial will focus on using the feature importance table output by the XGBoost Tree Ensemble Learner node. This algorithm is not a part of the basic installation. You must add the XGBoost extension to get access to it. Instructions on how to add this extension are provided below.

You will learn how to build a model without much preliminary data preparation and use various feature selection methods to prepare a data set for modeling. This tutorial will provide users with more practice in performing some basic operations in KNIME: loading data, connecting nodes, configuring nodes, and executing nodes. If it has been some time since you performed Tutorial D, go back to it and review the introduction to the basic operations. Complete explanations of some of these operations will be repeated in this and subsequent KNIME tutorials, and other operations will be documented only generally.

Tutorials H to O will serve to back-fill some common data integration and data preparation operations that may be required to be performed prior to modeling. The set of KNIME tutorials D, and G – O is presented as a sequential training course in the use of the tool to model many different outcomes, rather than being just a set of independent tutorials. This approach to learning predictive modeling was honed by many years of instruction at the University of California at Irvine by the authors of this book.

Modeling objective

In this tutorial, we will model the Tutorial_G_Alzheimer Data.csv dataset to predict the duration of stay in the hospice, using the 180 days variable as the target variable. Some of these patients were diagnosed with Alzheimer disease; some were not. Hospice administrators may want to know whether patients can be expected for either a short stay (≤ 180 days) or a long stay (> 180 days) in the hospice in order to serve them better by assigning them to appropriate rooms upon entering the facility. We will use the feature importance table output by the XGBoost Tree Ensemble Learner node to provide hospice administrators with a list of the 10 most important predictors of the expected stay of patients in the hospice.

Data

This exercise uses the Excel comma-separated values file named “Tutorial_D_Alzheimer’s Data.csv” available on the book web page. Download the data file before proceeding with Tutorial G. You may download also the Tutorial_G_Alzheimer.knwf file that contains the finished workflow. If you want to use the workflow directly, you must change the File Reader node to point to the dataset in the location on your computer where you downloaded the dataset.

Shortcuts used in the Tutorial Series

Windows shortcuts are specified only. If you want to use Apple computer shortcuts, substitute them as appropriate.

Adding an extension to KNIME

The basic installation of KNIME loads only the most commonly used list of nodes. Many nodes have been

developed by the KNIME Labs and the KNIME community of users, and some by private companies that are available for free use in KNIME.

1. Click the File tab in the top menu and select the option to Install KNIME Extensions.
2. On the next screen, fill in the top box with the string "XGBoost."
3. The next screen that displays automatically shows all extensions that contain the string XGBoost; there is only one.
4. Click on the box next to the extension to finish the configuration as shown in Fig. G.1.
5. Click the Next box.
6. Click the Next box on the following screen, showing a list of extensions to download.
7. Click the radio button to accept the download terms.
8. Click the Finish box.
9. A message at the bottom of the screen will show the download progress.
10. When the download is completed, click the Restart now box that displays.
11. KNIME will be closed, and it may reopen automatically. If not, reopen KNIME manually.
12. Now you are ready to use the XGBoost algorithm.

Creating the Tutorial G workflow

The completed workflow was not provided for Tutorial D; you had to build it yourself, following the detailed instructions. This tutorial explains how to build the Tutorial G workflow yourself using the more generalized instructions in this tutorial and by viewing the configured nodes in the workflow provided. Armed with this practice, you will be able to modify it to model other datasets for other purposes. This is a learn-by-doing exercise. Don't be satisfied by just modifying the provided workflow; you will not learn how to do it yourself, and it is quite likely that you will make mistakes in the cross-application of it for other purposes. Use the completed workflow to check on how to configure each node properly in the workflow you create yourself. Then you can say, "I did it!"

1. Download the Tutorial G Alzheimer.csv file from the book web page.
2. Download the Tutorial G Alzheimer.knwf workflow file from the book web page.
3. Open KNIME.

Note: There are no screenshots for instructions 4 to 6, because detailed instructions for completing these tasks were provided in Tutorials D and E.

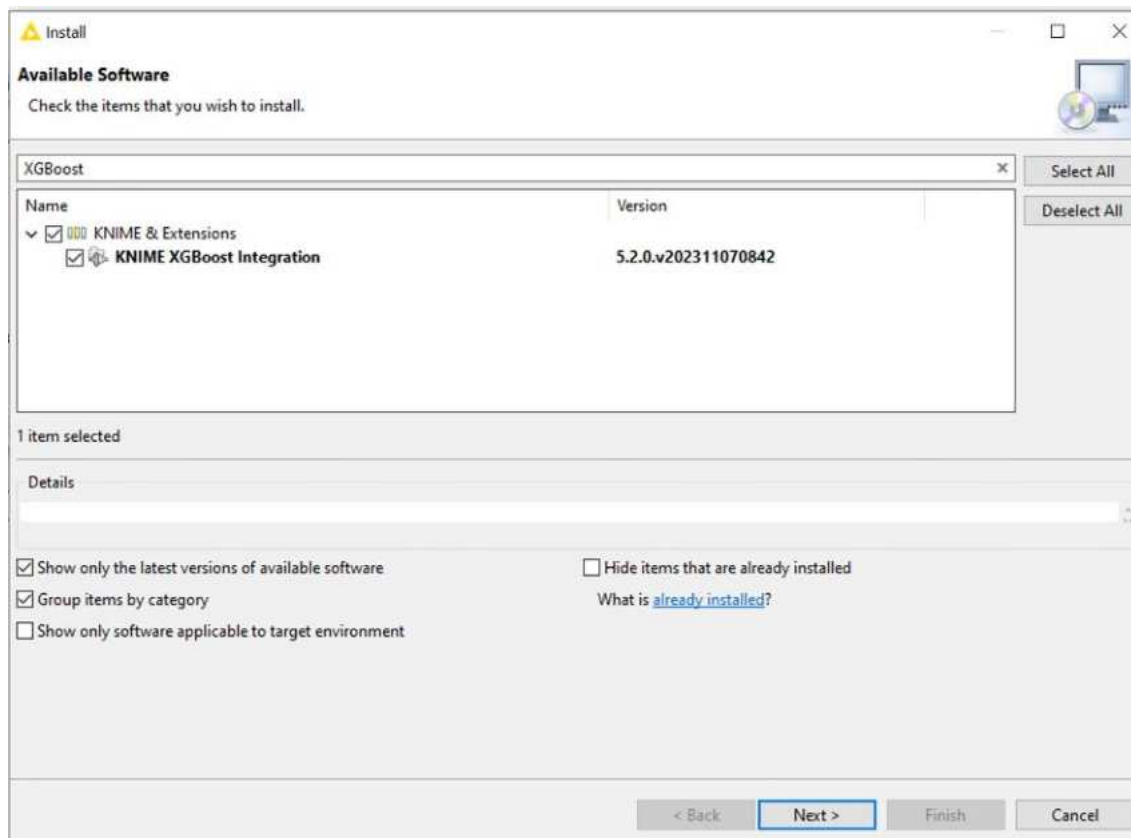


FIG. G.1 The Extension selection screen showing the KNIME XGBoost Integration box checked.

- Your KNIME Explorer listing will be different from the one in use when this tutorial was created; don't worry about it.
4. Click on the File tab on the top menu and select the New option.
 - a. Select the New KNIME Workflow Group option and click Next.
 - b. Enter "Tutorials" as the name for the new workflow group (leave the destination group as LOCAL).
 - c. Click Finish to install the new workflow group in the KNIME Explorer pane.
 5. Click on the File tab in the top menu and select the Import KNIME Workflow option.
 - a. Click the Browse box and navigate to where you saved the downloaded workflow.
 - b. Enter the destination folder in the KNIME Explorer listing where you want to store the workflow.
 6. Click on the New KNIME Workflow option.
 - a. The New KNIME Workflow option is highlighted by default; leave it so.
 - b. Click the Next box at the bottom of the screen.
 - c. Fill in the name you want to use for the new.
 - d. Click the Browse box next to the Destination of new workflow section.
 - e. Navigate to the LOCAL:/Tutorials folder.
 - f. Click on the Finish box at the bottom of the screen. You are ready to build the workflow shown in Fig. G.2. You can use it for reference while you try to build it yourself.
 7. In the Node Repository search box, enter "F."
 - a. KNIME populates the Repository listing with those nodes that begin with the letter "F."
 - b. Double-click on first node listed (the File Reader node), or click and hold on it and drag and drop it onto the modeling canvas. You can drag the node to the place on the workflow where you want to position it.
 8. Loading the dataset:
 - a. Double-click on the File Reader node to configure it.
 - b. Click on the Browse box on the upper-right of the screen
 - c. Navigate in the KNIME Explorer listing to where you stored the Tutorial G_Alzheimer Data.csv dataset and select it.
 - d. Because this file is comma delimited, KNIME loads it as such by default.
 - e. Click OK to return to the modeling canvas.
 9. Right-click on the configured File Reader node and select the Execute option to execute it.
 10. Right-click on the executed File Reader node and select the File Table option at the bottom of the list to view the dataset.
 - a. Notice that the very top row of the table shows the number of rows as 213 and the number of columns as 33.
 - b. Cursor through the data (or move the slider bar at the bottom of the screen) to the end of the rows to see the next to the last column labeled 180 days, with values ≤ 180 and > 180 in it. You can ignore the last column Sample Variable; it contains the same string "Training" in each row, signifying that this data set is a training data set.
 11. Eliminating some inappropriate variables – the Column Filter node:
 - a. Clear the search box in the Node Repository pane.
 - b. Enter "column f" in the box.
 - c. Click on the File Reader node to highlight it.
 - d. In the Node Repository screen, double-click on the Column Filter node to connect it automatically to the File Reader node.
 - e. Double-click on the Column Filter node to configure it.
 - i. Click on Pt Code in the Includes pane.
 - ii. Press and hold the Shift key and click on the "initial Dx not Alz/Dementia" variable to highlight both variables and those in between them.
 - iii. Click on the < symbol between the panes to move the variables to the Excludes pane.

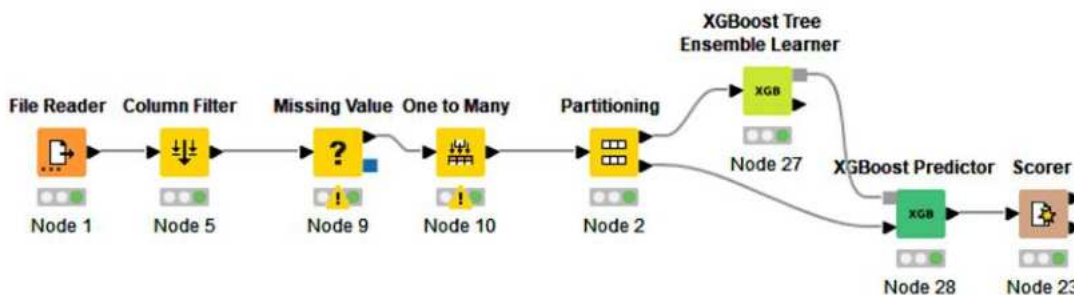


FIG. G.2 The completed KNIME workflow.

- f. Your screen should look like [Fig. G.3](#).
- g. Scroll down the Includes pane and double-click on Column 22 (another way to move variable from one pane to the other).
- h. Similarly, remove from the Includes pane: Last Status Date and Age at “last status” variables, because they do not pertain to patient admission information status.
- i. Similarly, remove Days from First Admit until Death (this variable is not known at the time of admission).
- j. Similarly, remove Sample Variable, because all rows have the same string in them, and the information is extraneous to the analysis.
- k. Check to make sure that the following variables are listed in the Excludes pane:
 - i. Pt Code
 - ii. City, State, Zip
 - iii. Flag
 - iv. initial Dx not Alz/Dementia
 - v. Column 22
 - vi. Last Status Date
 - vii. Age as “last status”
 - viii. Days from First Admit until Death
 - ix. Sample Variable
- l. Click OK to return to the modeling canvas.
- m. Right-click on the Column Filter node and select the Execute option.
(Note: From now on in the tutorial, the statement “execute” the node implies this operation.)
12. Connect a Missing Values node to the Column Filter node (following the process you used to connect other nodes).
 - a. You may remember from the File Table view for the File Reader node that there were some missing values (indicated by the “?” symbol).

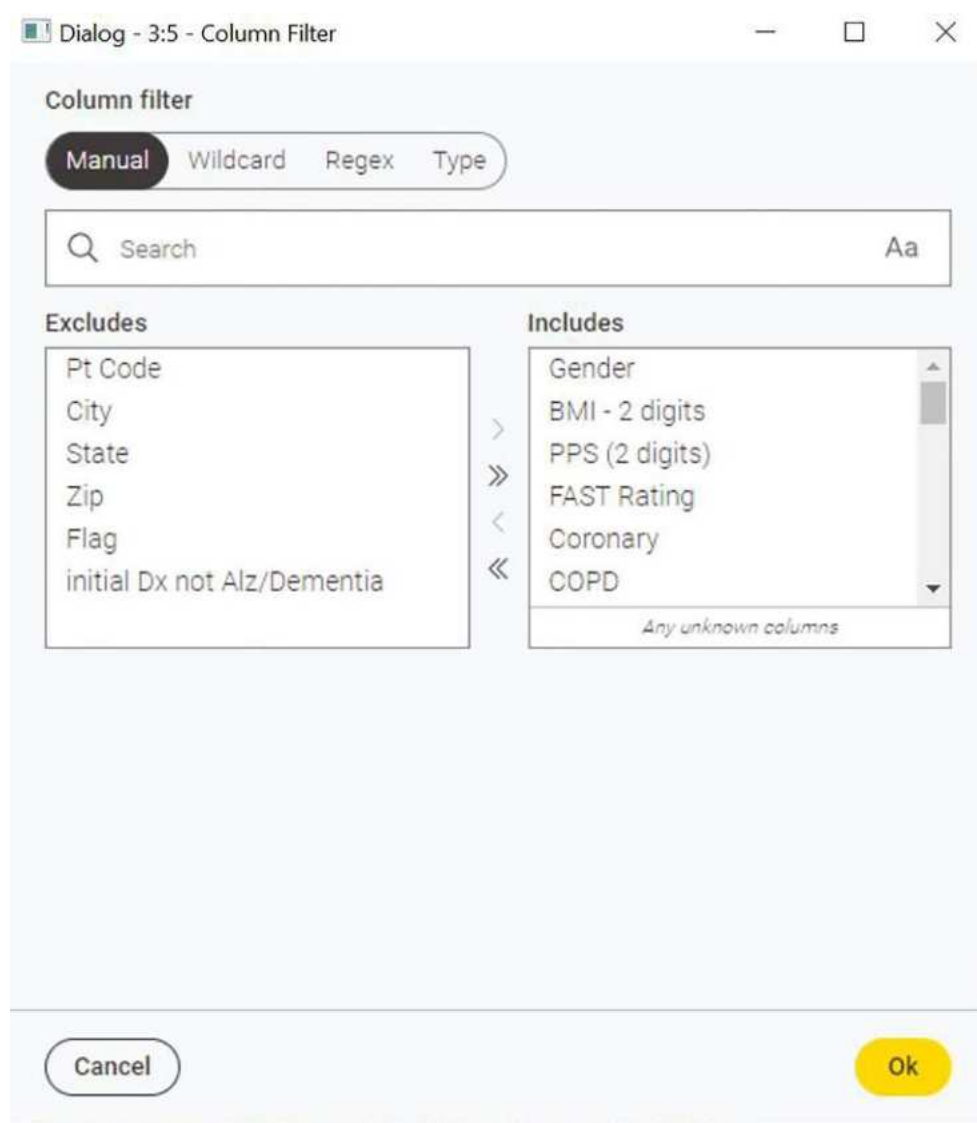


FIG. G.3 The partially configured Column Filter dialog screen.

- i. For machine learning algorithms, we must impute missing values with some value (0, or the mean value, or some other value or string), or delete the entire row.
- b. Double-click on the node to display the configuration dialog screen.
- c. For all three data types, select the Remove Row* option as shown in [Fig. G.4](#).

Note: The yellow exclamation mark can indicate an error in configuration or just a warning, as it does in this case. Move your cursor and place it on the yellow exclamation mark symbol, and KNIME will display the warning text. The warning is that the node is configured to use methods that cannot be expressed in the Predictive Model Markup Language (PMML). This is not a problem for this tutorial. It would be a problem only if you wanted to deploy this model in PMML.

13. The next operation shows how to use one of the most useful nodes in the KNIME Analytics Platform – the Many to One node. Our data has a mixture of numeric variables and categorical variables (with character strings in them). Decision tree algorithms like Gradient Boosted Trees can

build models with both input data types. They work better, however, if we make separate variables to hold the values for each variable. The One to Many node does this *automatically*! If you have hundreds of variables, this node can save you an enormous amount of time. You will have another opportunity to use the Many to One node in Tutorial K.

- a. Connect a One to Many node to the Column Filter node.
- b. Double-click on the node to configure it to exclude the 180 days variable (the Target variable we will predict; therefore we must have both values in the same variable). See [Fig. G.5](#).
- c. The new variables are called “One-hot” encoded variables or “Dummy” variables. Each dummy variable has either a “1” in it or a “0” in it. The value of 1 means that the row has the named category code in it; otherwise, it is not. Separating the categories into separate variables permits use of parametric statistical algorithms (like Logistic Regression) that require numerical values to work with datasets that have categorical (string)

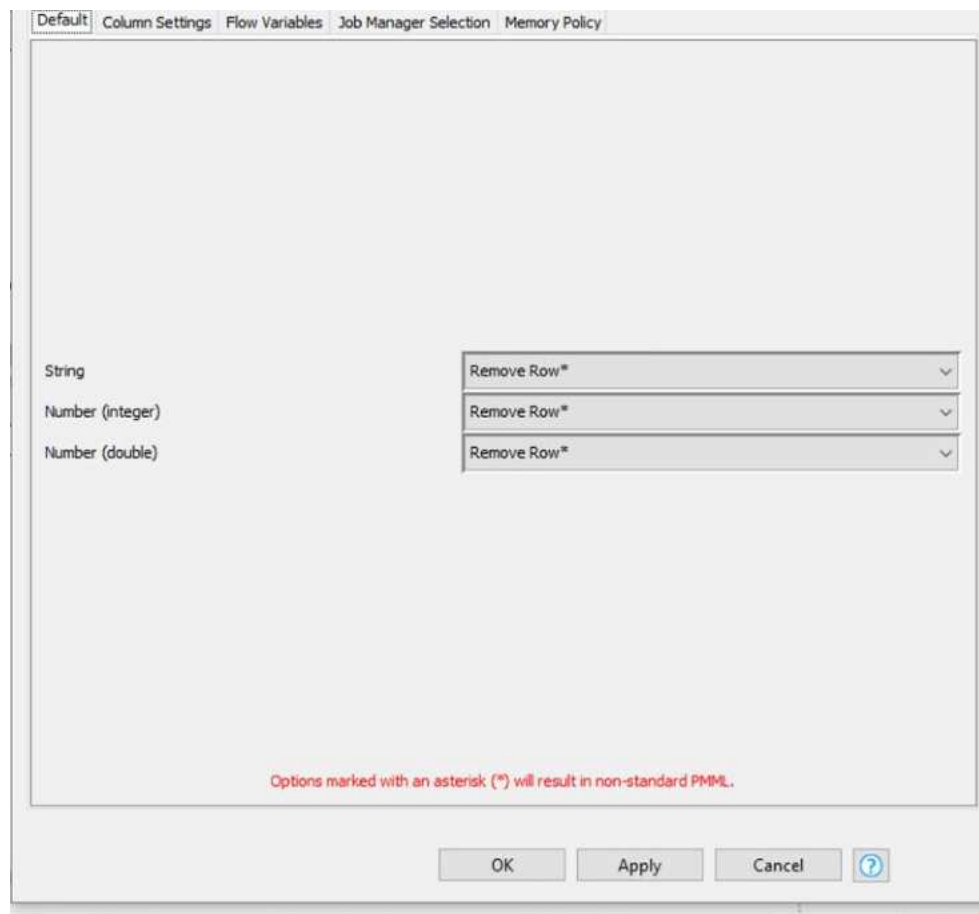


FIG. G.4 Configuration of the Missing Values node to remove rows with missing values in them.

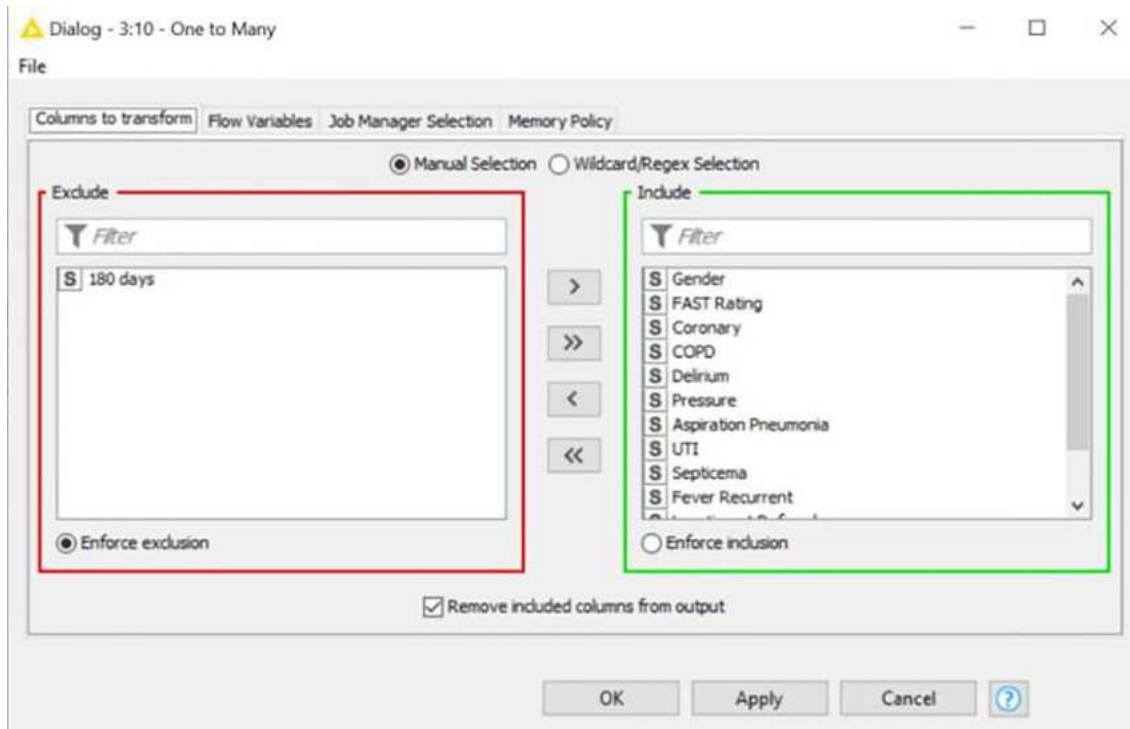


FIG. G.5 The configured One to Many node, showing the 180 days variable in the Exclude pane. (Note: This node has a warning also – duplicate category values among different variables will be distinguished by adding a suffix of the original column name to them.)

variables in them. For decision tree machine learning algorithms like XGBoost, it may have no effect, because they work with numerical variables and also with string variables. But in this case, the XGBoost algorithm appears to work better with the dummy variables than without them. You should try it both ways.

14. Connecting the Partitioning node:
 - a. Enter the character string “part” in the Node Repository search box.
 - b. See that the first node listed in the results of the fuzzy search is “Partitioning.”
 - c. Click on the One to Many node on the modeling canvas to highlight it.
 - d. Then double-click on the Partitioning node to automatically connect it to the File Reader node on the modeling canvas.
 - e. Double-click on the Partitioning node to configure it.
 - i. Confirm that the Relative[%] box for the first partition is set to 70 (a 70% sample of the rows of data in the dataset).
 - ii. Click on the Stratified sampling button and fill in the box with the 180 days variable (see Fig. G.6). This setting assures that each partition will have the same proportion of target variable categories.

- iii. Click OK to return to the modeling canvas.
- iv. Execute the Partitioning node.
- v. Make sure that the node status “traffic light” shows a green light. If not, go back and re-configure the node properly.

The XGBoost Tree Ensemble modeling algorithm

The XGBoost Tree Ensemble algorithm trains a group of decision trees with a different value of one of the modeling parameters. Processing uses a gradient descent method of optimization to find the best value of the parameter associated with the best model. The algorithm works by training a sequence of weak learners to an ensemble. Each new learner focuses on correcting the errors made by the existing ones by using the gradient descent approach.

1. Highlight the Partitioning node.
2. Remove any string in the Node Repository search box and enter <XG> in the Node Repository search string to see the XGBoost Tree Ensemble Learner and XGBoost Predictor nodes listed.
3. Double-click on the XGBoost Tree Ensemble Learner node to connect it to the Partitioning node.

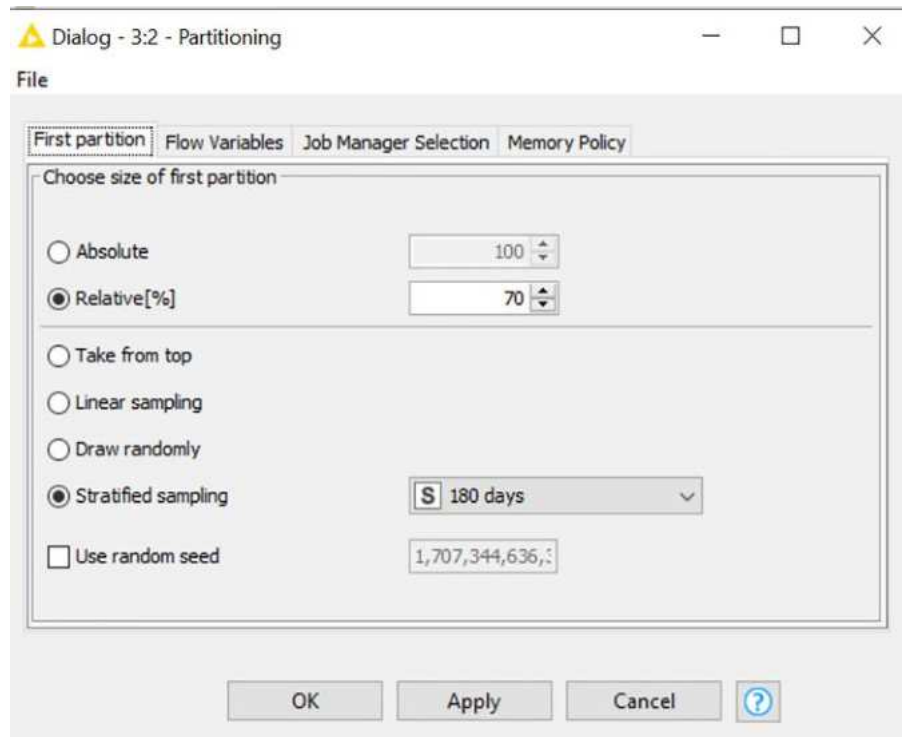


FIG. G.6 The configured Partitioning node. (Note: This stratified sampling operation makes sure that the sample subsets have the same number of rows of each data value [≤ 180 days and > 180 days].)

4. Click on the XGBoost Predictor node in the Node Repository and drag and drop it on the modeling canvas.
5. Position the XGBoost Tree Ensemble node and the XGBoost Predictor nodes on the workflow shown in Fig. G.2.

6. Connect algorithm nodes to the Partitioning node and to each other as shown in Fig. G.2.

Note: Now you can see why it was necessary to drag and drop the Predictor node onto the modeling canvas and not just double-click it to connect it automatically. The reason is that the Predictor node must be connected to two different previous nodes: (1) the lower data output port of the Partitioning node to the input data port of the Predictor node; (2) the model output port of the Learner node to the model input port of the Predictor node.

7. Double-click on the XGBoost Tree Ensemble Learner node to confirm that the 180 days variable is selected as the Target column and all of the other columns are listed in the Include pane. There is no need to configure the Predictor node for this tutorial.
8. Right-click on the XGBoost Predictor node and execute it.

Notice that the Gradient Boosted Trees Learner node is executed also. This is another way to execute nodes. When you execute a node in a workflow, it

checks to see if all previous nodes have been executed and, if they have not, KNIME executes them in order.

Model evaluation

The last node to connect is the Scorer node, which is used to calculate various prediction accuracy metrics.

1. Clear the Node Repository search string, and enter “scorer” into the cleared box.
2. Click on the Gradient Boosted Predictor node to highlight it and double-click on the Scorer node in the Node Repository listing.
3. Now, the workflow on your screen should look like the workflow in Fig. G.2.
4. Double-click on the Scorer node to configure it.
5. Select the 180 days feature for the first column and leave the second column as it is.
6. Click OK to return to the modeling canvas.
7. On the top menu of the KNIME interface, click on the gray button with two white triangles in it. This operation will execute all executable nodes in the workflow in sequence.

Note: This is another way to execute nodes. Clicking the gray button with the single white triangle in it will execute only the selected (highlighted) node

on the workflow. You can use these buttons to execute each node as you build the workflow, instead of right-clicking on the node and selecting the Execute option on the option screen.

8. Right-click on the executed Scorer node and select the Accuracy statistics option to see [Table G.1](#), showing the results of the evaluation operation.

Discussion of model accuracy results

The Row ID column in [Table G.1](#) labels to which target category the accuracy results in the columns to the right pertain. We are most interested in the ≤ 180 category prediction results. The next four columns (TruePositives, FalsePositives, TrueNegatives, and FalseNegatives) are populated with values from the confusion matrix output table of the Scorer node. We will consider only the Sensitivity and Specificity accuracy measures in this tutorial.

Sensitivity: the accuracy of predicting the positive value of a given target category correctly.

Specificity: the accuracy of predicting the negative value of the target category correctly.

For the Target category of interest (≤ 180), the Sensitivity value is 0.744 and the Specificity value is 0.409. Those numbers mean that the model predicts actual values for the ≤ 180 target category moderately well, but predictions include a lot of false-positive predictions (>180 values predicted as ≤ 180). This means that hospice attendants can ID most of the short-term stay patients moderately well, but there are a lot of false

predictions in the results. Hospice management can use the model to give them only a general indication of the expected stay length of new patients.

A more complete discussion of the Sensitivity and Specificity metrics along with formulas for calculating them is presented in [Chapter 15](#) – Model Evaluation and enhancement.

Feature importance

Hospice administrators might want to know what features are most important in the model. The XGBoost Learner node provides an output table that calculates “Gain,” which is a measure of the information gain associated with the inclusion of each feature in the model.

1. Right-click on the XGBoost Tree Ensemble Learner node and select the XGBoost Feature Importance.
2. Click on the Gain column header, and select the Sort Descending option to see a list of features in descending order of Gain in [Table G.2](#).

This tutorial shows you how to do a complete modeling project with the KNIME Analytics Platform. Tutorials D and E provide more detailed instructions for performing basic KNIME tasks such as finding nodes, connecting nodes, executing nodes, and interpreting the output results of KNIME nodes. Tutorials H to O will as a group take a step backward in the predictive analytics process described in [Chapter 3](#) and provide additional instructions for accomplishing a number of common data preparation tasks that some data sets require before building models with them.

TABLE G.1 The Accuracy statistics table output by the Scorer node.

Row ID	I TruePo...	I FalsePo...	I TrueNe...	I FalseN...	D Recall	D Precision	D Sensitivity	D Specificity	D F-meas...	D Accuracy	D Cohen'...
≤ 180	29	13	9	10	0.744	0.69	0.744	0.409	0.716	?	?
>180	9	10	29	13	0.409	0.474	0.409	0.744	0.439	?	?
Overall	?	?	?	?	?	?	?	?	?	0.623	0.157

Note: Your values may be slightly different from those in [Table G.1](#) due to random selection differences in your workflow compared to the downloaded workflow.

TABLE G.2 A portion of the Feature Importance output table of the XGBoost Tree Ensemble Learner node, showing the top 10 most important predictor features.

Row ID	S Feature Name	D Weight	D ▼ Gain	D Cover	D Total Gain	D Total C...
Row48	All Other Referrals_Referral Group	14	0.717	20.202	10.039	282.824
Row3	Use of Foley	32	0.603	16.216	19.29	518.911
Row28	No_Septicema	2	0.488	13.931	0.975	27.862
Row6	Admit_Status	76	0.45	20.866	34.209	1,585.794
Row47	Family/Friend_Referral Group	10	0.416	21.063	4.164	210.633
Row43	All other physicians_Referral Group	2	0.384	24.605	0.767	49.21
Row4	Residence Description (not location at referral)	80	0.382	17.662	30.588	1,412.958
Row10	7f_FAST Rating	4	0.374	15.708	1.497	62.832
Row0	BMI - 2 digits	50	0.345	12.219	17.254	610.941
Row22	No_Pressure	16	0.278	11.167	4.449	178.665

H

Data preparation: Merging data sources

Roberta Bortolotti, Senior Director, Nasdaq

An important part of the initial data preparation efforts consists of tasks of data acquisition and data integration. Data acquisition often can be stressful, because the stakeholders of the dataset may require a rather extensive set of security steps, sometimes from different sources, before data can be accessed. Assuming that four datasets are available for use, this tutorial walks through a task to integrate data found in multiple data sources. Here, it is assumed that the datasets are in a flat-file format. The data sources are combined into a suitable structure where all variables are fields in a record. The result of this tutorial will be one single file containing all variables and their relevant data to be input into a mining tool.

1. Download the following datasets from the book Web page and store them in your Data folder for tutorials, because you will use all four of these datasets in this tutorial:
 - a. Donors2.csv
 - b. Demographics.csv
 - c. Donors_Hist.csv, and
 - d. Promo_Hist.csv
2. Open **KNIME**.
3. Click on **File** → **New** on the top bar menu to create a new workflow, as shown with the blue highlight in Fig. H.1.
4. In the Wizard window, select **New KNIME Workflow** option, and click **Next** (Fig. H.2).
5. In the next screen, shown in Fig. H.3, name the new workflow **Tutorial-H_Data-integration**. Click on **Browse** to specify a Tutorial folder, if necessary, and click **Finish**.
6. The new workflow will display in KNIME explorer as an active workflow with a green dot in the folder you specified (Fig. H.4). The green dot means that it is ready for editing.
7. On the **Node Repository** section, expand the **IO** → **Read** category of nodes, and drag the **CSV Reader** node to the workflow space (or double-click on the **CSV Reader** node and move it to position it on the workflow), as shown in Fig. H.5. Alternatively, you can enter “**csv**” in the Node Repository search box, and only those nodes that have “**csv**” in their names will display.
 8. Right-click on the **CSV Reader** node and select **Configure** (or just double-click on the node). Or double-click on the **CSV Reader** node to display the *Configuration Dialog* screen.
 - a. Click on **Browse**, navigate to the Data folder where you stored the tutorial datasets, and select **Donors2.csv** file.
 - b. Under **Reader Options**, check the box labeled “**Has column header**,” as shown in Fig. H.6.
 - c. Confirm that the “**Has column header**” box is checked.
 - d. Click on the **Advanced Settings** tab, and uncheck the box for **Limit data rows scanned**.
 - e. Click **OK** to return to the modeling canvas.
 9. Click on the **CSV Reader** node to select it (a box appears around the node).
 10. Execute the node. There are three ways to execute a node:
 - a. By clicking on the execute icon on the toolbar (shown by the red arrow in Fig. H.7);
 - b. By right-clicking on the node and selecting the “**Execute**” option;
 - c. By pressing the **F7** function key to execute the selected node.
 11. After successful execution of the node, the yellow status light turns to green. Double-click on the node label and name it *Donors*, as shown in Fig. H.8.
 12. On the **Node Repository** section, expand the **Manipulation** → **Column** → **Split & Combine** node, and either:

enter “**csv**” in the Node Repository search box, and only those nodes that have “**csv**” in their names will display.

The **CSV Reader** node is used to read data from an ASCII file or a URL location. The node can also be configured to read from various file formats.

8. Right-click on the **CSV Reader** node and select **Configure** (or just double-click on the node). Or double-click on the **CSV Reader** node to display the *Configuration Dialog* screen.

a. Click on **Browse**, navigate to the Data folder where you stored the tutorial datasets, and select **Donors2.csv** file.

b. Under **Reader Options**, check the box labeled “**Has column header**,” as shown in Fig. H.6.

c. Confirm that the “**Has column header**” box is checked.

d. Click on the **Advanced Settings** tab, and uncheck the box for **Limit data rows scanned**.

e. Click **OK** to return to the modeling canvas.

The “traffic light” symbol below the node will display a yellow light, meaning that it is ready for execution. The symbol is the status indicator of the node.

9. Click on the **CSV Reader** node to select it (a box appears around the node).

10. Execute the node. There are three ways to execute a node:

a. By clicking on the execute icon on the toolbar (shown by the red arrow in Fig. H.7);

b. By right-clicking on the node and selecting the “**Execute**” option;

c. By pressing the **F7** function key to execute the selected node.

11. After successful execution of the node, the yellow status light turns to green. Double-click on the node label and name it *Donors*, as shown in Fig. H.8.

12. On the **Node Repository** section, expand the **Manipulation** → **Column** → **Split & Combine** node, and either:

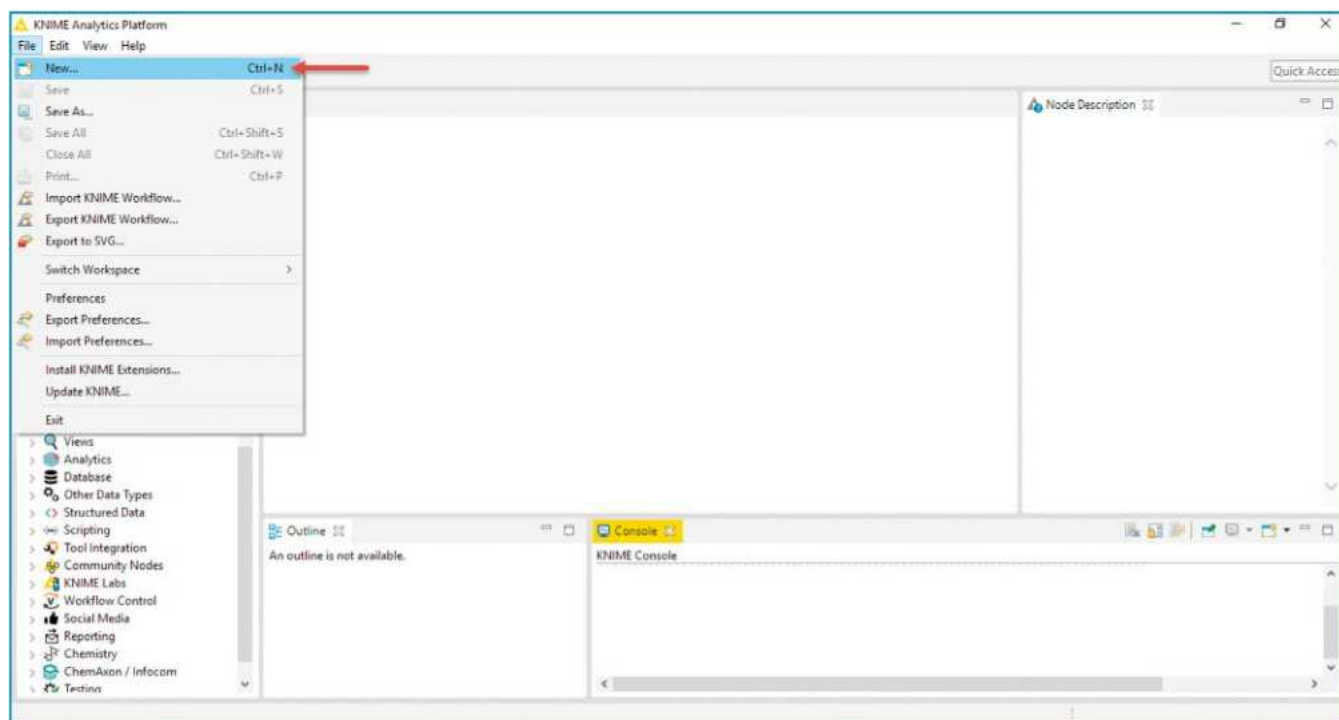


FIG. H.1 How to open a new workflow.

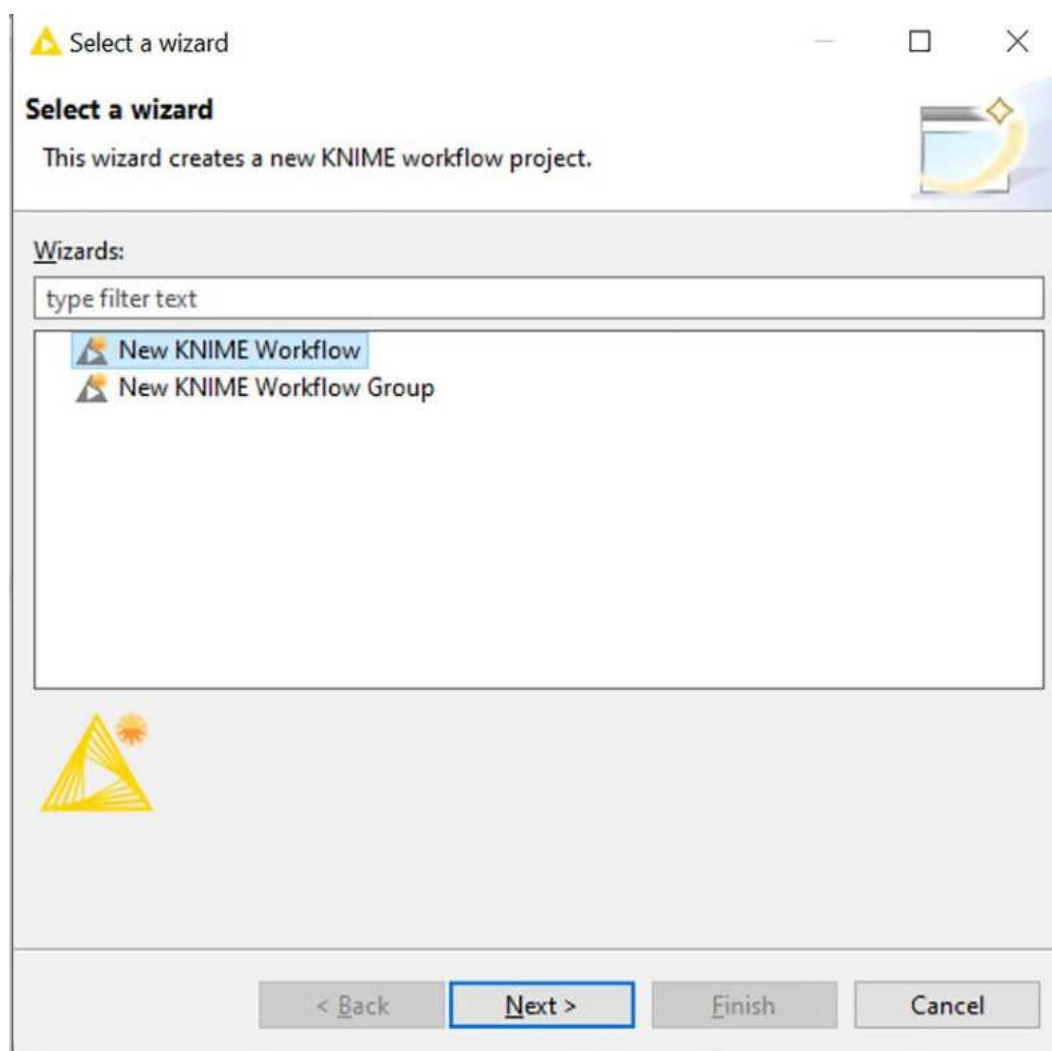


FIG. H.2 Option to create a new KNIME workflow.

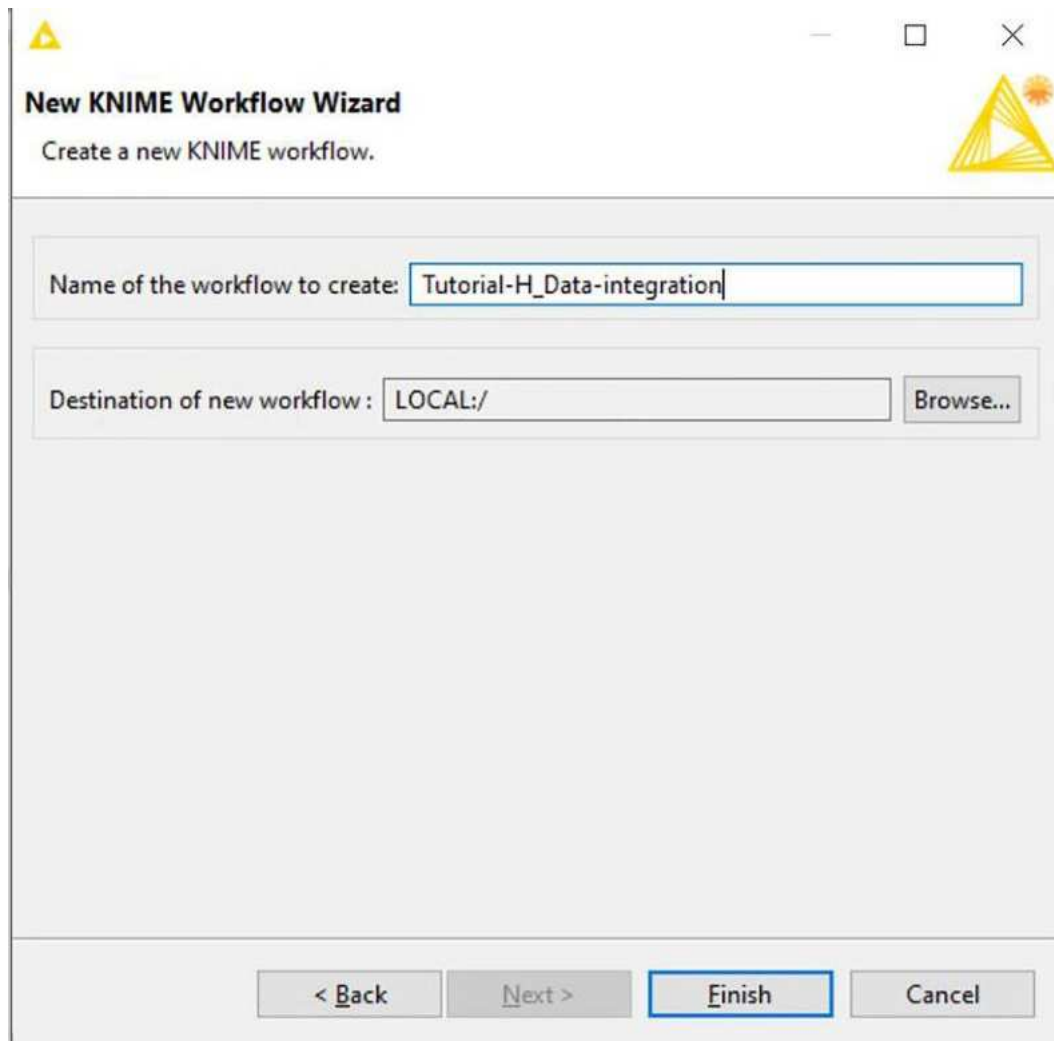


FIG. H.3 Naming the new workflow.

- a. Select **Joiner** node, drag it to the workflow, and connect it to the **CSV Reader** node; or
 - b. Enter “**joiner**” into the search box in the **Node Repository**, highlight the **CSV Reader** node, and double-click on the **Joiner** node to connect it automatically with the **CSV Reader** node. This option was followed to produce Fig. H.9.
13. Repeat steps 5 to 11 to add another **CSV Reader** node using the **Demographics.csv** file for “**Input location:**” in the *Configuration Dialog* screen. Rename the **CSV Reader** node as *Demographics*. Configure the node as you did the *Donors2 CSV Reader* node.
 - a. Check the box for **Has column header**.
 - b. Uncheck the box in the **Advanced Settings** tab for **Limit rows scanned**.
14. Connect the new **CSV Reader** node to the bottom triangle of the **Joiner** node. To connect nodes manually, click and hold the mouse button on an output port on the right side of a node and draw a connecting arrow to an input port on the left side of another node. In this way, the **Joiner** node connects to two tables, as shown in Fig. H.10, based on a common joining column of each table as specified in the *Configuration Dialog* screen.
15. Right-click on the **Joiner** node and select **Configure**.
 - a. Click on the “**Add matching criterion**” box to identify the row that contains the row IDs.
 - b. Click on the down caret in the **Row ID** box of the **Top input** and select the **RECNO** variable as the join key.
 - c. Click on the down caret in the **Row ID** box of the **Bottom input** and select the **RECNO** variable as the join key.
 - d. Fig. H.11 shows the configured screen of the **Joiner** node.
 - e. Under the **Joiner Settings** tab of the *Configuration Dialog* screen, notice that only the “**Matching**

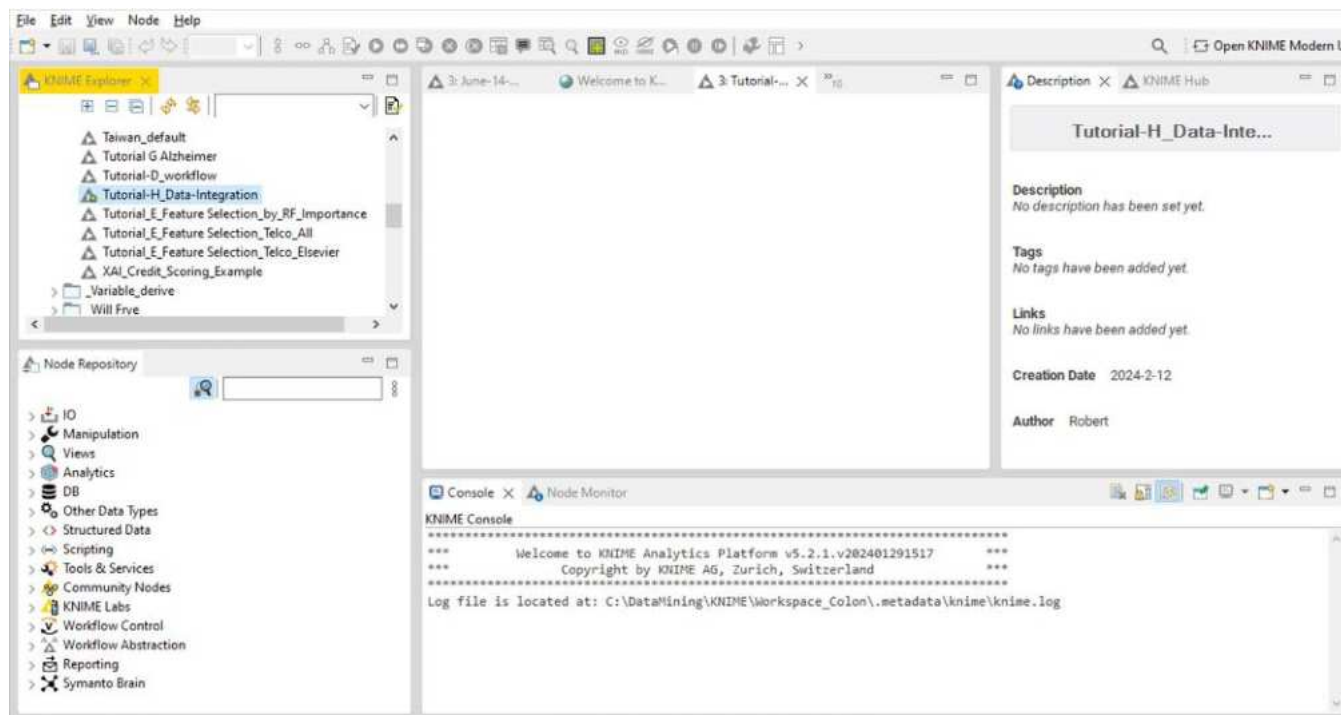


FIG. H.4 Blank workflow ready for editing.

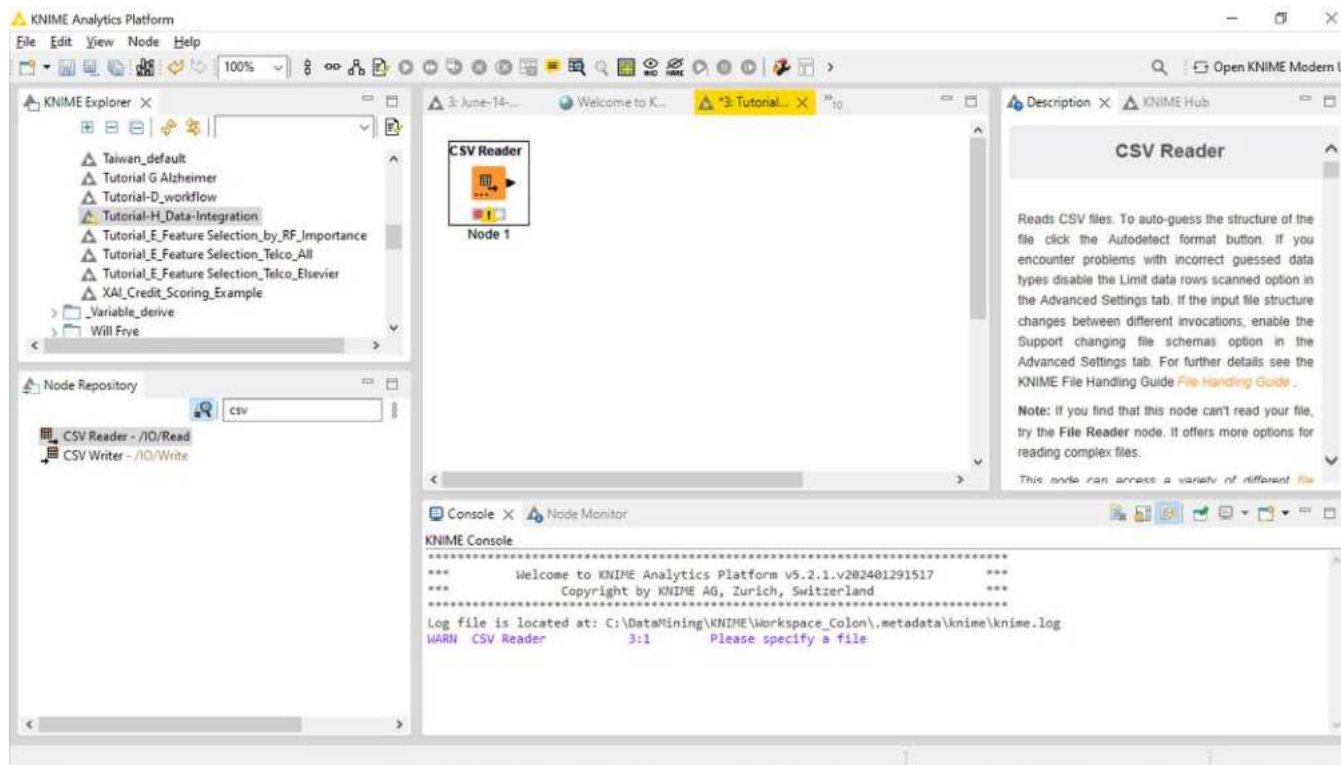


FIG. H.5 Adding an unconfigured CSV Reader node to the workflow.

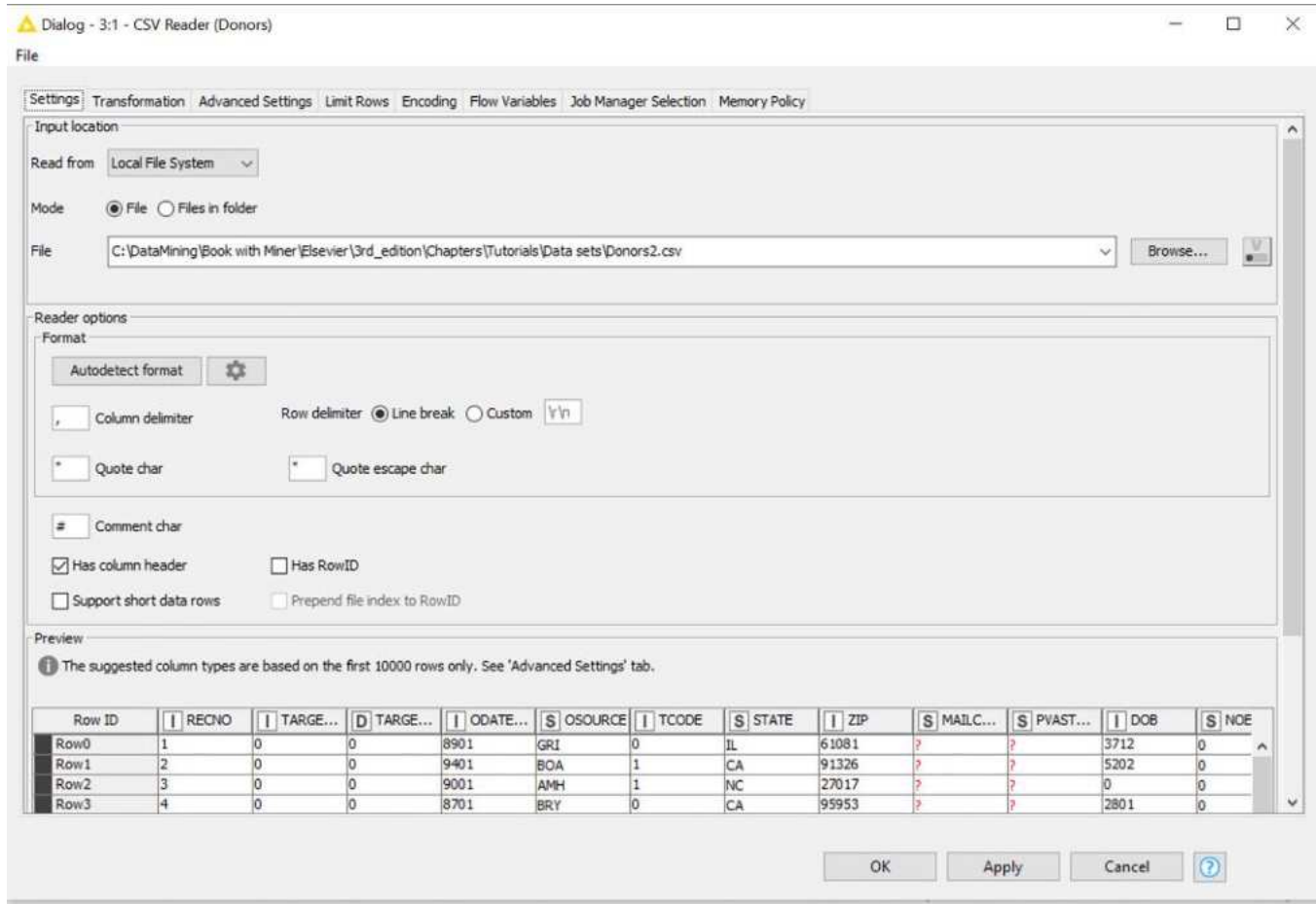


FIG. H.6 CSV Reader Configuration Dialog screen.

rows” box is checked, and the join Venn diagram shown to the right is labeled as “**Inner join.**” We will leave it this way because all of the Row IDs in the Donors file are in the Demographics, and there are no others present in either file. The colored portion of the two overlapping circles shown to the right refers only to the intersection of the areas of the circles representing the matching Row IDs.

Click the Left unmatched rows box to see what happens: the rest of the left circle is colored also. This is the *left-outer join* logical join condition.

The left-outer join of the tables will add all fields from the **Donors** table with the fields on the **Demographics** table that match a common key. Since all records of both data sources have the same RECNO key values, the **Inner Join** option could also be used in this case, and it will be used for all Joiner nodes in this tutorial. The Row ID values are also the same in both files; therefore we could use the Row ID columns as join keys.

It is usually the case, however, that two data files to be joined come from different data sources, and some of the join keys of the first file

are not contained in the second file. A left-outer join includes all rows of the first (the primary) file and only those records in the second file that match the join keys in the first file. Because this situation is most common in business databases, most of the joins performed in business analytics model-building applications are left-outer joins.

16. With the **Joiner** configuration as shown in FIG. H.11 still on display, click on the **Column Selection** tab.

The **Column Selection** tab allows a combination of join and filter operations whenever defining which fields to join. Note also that all fields (columns) from both files are selected automatically, and we will leave all as selected. You could omit one to many fields in each file to be joined to form the output dataset.

- a. Click **OK** to save the configuration and close the *Configuration Dialog* screen.
17. Right-click the **Joiner** node and select **Execute**.

Notice that the bottom two output ports of the executed Joiner node have red X symbols across them. Position your **cursor** over the middle output

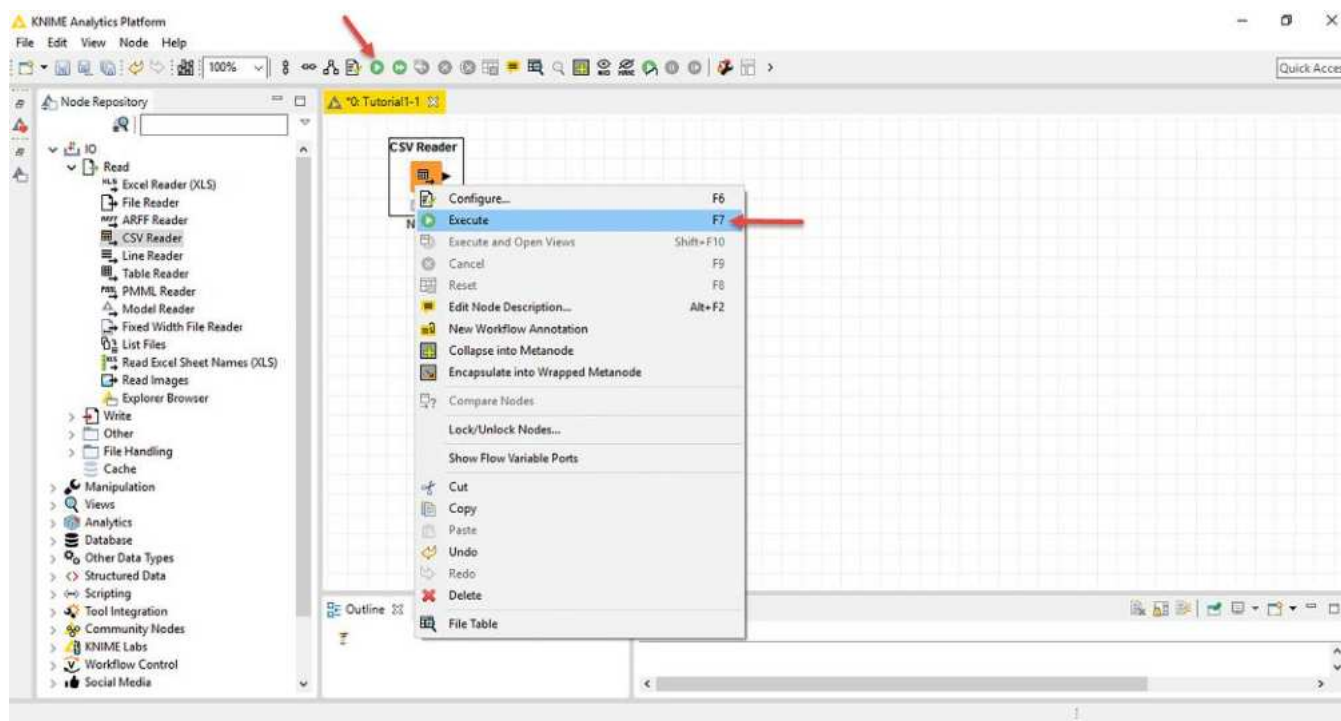


FIG. H.7 Options for executing KNIME nodes.

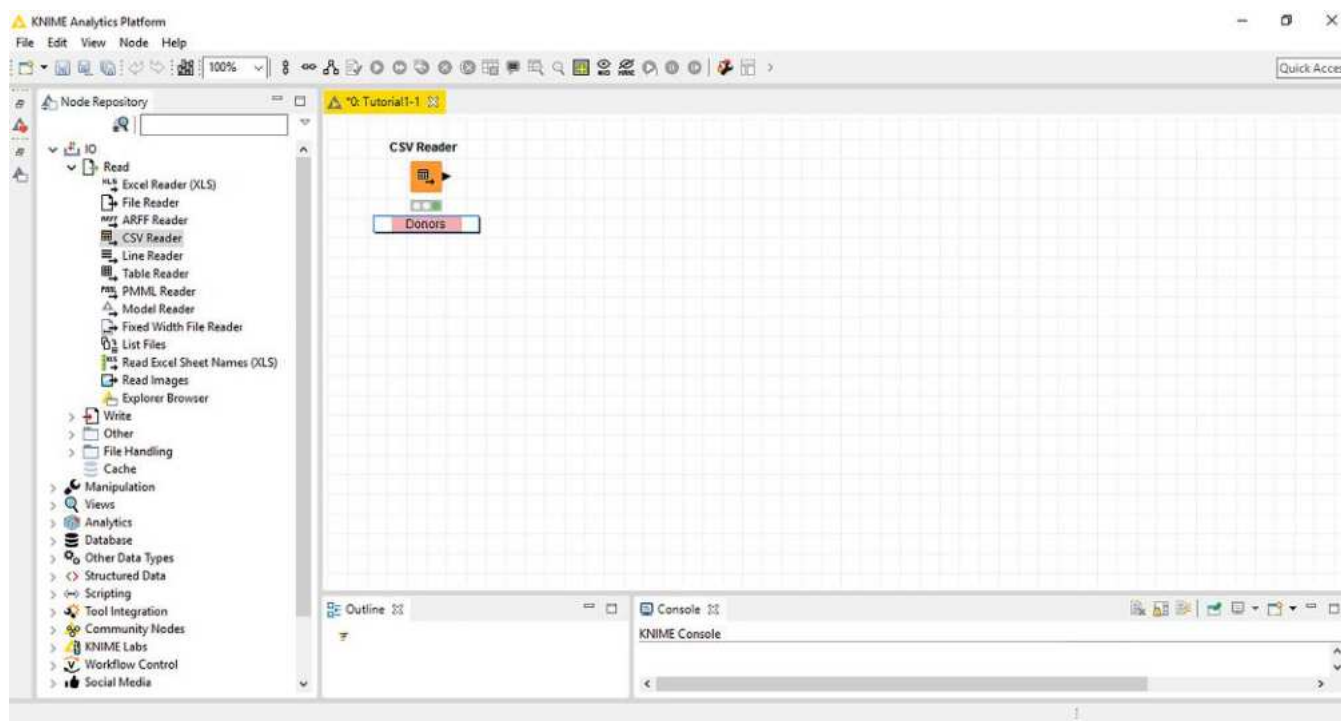


FIG. H.8 Labeling the node.

port to see that it would contain left unmatched rows, if there were any (there are none). The lower output port would contain right unmatched rows if there were any (there are none). Other configura-

tions of the **Joiner** node would direct some rows into these output ports.

a. In order to activate the lower two output ports of the **Joiner** node, right-click on the node and

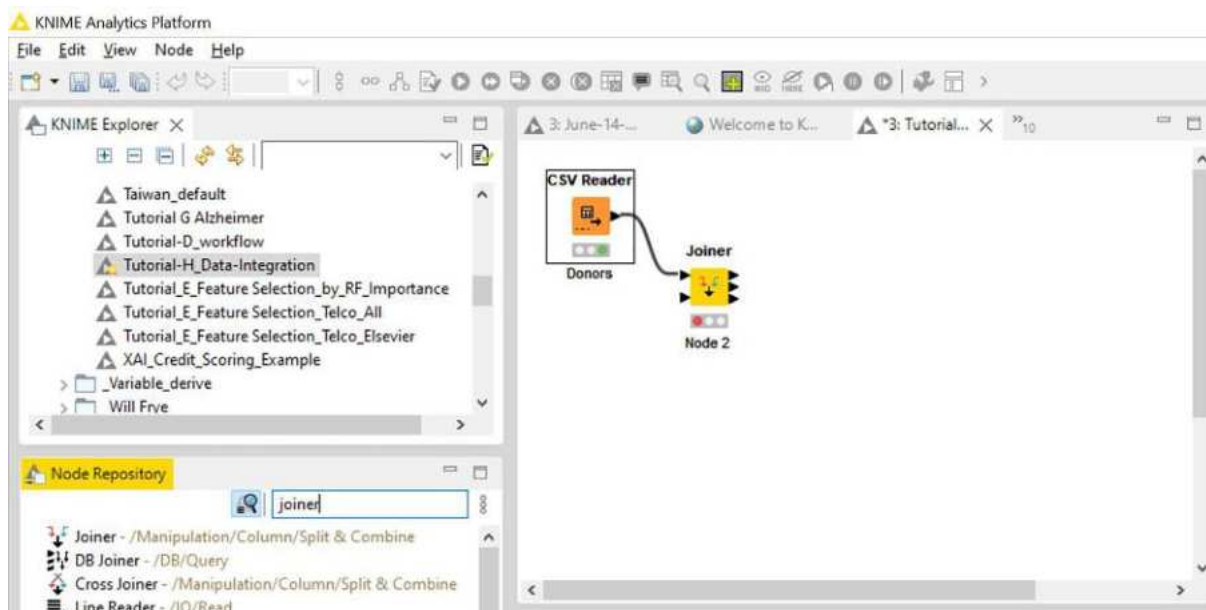


FIG. H.9 Finding a new node and automatically connecting it to an existing node on the workflow.

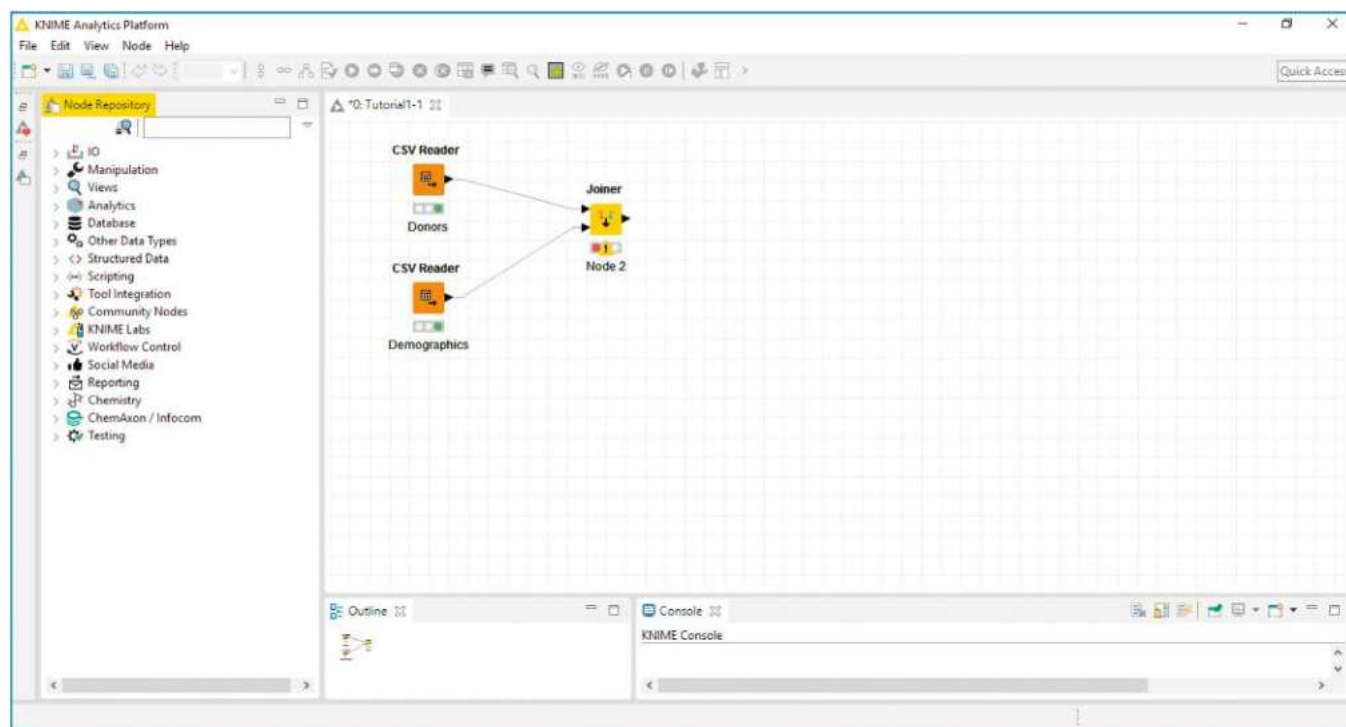


FIG. H.10 Adding a second data source to be integrated.

- choose **Configure** to open the *Configuration Dialog* screen.
- b. Under the **Output options** section, check the box for “**Split join result into multiple tables.**”
18. View the joined table by right-clicking on the **Joiner** node and selecting “**Joined table.**” The

Joined table displays and has 19,058 rows and 102 columns, as shown in Fig. H.12.

It is important to check the resulting file to validate that the columns were joined in the correct order. This combination of data sources will create the Analytical Record and, based on the modeling

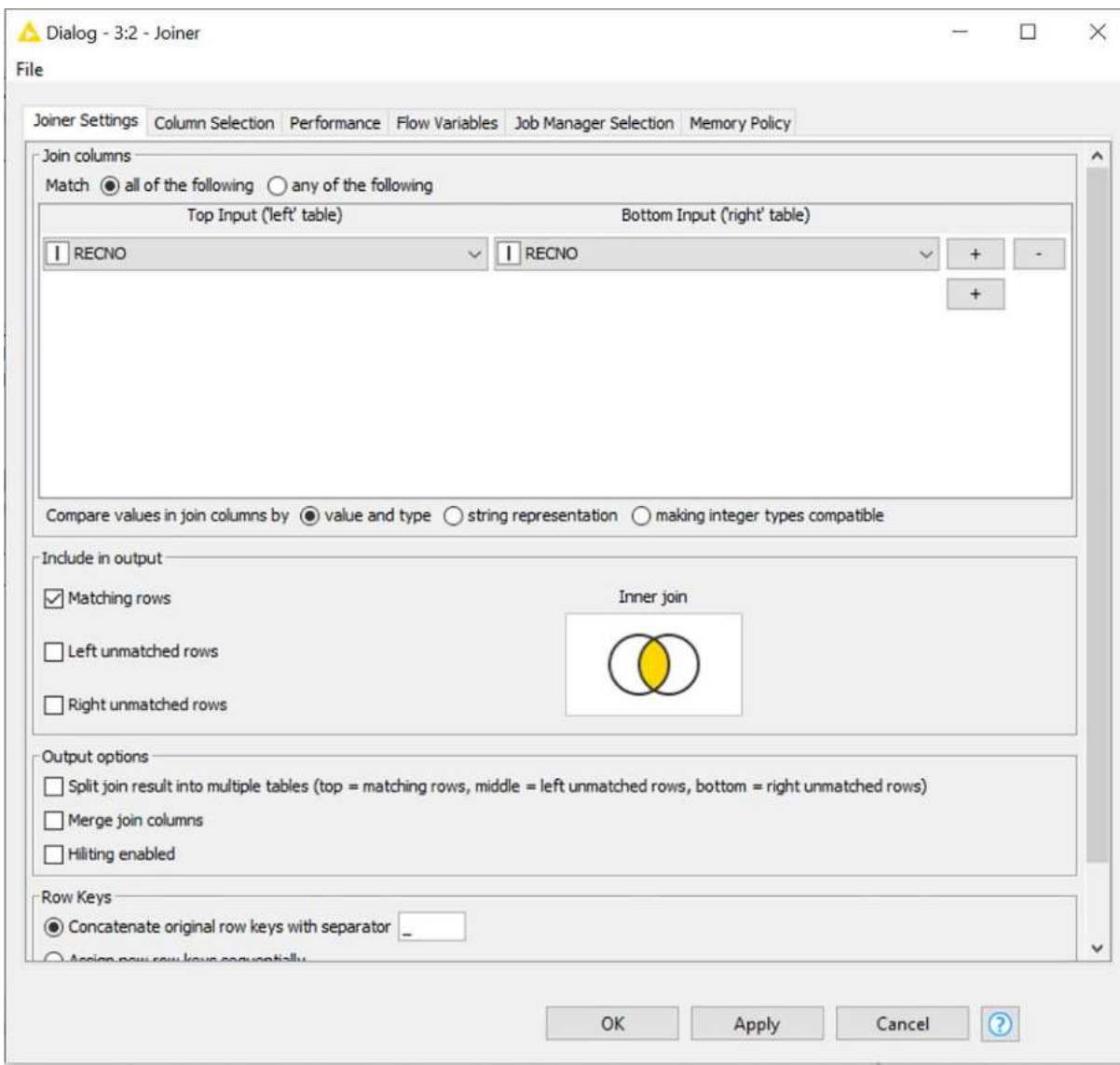


FIG. H.11 The configured Joiner node.

Joined table - 0:2 - Joiner

File

Table "default" - Rows: 19049 Spec - Columns: 102 Properties Flow Variables

Row ID	I RECNO	I TARGE...	D TARGE...	I ODATE...	S OSOURCE	I TCODE	S STATE	I ZIP	S MAILC...	I
Row0_Row0	1	0	0	8901	GRI	0	IL	61081	?	?
Row1_Row1	2	0	0	9401	BOA	1	CA	91326	?	?
Row2_Row2	3	0	0	9001	AMH	1	NC	27017	?	?
Row3_Row3	4	0	0	8701	BRY	0	CA	95953	?	?
Row4_Row4	5	0	0	8601	?	0	FL	33176	?	?
Row5_Row5	6	0	0	9401	CWR	0	AL	35603	?	?
Row6_Row6	7	0	0	8701	DRK	0	IN	46755	?	?
Row7_Row7	8	0	0	9401	NWN	0	LA	70611	?	?
Row8_Row8	9	0	0	8801	LIS	1	IA	51033	?	?
Row9_Row9	10	0	0	9401	MSD	1	TN	-37127	?	?
Row10_Row10	11	0	0	9601	AGR	0	KS	67335	?	?
Row11_Row11	12	0	0	9601	CSM	1	IN	46220	?	?
Row12_Row12	13	0	0	8901	ENQ	0	MN	56475	?	?
Row13_Row13	14	0	0	9201	HCC	1	LA	70791	?	?
Row14_Row14	15	0	0	9301	USB	1	UT	84720	?	?
Row15_Row15	16	0	0	9401	FRC	1	CA	90056	?	?
Row16_Row16	17	0	0	9401	RKB	0	MI	48067	?	?

FIG. H.12 The joined table so far.

needs, the order of fields is important. The fields from the input file of the first **CSV Reader** *Donors* should come first, then the fields from the input file of the second **CSV Reader** *Demographics* should be appended as an outer join.

- a. Close the window by clicking on the X in the upper-right corner of the screen.

19. Click **File** → **Save** to save the workflow.

To be on the safe side, save your workflow often.

20. Now, add a third table to the joined dataset by adding a third **CSV Reader** node to the workflow below the second **CSV Reader** node.

- a. Configure the node to load the *Donors_Hist2.csv*, as you did for the other two data sources.
 - i. Check the box for Has column header.
 - ii. Uncheck the box in the Advanced Settings tab for Limit rows scanned.
- b. Name the third **CSV Reader** node as *Donor Hist*.
- c. Execute the node.

In case you get an error (a red X on the Node), go back to the configuration screen and:

- i. Click on the **Advanced settings** tab and uncheck the **Limit data rows scanned** box. There was probably some entry in one of the variables in a row that did not match the data type estimated by scanning the first 10,000 rows. Unchecking this box causes KNIME to scan all rows in a data file.
- ii. Execute the node again. That should fix the error in data type estimation.

21. Add a second **Joiner** node to the workflow and connect it to the third **CSV Reader** node, as shown in Fig. H.13.

The workflow shown in Fig. H.13 may not fit on your display screen. To go to full-screen mode for that display pane, double-click on the workflow name in the upper bar of the modeling screen. To return to the full interface screen, double-click again on the workflow name.

- a. Configure the second **Joiner** node, as you did for the first **Joiner** node and execute it.

22. Connect the output triangle of the first **Joiner** node to the top triangle of the new **Joiner** node. Then connect the third **CSV Reader** node *Donor Hist* to the bottom triangle of the new **Joiner** node.

23. Execute the second **Joiner** node.

- a. Right-click on the second **Joiner** node and select “**Joined table**” to validate that the columns were joined in the correct order.

24. Last, add the fourth table to the joined dataset by adding a fourth **CSV Reader** node to the workflow below the third **CSV Reader** node.

- a. Configure the node to load the *Promo_Hist.csv* as you did for the other data sources.

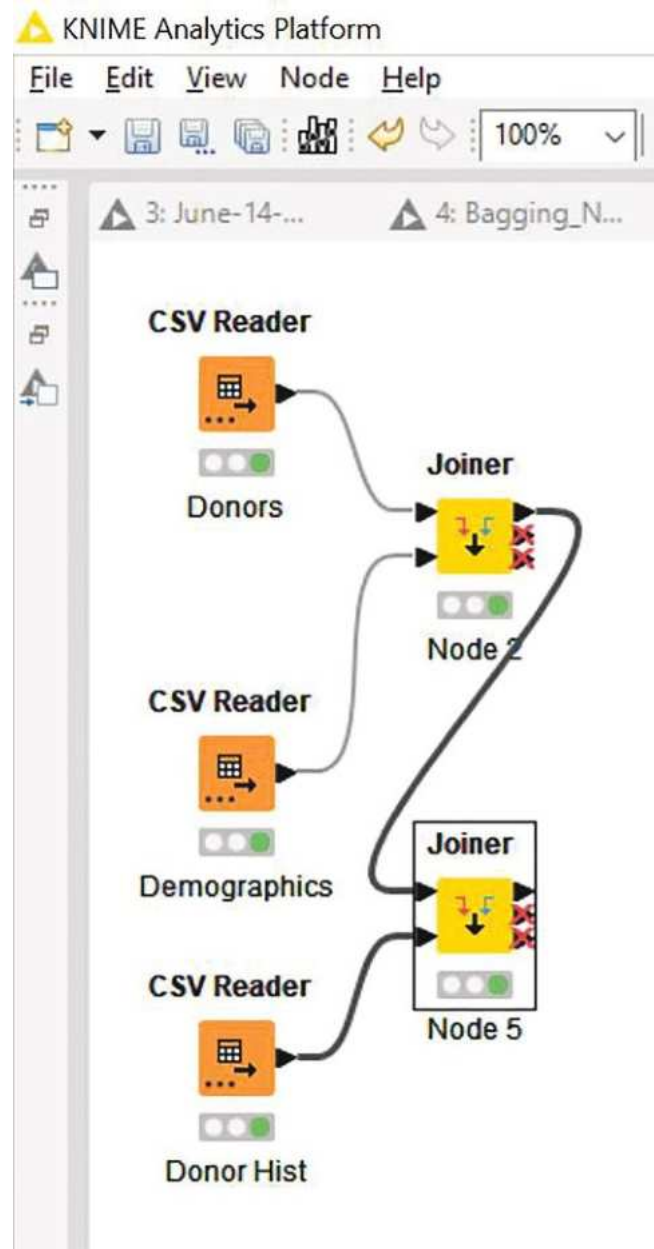


FIG. H.13 File connections with Joiner nodes to integrate three data sources.

- i. Check the box for Has column header.
- ii. Uncheck the box in the Advanced Settings tab for Limit rows scanned.
- b. Name the third **CSV Reader** node *Promo Hist*.
- c. Execute the node.
25. Add a third **Joiner** node, connect it to the new **CSV Reader** node, and configure it as the other **Joiner** nodes.
26. Execute the third **Joiner** node and check the **Joined Table**.

- a. Right-click on the third **Joiner** node and select “**Joined result table**” to validate that the columns were joined in the correct order.
- b. Fig. H.14 shows the workflow at this stage.
27. Select the bottom **Joiner** node.
28. On the **Node Repository** section, expand the **IO** section:
 - a. Enter “**CSV**” into the search box in the **Node Repository** section.
 - b. Double-click on the **CSV Write** node to connect it to the selected bottom **Joiner** node, as shown in Fig. H.15.
29. Double click the **CSV Writer** node on the workflow.
 - a. Select the **Settings** tab in the *Configuration Dialog* screen and specify the output file location and name for **Output Location**.
 - b. Name the file **Tutorial-H_Integrated-file.csv**. Make sure the file extension **.csv** is explicitly added to the filename; otherwise, KNIME will not recognize the file type.
- c. Check the **Write column header** checkbox.
- d. Select **overwrite** radio button in the “**If exists:**” section, as shown in Fig. H.16.
- e. Click **OK** to return to the workflow screen.
30. Click on the  icon on the top menu to execute all nodes in the workflow.

This is an easy way to execute all nodes in the workflow at once after making changes in any of the nodes without having to reexecute them one by one. Another way to do this is to execute the last node in the workflow, as it will execute all up-stream nodes as well.
- After executing all nodes, note that all of them have a green dot in the status bar, indicating that the execution was successful, as shown in Fig. H.17.
31. Click **File** → **Save** to save the workflow.

Congratulations! You have integrated four data sources successfully. You will use this integrated file as an input dataset for Tutorials I to O.

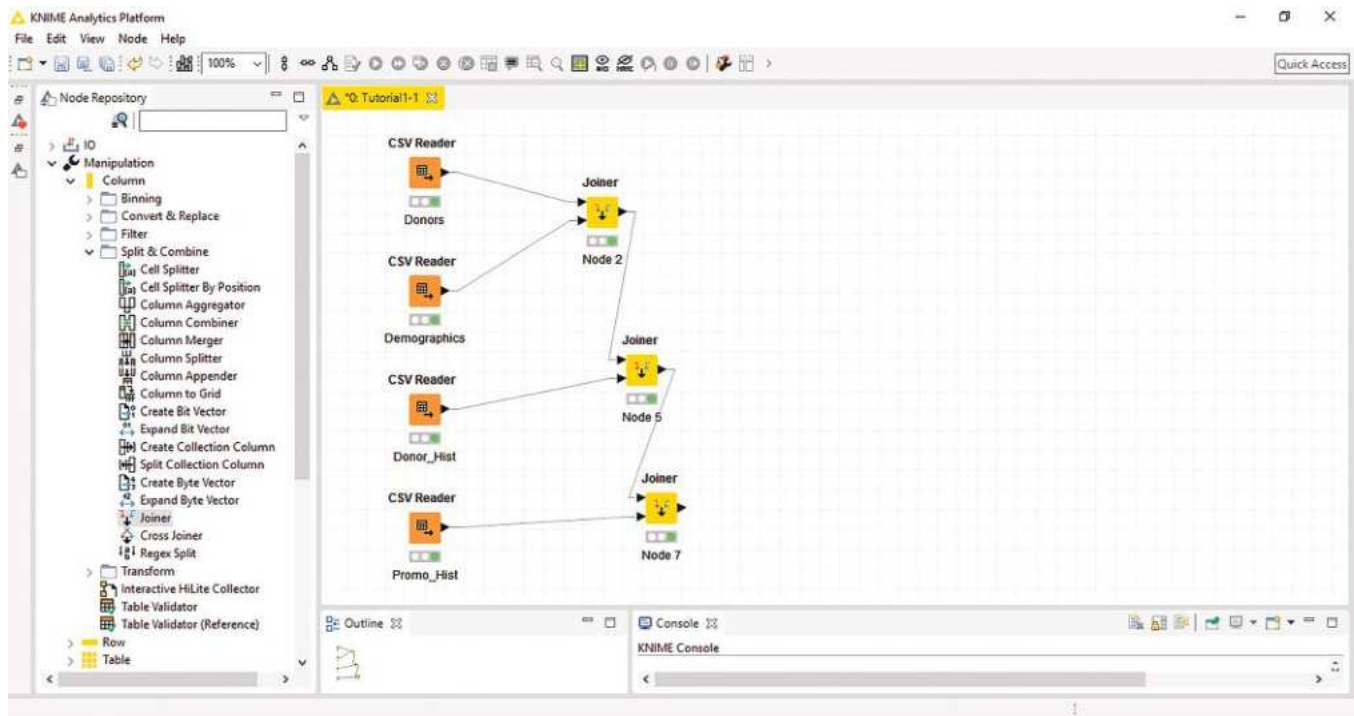


FIG. H.14 Joining four data sources with three joiner nodes.

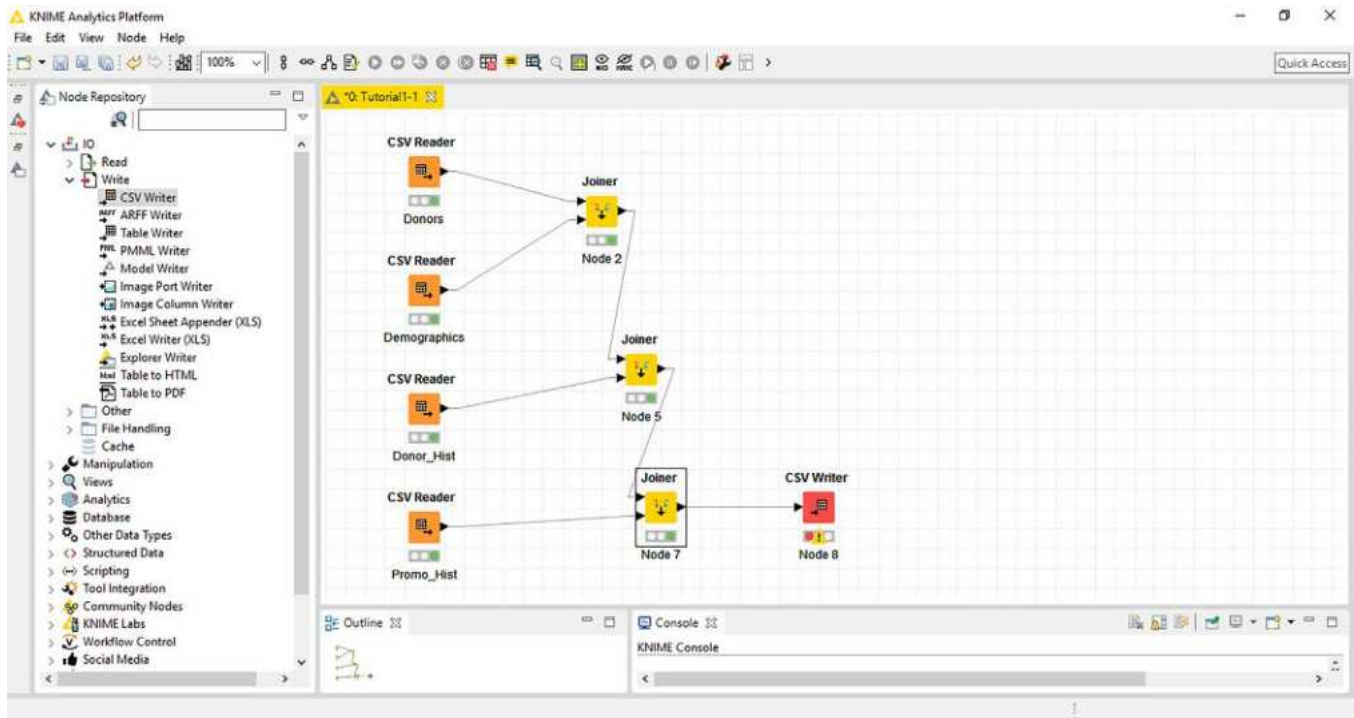


FIG. H.15 Writing the joined results to a file.

The screenshot shows the 'CSV Writer Configuration Dialog' screen. The 'Settings' tab is selected, showing various configuration options. The 'Output location' section includes a 'Write to' dropdown set to 'Local File System', a 'File' text box containing the path 'C:\DataMining\Book with Miner\Elsevier\3rd_edition\Chapters\Tutorials\Data sets\Tutorial-H_Integrated-file.csv', and a 'Browse...' button. The 'Write options' section has checkboxes for 'Create missing folders' and 'If exists' with radio buttons for 'overwrite' (selected), 'append', and 'fail'. The 'Format' section includes fields for 'Column Delimiter' (set to 'System Default'), 'Row Delimiter', 'Quote Char', and 'Quote Escape Char'. The 'Header' section has checkboxes for 'Write column header' (checked), 'Don't write column headers if file exists', and 'Write RowID'. At the bottom, there are 'OK', 'Apply', 'Cancel', and a help icon buttons.

FIG. H.16 The CSV Writer Configuration Dialog screen.

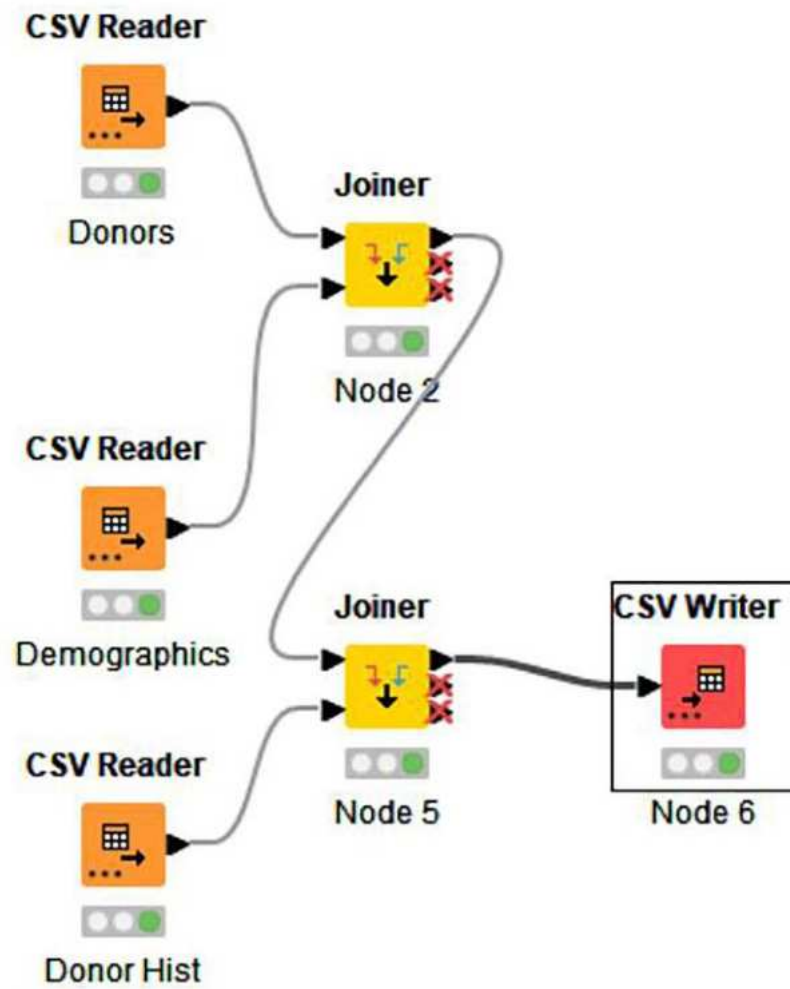


FIG. H.17 The completed Tutorial H workflow.

I

Data description

Roberta Bortolotti, Senior Director, Nasdaq

The data description activity encompasses the analysis of variables in regard to their descriptive statistics (mean, standard deviation, minimum, maximum, frequency tables, and histograms), variable pair relationships, and visual techniques to identify existing complex relationships. Tutorial I is focused on counting the number of rows for each class in the DOMAIN variable (column of the table). Armed with this information:

1. You can guide the modeler in making a choice to combine several of the classes into a single class.
2. It can help you find inappropriate codes for this variable – errors that occur occasionally in database management. Sometimes, orphan codes are left behind and must be updated.
3. You can develop a code list for each variable.

This and subsequent tutorials will abbreviate some basic instructions presented in detail in previous KNIME tutorials. Together, Tutorials H through O constitute an integrated set of examples for performing common data description and data preparation operations of the CRISP-DM Data Analytics process described in detail in [Chapter 3](#). You should perform the operations in each tutorial in sequence in order to learn how to analyze and model your own datasets.

1. Open **KNIME**.
2. Click on **File > New** on the top bar menu to create a new workflow, as you did in Tutorial H.
3. In the Wizard window, select **New KNIME Workflow** and click **Next**.
4. In the next screen, name the new workflow **Tutorial-I_Data-description**. Click on **Browse** to specify the Tutorial folder, if necessary, and click **Finish**.
5. The new workflow will display in KNIME Explorer as an active workflow marked with a green dot.
6. Add a **CSV Reader** node to the workflow space, as you did in Tutorial H.
 - a. Right-click on the **CSV Reader** node.

- b. In the *Configuration Dialog* screen, click on **Browse** for “**Input location:**”
 - c. Navigate to the folder where you stored the output file for Tutorial H.
 - d. Select the **Tutorial-H_Integrated-file.csv** file.
 - e. Confirm that the “**Has Column Headers**” checkbox is checked and “**Has Row Headers**” checkbox is unchecked, as shown in [Fig. I.1](#).
 - f. Click on the **Advanced Settings** tab and uncheck **Limit data rows scanned** to scan all the rows in the file.
 - g. Click **OK** to return to the workflow.
 - h. Execute the workflow using one of the three ways described in Tutorial H.
7. Select the **CSV Reader** node.
8. On the **Node Repository** section, enter “**value**” into the search box.
 - a. Double-click the **Value Counter** node to connect it automatically to the selected **CSV Reader** node.
 The **Value Counter** node generates a frequency table of values that shows the total count of rows for a given unique class code in the DOMAIN column. With this node, it is also possible to see the list of unique values for a specific column.
9. Right-click on the **Value Counter** node and select **Configure**, as shown in [Fig. I.2](#).
10. In the *Configuration Dialog* screen, select DOMAIN for “**Column with values to count,**” as shown in [Fig. I.3](#).
 - a. Click **OK** to return to the dialog screen and then to the workflow.
 - b. Execute the **Value Counter** node.
11. Right-click on the **Value Counter** node and select the **Occurrences** option at the bottom of the list.
12. Once the **Occurrences** table opens, the total count for each unique class code of the DOMAIN variable displays.

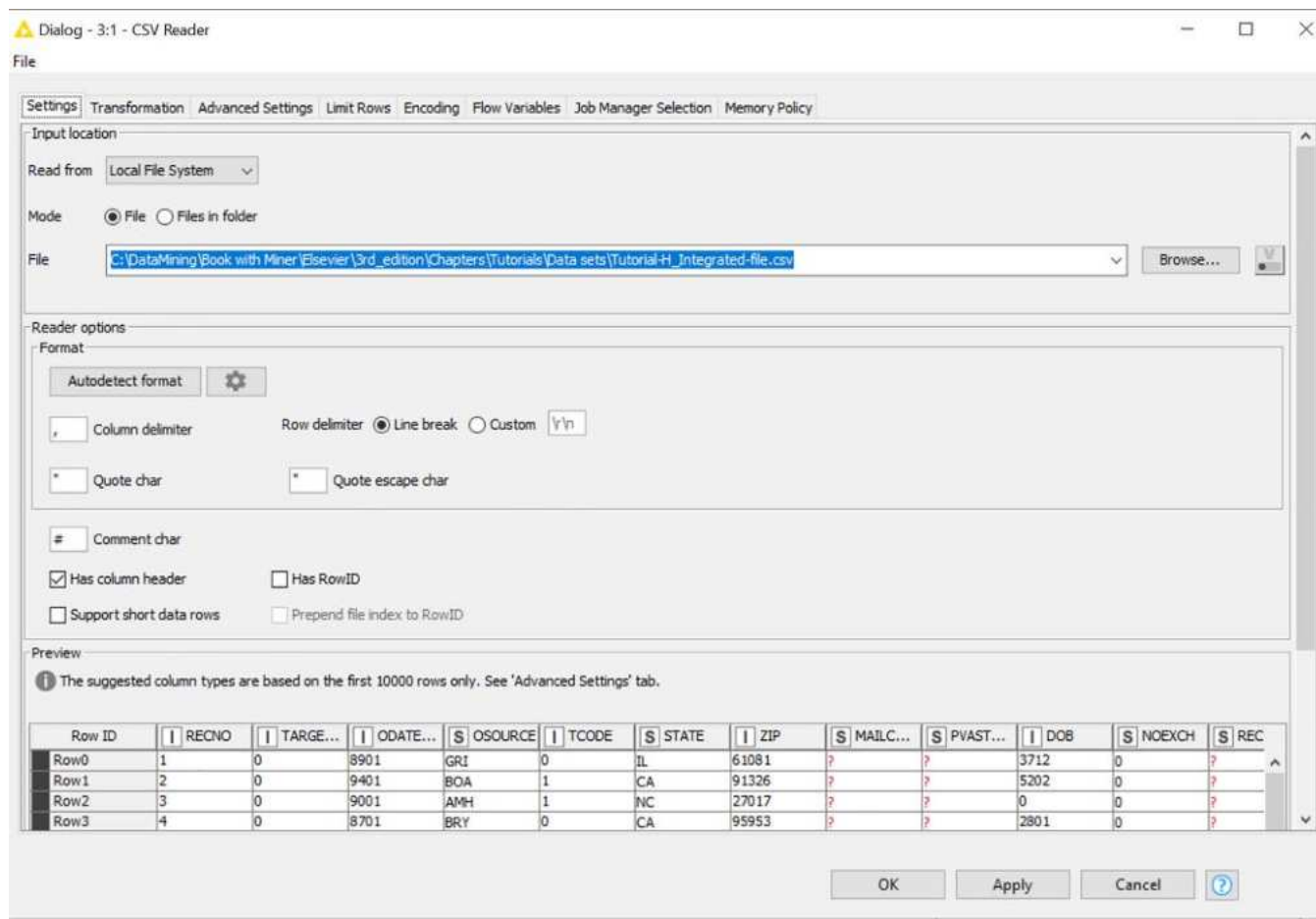
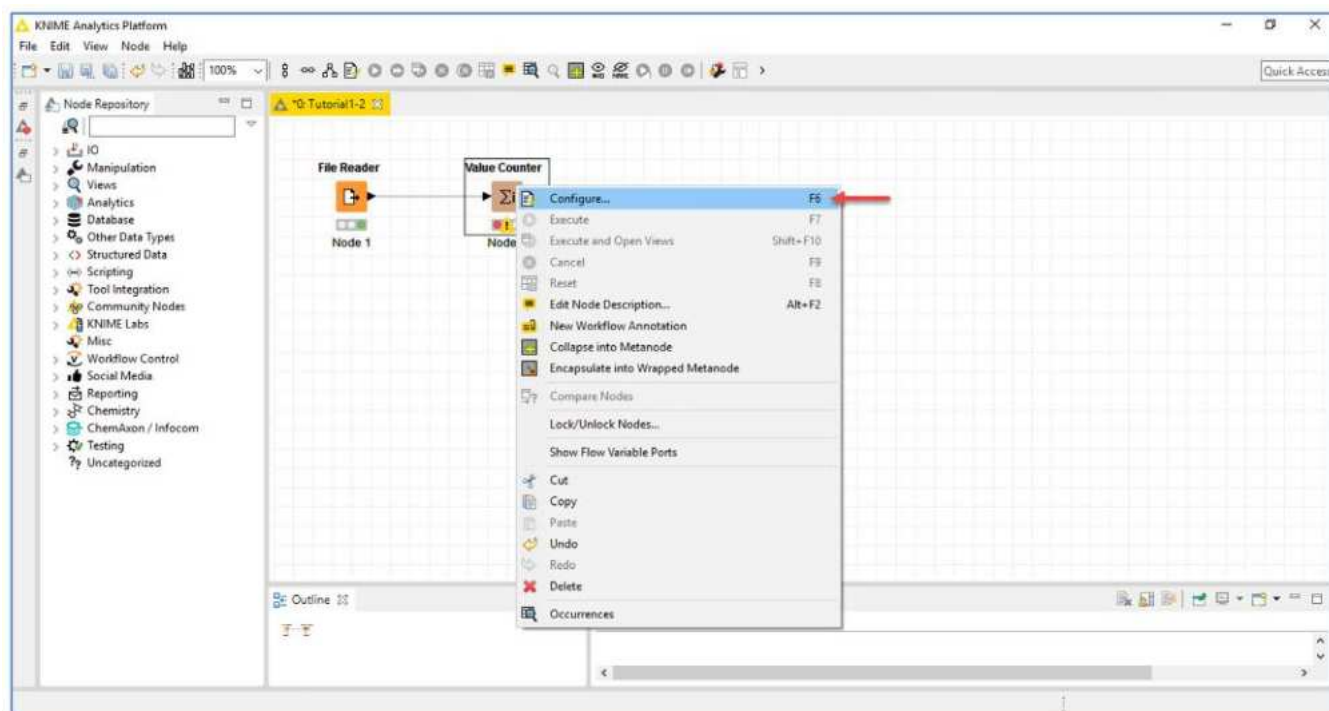


FIG. I.1 Loading the integrated dataset.

FIG. I.2 Using the menu to view the *Configure Dialog* screen.

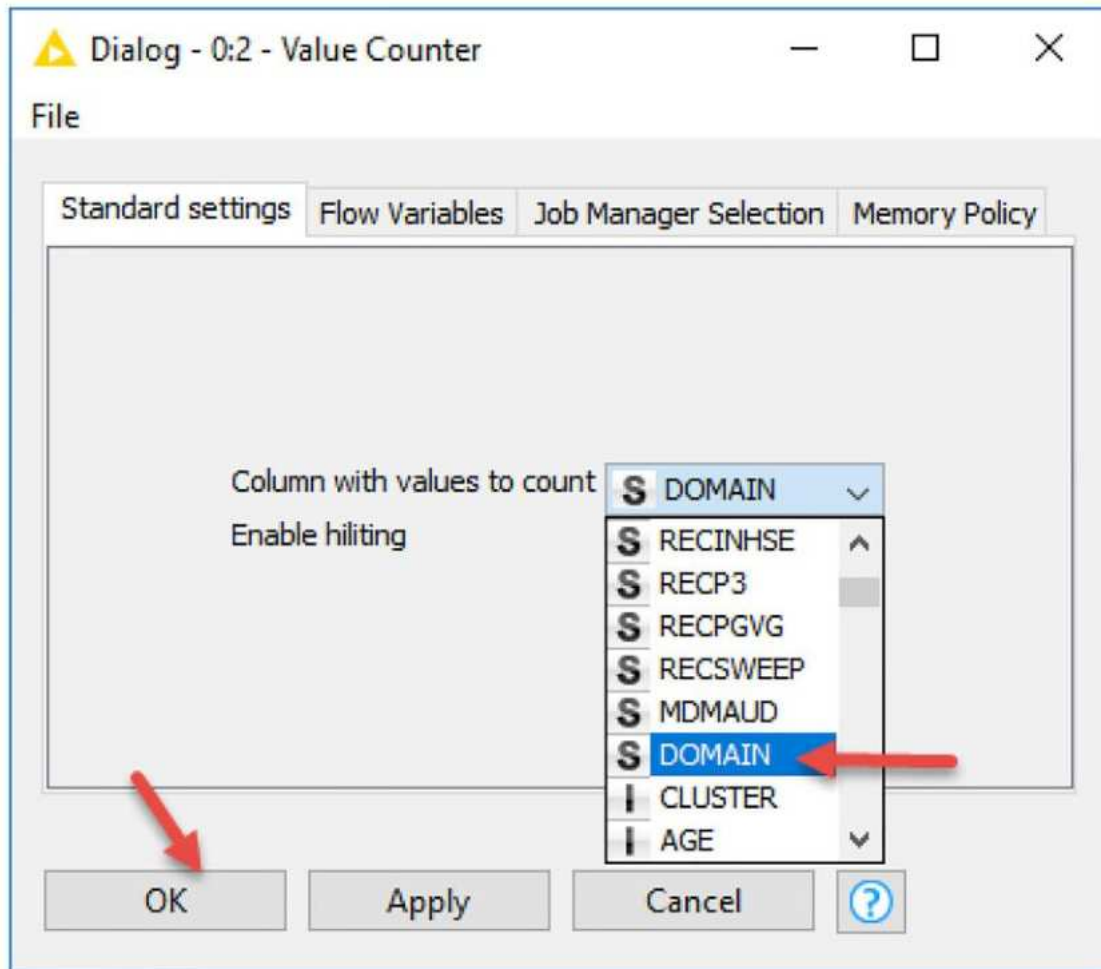


FIG. I.3 Selecting DOMAIN as the field for class counts.

The column on the left is the unique class code and the column on the right named Count is the total number of occurrences for that specific class in the dataset, as shown in Fig. I.4.

- a. Click on the **Count** column.
- b. Select the **Sort Descending**.
This sorting feature is available for each variable (column) in a table by clicking on the column header and selecting the descending sorting option.

Note that in the sorted table (Fig. I.5) the five highest code occurrences (frequencies) are for R2, T2, S1, C2, and S2.

Knowing the frequency of each class code provides a glance at the entire data and can indicate if there are data points concentrated in one area or spread out across a scale. When the frequency distribution is plotted, it becomes clear and easier to understand and draw meaningful interpretation of the data.

- c. Close the table.

13. Click **File > Save** to save the workflow.

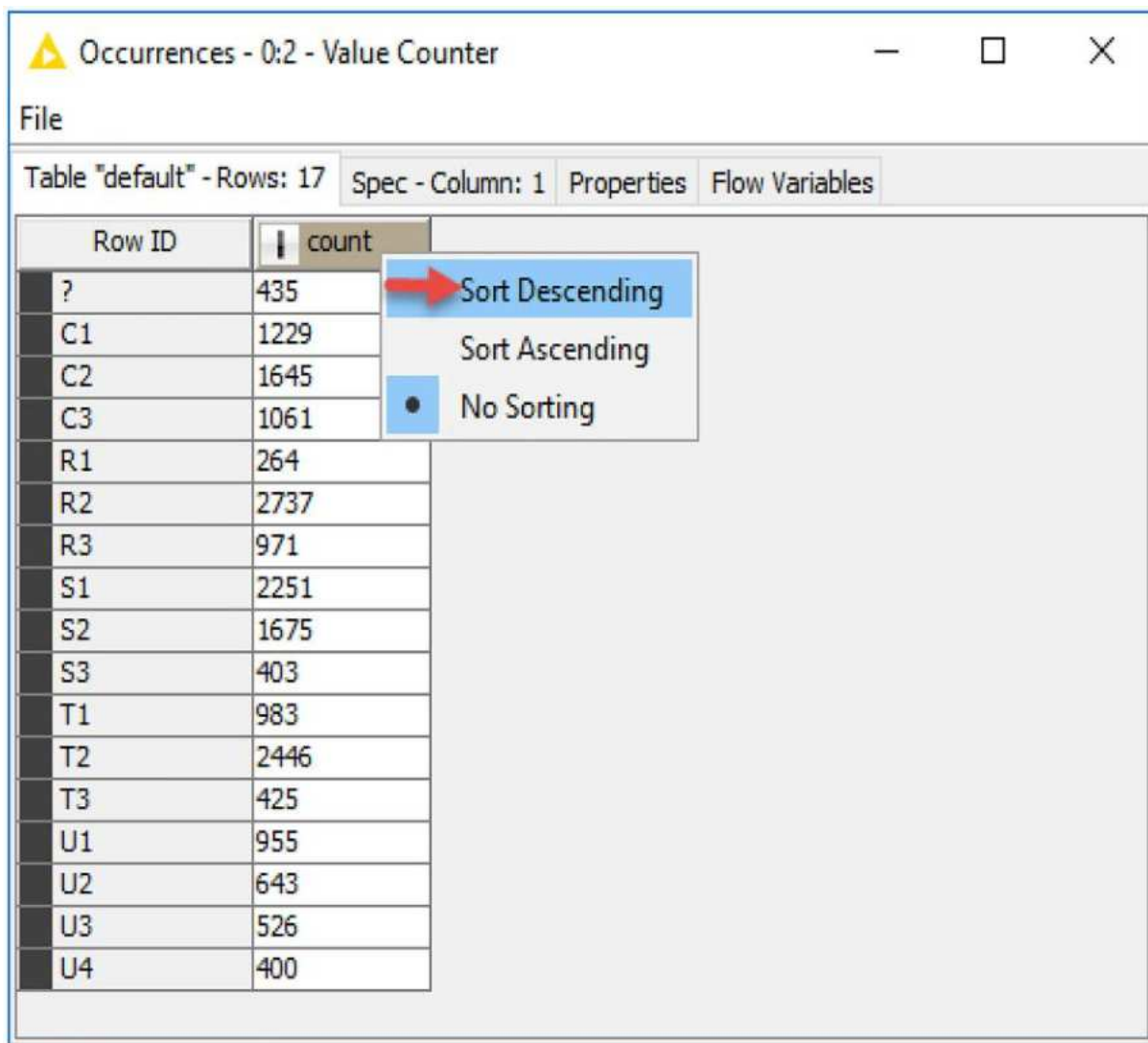


FIG. I.4 How to sort values in a column of a table.

Occurrences - 0:2 - Value Counter

File

Table "default" - Rows: 17 Spec - Column: 1 Properties Flow Variables

Row ID	count
R2	2737
T2	2446
S1	2251
S2	1675
C2	1645
C1	1229
C3	1061
T1	983
R3	971
U1	955
U2	643
U3	526
?	435
T3	425
S3	403
U4	400
R1	264

FIG. I.5 The sorted list of counts of classes of the DOMAIN column.

J

Data cleaning and recoding

Roberta Bortolotti, Senior Director, Nasdaq

Data preparation is an important process that includes very often the cleaning and transformation of the raw data. These operations are needed to correct and filter bad data, reformat, or even filter out data that are too detailed, causing more noise in the model than anything else. With careful and comprehensive data preparation, the data will speak for itself during its processing and analysis, highlighting better insights and, with no doubt, better outcomes from any modeling effort.

This tutorial demonstrates one example of how to recode the data so that errors that occurred during data extraction are addressed.

This and subsequent tutorials will abbreviate some basic instructions presented in detail in previous KNIME tutorials. Together, Tutorial H through Tutorial O constitute an integrated set of examples for performing common data description and data preparation operations of the CRISP-DM Data Analytics process described in detail in [Chapter 3](#). You should perform the operations in each tutorial in sequence in order to learn how to analyze and model your own datasets.

1. Open **KNIME**.
2. Click on **File > New** on the top bar menu to create a new workflow, as you did in Tutorial I.
3. In the Wizard window, select **New KNIME Workflow** and click **Next**.
4. In the next screen, name the new workflow **Tutorial-J_Data-cleaning-recoding**. Click on **Browse** to specify a Tutorial folder, if necessary, and click **Finish**.
The new workflow will display in KNIME Explorer as an active workflow marked with a green dot.
5. Add a **CSV Reader** node to the workflow space, as you did in Tutorial I.
 - a. Right-click on the **CSV Reader** node.
 - b. In the *Configuration Dialog* screen, click on **Browse** for “**Input location:**”
 - c. Navigate to the folder where you stored the output file for Tutorial H.
 - d. Select the **Tutorial-H_Integrated-file.csv** file.

- e. Confirm that the “**Has Column Headers**” checkbox is checked and “**Has Row Headers**” checkbox is unchecked.
- f. Uncheck the box in the Advanced Settings tab for **Limit rows scanned**.
6. Execute the **CSV Reader** node.
7. Right-click on the **CSV Reader** node and select **File Table**.
8. Once the table displays, click on **Spec – Columns** tab.
 - a. Look for variable **ZIP** under **Columns**.
Note that there are negative values for **ZIP** and that its type is set to integer in the **Column Type**. **ZIP** codes are not actually numbers, but a system used by the US Postal Service, consisting of five digits. Each of them and their order defines a location in the United States: the first digit represents a group of US states, the second and third digits represent a region in that group (or large city), and the fourth and fifth digits represent a group of delivery addresses within that region.
With this knowledge, it is safe to conclude that there are a few data errors to fix here in our dataset:
 - i. **ZIP** codes cannot be negative numbers.
 - ii. **ZIP** codes must have five digits.
 - iii. **ZIP** codes should be considered strings (or text) as they are not really numbers.
 - b. Close the **File Table**.
9. On the **Node Repository** section, enter “**math**” into the search box.
 - a. Double-click the **Math Formula** node.
 - b. Connect the output triangle of the first **CSV Reader** node to the left triangle of the **Math Formula** node.
The **Math Formula** node computes data transformation based on a specified mathematical expression, that is, which output values can be appended as a new column in the dataset or replaced in the input column itself.

File Table - 3:6 - CSV Reader

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Columns: 220	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...	Value 0	Value 1	Value 2	Valu
RECNO	Number (integer)	0					1	19,049	?	?	?	?
TARGET_B	Number (integer)	1					0	1	?	?	?	?
TARGET_D	Number (double)	2					0	100	?	?	?	?
ODATEDW	Number (integer)	3					8,601	9,701	?	?	?	?
OSOURCE	String	4					?	?	?	?	?	?
TCODE	Number (integer)	5					0	72,002	?	?	?	?
STATE	String	6					?	?	IL	CA	NC	FL
ZIP	Number (integer)	7					-99,669	99,901	?	?	?	?
MAILCODE	String	8					?	?	B	?	?	?
PVASTATE	String	9					?	?	P	E	?	?
DOB	Number (integer)	10					0	9,508	?	?	?	?
NOEXCH	Number (integer)	11					0	1	?	?	?	?
RECINHSE	String	12					?	?	X	?	?	?
RECP3	String	13					?	?	X	?	?	?
RECPGVG	String	14					?	?	X	?	?	?
RECSWEEP	String	15					?	?	X	?	?	?
MDMAUD	String	16					?	?	XXXX	C1CM	D1CM	L1CM
DOMAIN	String	17					?	?	T2	S1	R2	S2
CLUSTER	Number (integer)	18					1	53	?	?	?	?
AGE	Number (integer)	19					2	98	?	?	?	?
AGEFLAG	String	20					?	?	E	I	?	?
HOMEOWNR	String	21					?	?	H	U	?	?
CHILD03	String	22					?	?	M	F	B	?
CHILD07	String	23					?	?	M	B	F	?
CHILD12	String	24					?	?	F	M	B	?
CHILD18	String	25					?	?	M	F	B	?
NUMCHLD	Number (integer)	26					1	7	?	?	?	?
INCOME	Number (integer)	27					1	7	?	?	?	?

FIG. J.1 Inspecting data error of the ZIP variable.

10. Right-click on the **Math Formula** node and select **Configure** (see Figure J.2).
11. In the *Configuration Dialog*, **Math Expression** tab, place your cursor in the **Expression** box.
The way the Expression box works is: first we select the function to use, then we fill in the function with specific columns (variables) or values, following the function expression.
 - a. Then scroll down the list in the **Function List** window up to function **if(x,y,z)**.
This function operates like the Excel if-statement. It means "If the x-expression is true, do y, else do z."
 - b. Double-click the function **if(x,y,z)** for the function to appear in the **Expression** box.
 - c. The incomplete function specification is shown in Figure J.3.
 - d. Place the cursor in the x-expression position and double-click on the **ZIP** variable in the **Column List** window.
Note that **ZIP** variable appears in the **Expression** box enclosed with the dollar sign (\$ZIP\$), which is a Java convention. Don't remove the dollar signs.
 - e. Add "<0" after \$ZIP\$. This is the condition to use in the evaluation of our mathematical formula.
 - f. Place the cursor in the y-expression position and double-click on **ZIP** variable in the **Column List** window.
Note that **ZIP** variable appears in the **Expression** box enclosed with the dollar sign (\$ZIP\$) as before.
 - g. Add a minus sign "-" before \$ZIP\$. This represents the negative ZIP values observed in Step 9.
 - h. Place the cursor in the z-expression position and double-click on **ZIP** variable in the **Column List** window. Again, the **ZIP** variable appears in the **Expression** box enclosed with the dollar sign (\$ZIP\$).
 - i. Select **Replace Column** and choose the column name **ZIP** from the dropdown.
 - j. Check the **Convert to Int** checkbox.
ZIP is typed as an integer but treated by the node as a real number (include positive and negative numbers, fractions of numbers, and irrational numbers), and the output is rendered as a real number with no decimal places. The output must be converted back into an integer.
 - k. Click **OK**.
The complete expression should look as shown in Fig. J.4. It translates to "If ZIP value < 0 is equal to true, then use the positive value."

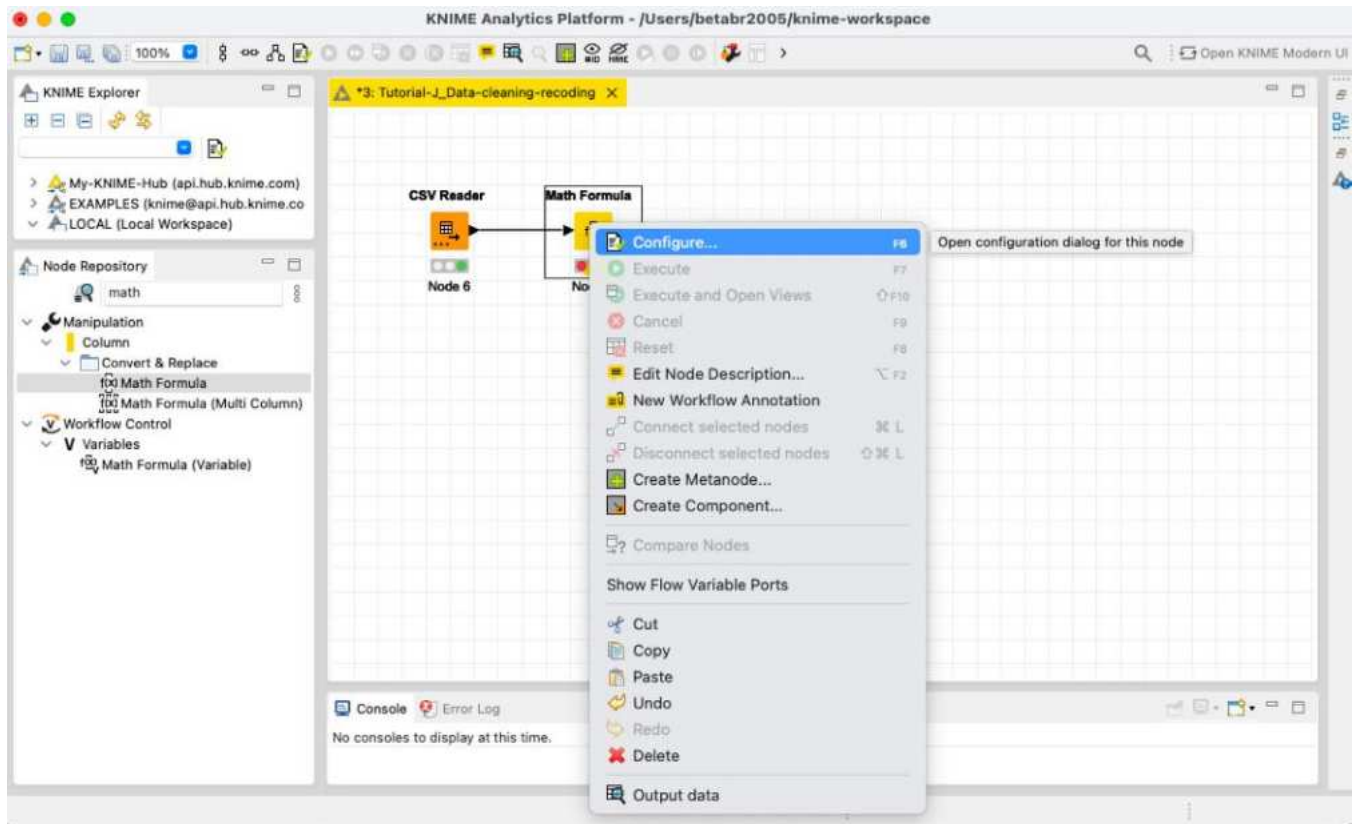


FIG. J.2 Using the menu options to view the *Configuration Dialog* screen for the Math Formula node.

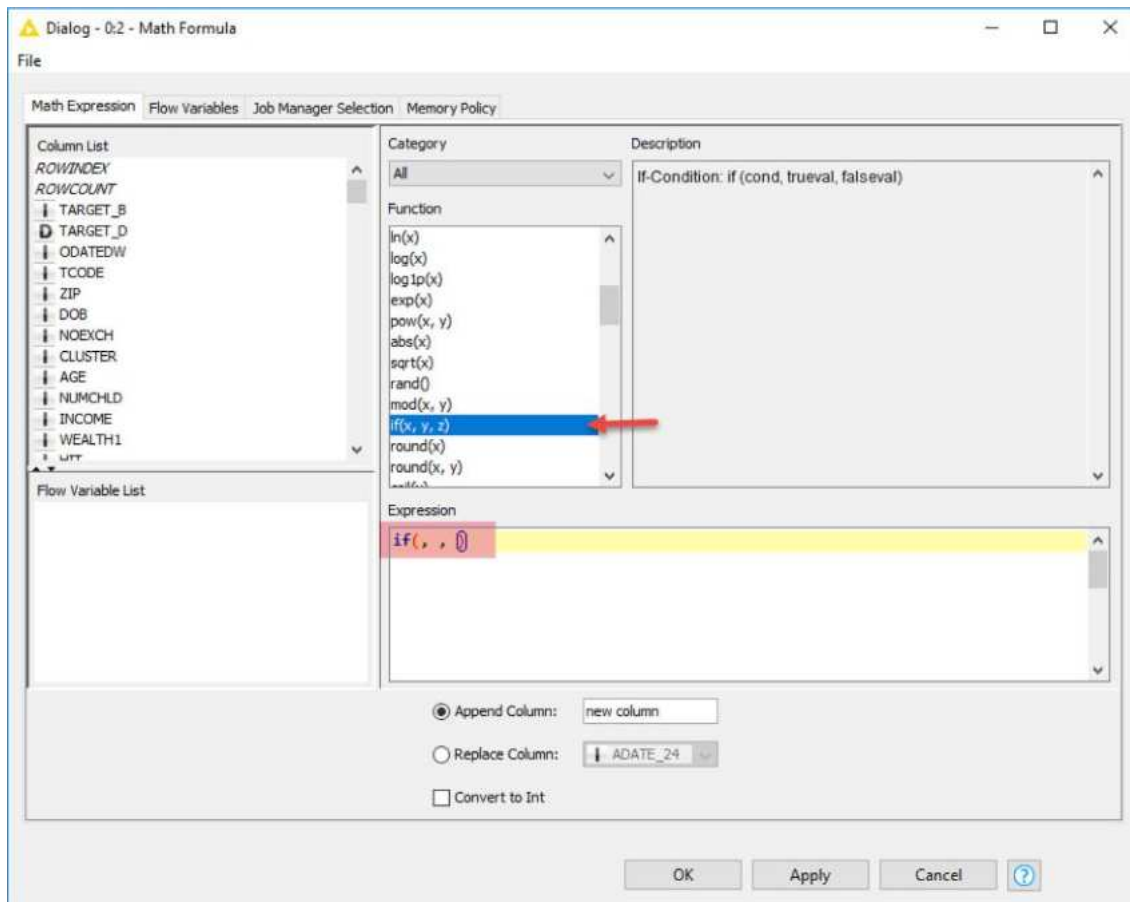


FIG. J.3 Selecting a function expression to be used in the Math Formula node.

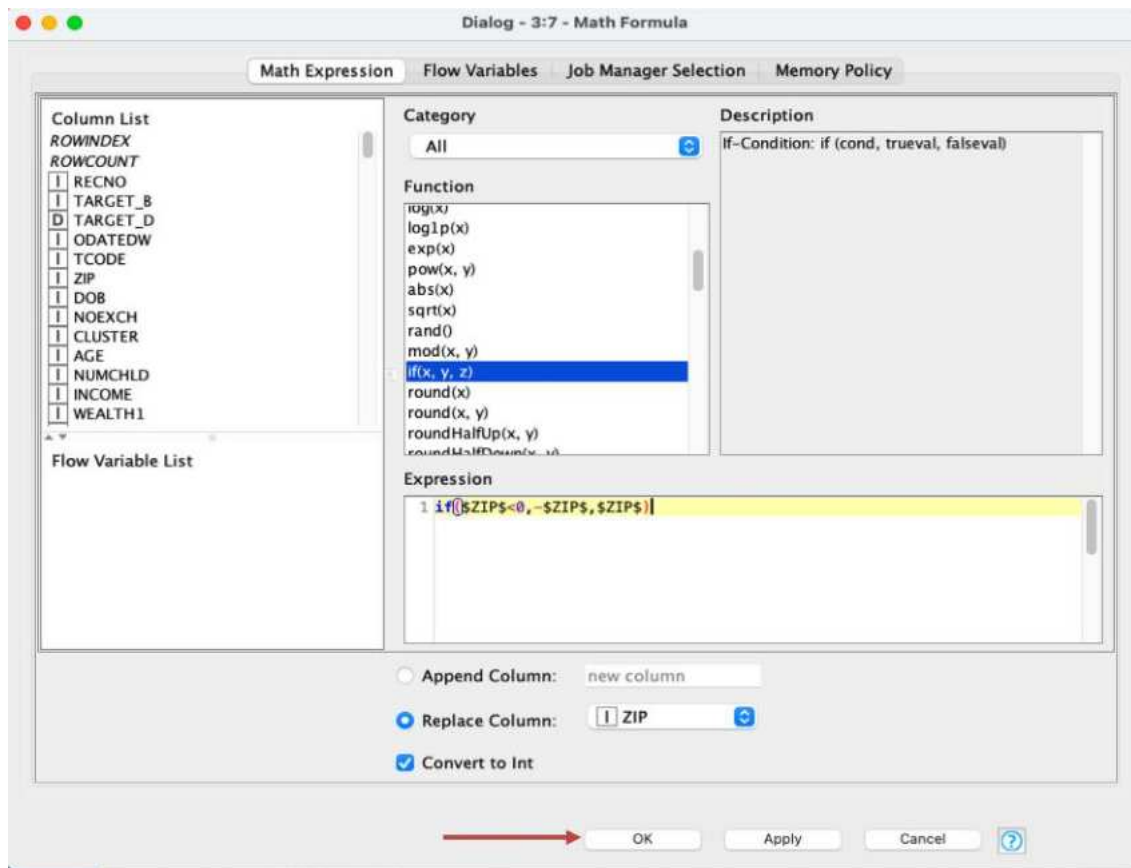


FIG. J.4 Setting a condition expression in the Math Formula node.

Then we define what to do with the output of the expression, which is to replace the values in the ZIP column.

12. Execute the **Math Formula** node.
13. Right-click on the Math Formula node and select **Output Data**.
 - a. On the **Spec – Columns** tab, verify that there are no negative values for ZIP anymore. The negative value issue was fixed. However, remember that a ZIP code needs to have five digits and notice that the minimum value for ZIP is 1754 (see Figure J.5). This reveals a leading zeros issue that still needs to be fixed.
14. Click on **Table Default** tab.
 - a. Click on ZIP header and select **Sort Ascending**. Note that there are some ZIP codes with four digits as shown in Fig. J.6. The leading zero has been stripped off by an autotyping process when they were converted to numbers. This is the same issue with TARGET_B variable. In the next steps, we will properly fix this by converting ZIP and TARGET_B variables to be of String type.
 - b. Close the table.
15. On the **Node Repository** section, enter “**Number to String**” in the search box.
 - a. Double-click the **Number to String** node and connect it to the selected **Math Formula** node. The **Number to String** node converts numbers in a column (or a set of columns) into strings.
- b. Connect the output triangle of the **Math Formula** node to the left triangle of the **Number to String** node.
16. Right-click the **Number to String** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, note that all variables are selected automatically as to be included.
 - b. Click on **Remove All** using the «symbol.
 - c. From the **Excludes** column, select TARGET_B and ZIP variables and click on the »symbol. Note that TARGET_B and ZIP variables are placed in the **Include** column (see Figure J.7).
 - d. Click OK.
17. Execute the **Number to String** node.
18. On the **Node Repository** section, enter “**Column exp**” in the search box.
 - a. Click on the **Number to String** node to highlight it.
 - b. Double-click the **Column Expressions** node to connect it to the highlighted node automatically.

Output data - 3:7 - Math Formula

File

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Columns: 220	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...	Value 0	Value 1	Value
RECNO	Number (integer)	0					1	19,049	?	?	?
TARGET_B	Number (integer)	1					0	1	?	?	?
TARGET_D	Number (double)	2					0	100	?	?	?
ODATEDW	Number (integer)	3					8,601	9,701	?	?	?
OSOURCE	String	4					?	?	?	?	?
TCODE	Number (integer)	5					0	72,002	?	?	?
STATE	String	6					?	?	IL	CA	NC
ZIP	Number (integer)	7					1,754	99,901	?	?	?
MAILCODE	String	8					?	?	B	?	?
PVASTATE	String	9					?	?	P	E	?
DOB	Number (integer)	10					0	9,508	?	?	?
NOEXCH	Number (integer)	11					0	1	?	?	?
RECINHSE	String	12					?	?	X	?	?
RECP3	String	13					?	?	X	?	?
RECPGVG	String	14					?	?	X	?	?
RECSWEEP	String	15					?	?	X	?	?
MDMAUD	String	16					?	?	XXXX	C1CM	D1CM
DOMAIN	String	17					?	?	T2	S1	R2
CLUSTER	Number (integer)	18					1	53	?	?	?
AGE	Number (integer)	19					2	98	?	?	?
AGEFLAG	String	20					?	?	E	I	?
HOMEOWNR	String	21					?	?	H	U	?
CHILD03	String	22					?	?	M	F	B
CHILD07	String	23					?	?	M	B	F
CHILD12	String	24					?	?	F	M	B
CHILD18	String	25					?	?	M	F	B
NUMCHLD	Number (integer)	26					1	7	?	?	?
INCOME	Number (integer)	27					1	7	?	?	?

FIG. J.5 Inspecting data error of the ZIP variable.

Output data - 3:7 - Math Formula

File Edit Hilite Navigation View

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Row ID	RECNO	TARG...	D TARG...	ODAT...	S OSOU...	TCODE	S STATE	ZIP	S MAIL...	S PVAST...	DOB
Row10680	10681	0	0	8701	DRK	0	MA	1754	?	?	0
Row7833	7834	0	0	9501	ARG	0	MA	1876	?	?	5901
Row6030	6031	0	0	9401	MBC	0	MA	2038	?	?	5601
Row2980	2981	0	0	9501	AVN	0	MA	2124	?	?	0
Row12943	12944	0	0	9201	SSS	1	ME	4078	?	?	1301
Row367	368	0	0	9101	BHG	0	ME	4364	?	?	0
Row11631	11632	0	0	9601	CHT	0	ME	4438	?	?	0
Row3295	3296	0	0	9101	AML	1	VT	5033	?	?	2001
Row16577	16578	0	0	9501	HAM	0	VT	5663	?	?	6801
Row7755	7756	0	0	9601	HHN	1	CT	6335	?	?	4401
Row16542	16543	0	0	8601	DRK	2	CT	6460	?	?	0
Row6218	6219	1	11	8601	DRK	2	CT	6468	?	?	5001
Row6169	6170	0	0	8901	DUR	0	CT	6484	?	?	0
Row15022	15023	0	0	9401	STV	0	CT	6497	?	?	0
Row2887	2888	0	0	9401	L15	1	NJ	8008	?	?	2401
Row7730	7731	0	0	8601	BHG	2	NJ	8098	?	?	2001
Row2464	2465	0	0	9501	HAM	1	NJ	8225	?	E	5801
Row6916	6917	0	0	9501	HCC	2	NJ	8401	?	?	0
Row18847	18848	0	0	9101	DUR	0	NJ	8527	?	?	0
Row9987	9988	0	0	9101	UCA	1002	AE	9012	?	?	5801
Row6159	6160	0	0	9501	HHN	0	AE	9173	?	?	0
Row18609	18610	0	0	9001	PV3	1	AE	9182	?	?	5601
Row14910	14911	0	0	9501	DCD	14	AE	9459	?	?	6201
Row12995	12996	1	50	9501	FRC	0	AE	9645	?	?	4701
Row11058	11059	0	0	8701	FCR	1	NY	10001	?	?	3111
Row10546	10547	0	0	9201	LVT	1	NY	10014	?	?	1106
Row18033	18034	0	0	9401	HHH	0	NY	10962	?	?	0
Row15541	15542	0	0	8801	HHH	0	NY	11554	?	?	0

FIG. J.6 Leading zeros data issue for the ZIP variable in the dataset.



FIG. J.7 Configuring transforming values from number to string.

- c. Connect the output triangle of the **Number to String** node to the left triangle of the **Column Expressions** node.

In the next step, we have to add the leading zero to all four-digit ZIP codes. The **Column Expressions** node uses an if-then-else expression syntax similar to Excel..

19. Right-click the **Column Expressions** node and select **Configure**.
 - a. To add an expression, click on the plus “+” in the upper left corner of the screen.
 - b. Click the **Replace Column** box
 - c. Click on the RECNO field in the Output Column box and select the ZIP column as the column to replace.
 - d. Click on the yellow line in the Expression Editor; enter: **if(**
 - e. Position the cursor after the open parenthesis mark and click on the **+ Column** box above.
 - f. Double-click on the ZIP column to paste it into the expression.

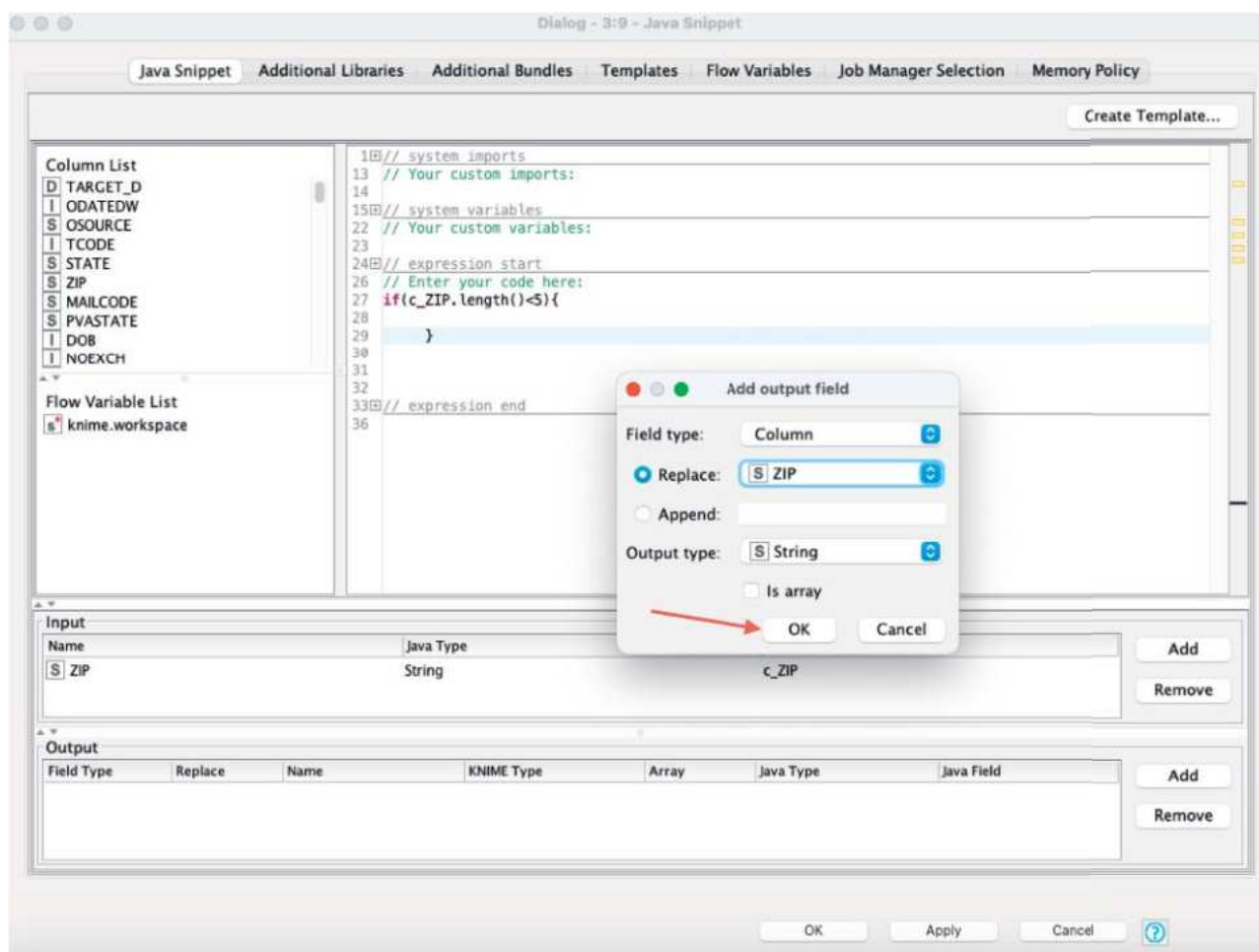


FIG. J.8 Fixing 4-character ZIPs.

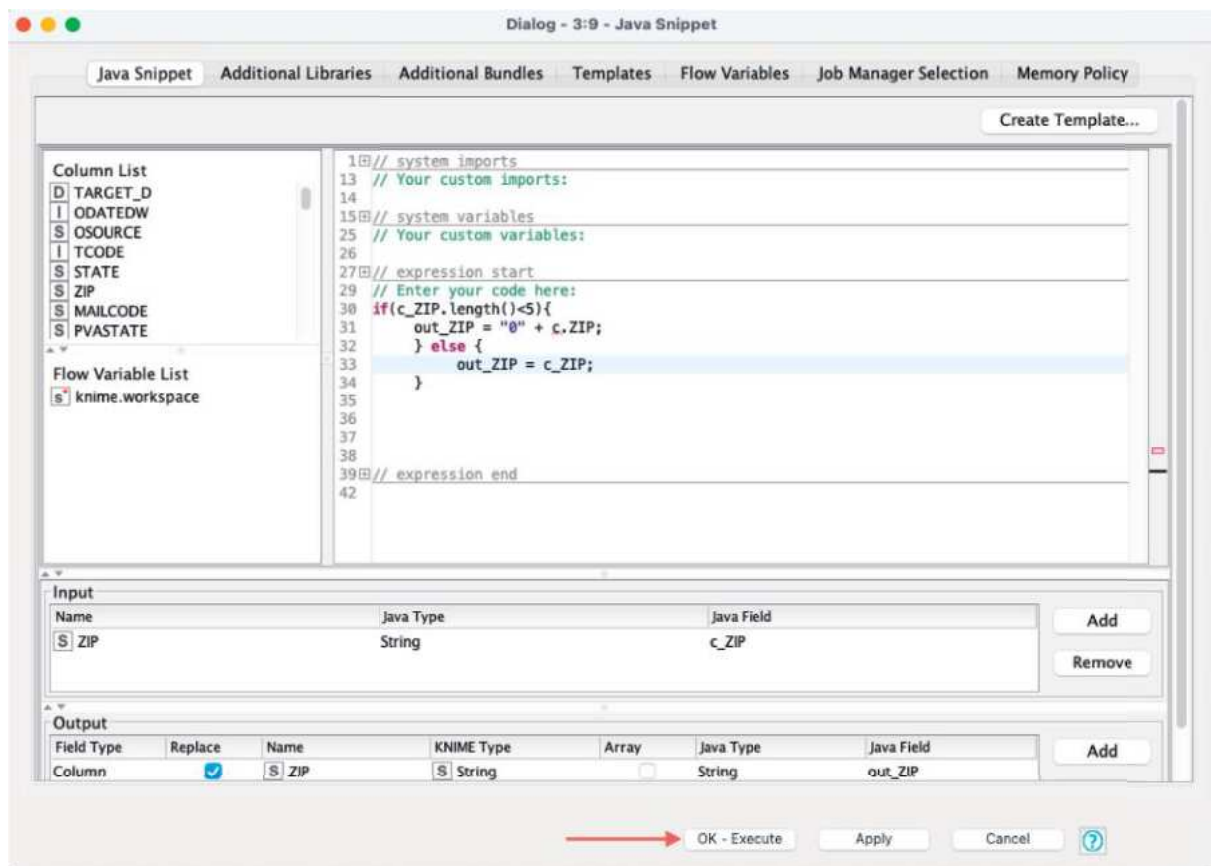


FIG. J.9 Adding leading zeros to 4-character ZIP codes.

Appended table - 3:9 - Java Snippet

File Edit Hilite Navigation View

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Row ID	I RECNO	S TARG...	D TARG...	I ODAT...	S OSOU...	I TCODE	S STATE	S ZIP	S MAILC...	S PVAST...	I DOB
Row10680	10681	0	0	8701	DRK	0	MA	01754	?	?	0
Row7833	7834	0	0	9501	ARG	0	MA	01876	?	?	5901
Row6030	6031	0	0	9401	MBC	0	MA	02038	?	?	5601
Row2980	2981	0	0	9501	AVN	0	MA	02124	?	?	0
Row12943	12944	0	0	9201	SSS	1	ME	04078	?	?	1301
Row367	368	0	0	9101	BHG	0	ME	04364	?	?	0
Row11631	11632	0	0	9601	CHT	0	ME	04438	?	?	0
Row3295	3296	0	0	9101	AML	1	VT	05033	?	?	2001
Row16577	16578	0	0	9501	HAM	0	VT	05663	?	?	6801
Row7755	7756	0	0	9601	HHN	1	CT	06335	?	?	4401
Row16542	16543	0	0	8601	DRK	2	CT	06460	?	?	0
Row6218	6219	1	11	8601	DRK	2	CT	06468	?	?	5001
Row6169	6170	0	0	8901	DUR	0	CT	06484	?	?	0
Row15022	15023	0	0	9401	STV	0	CT	06497	?	?	0
Row2887	2888	0	0	9401	L15	0	NJ	08008	?	?	2401
Row7730	7731	0	0	8601	BHG	2	NJ	08098	?	?	2001
Row2464	2465	0	0	9501	HAM	1	NJ	08225	?	E	5801
Row6916	6917	0	0	9501	HCC	2	NJ	08401	?	?	0
Row18847	18848	0	0	9101	DUR	0	NJ	08527	?	?	0
Row9987	9988	0	0	9101	UCA	1002	AE	09012	?	?	5801
Row6159	6160	0	0	9501	HHN	0	AE	09173	?	?	0
Row18609	18610	0	0	9001	PV3	1	AE	09182	?	?	5601
Row14910	14911	0	0	9501	DCD	14	AE	09459	?	?	6201
Row12995	12996	1	50	9501	FRC	0	AE	09645	?	?	4701
Row11058	11059	0	0	8701	FCR	1	NY	10001	?	?	3111
Row10546	10547	0	0	9201	LVT	1	NY	10014	?	?	1106
Row18033	18034	0	0	9401	HHH	0	NY	10962	?	?	0

FIG. J.10 The leading zeros data issue for the ZIP variable is fixed.

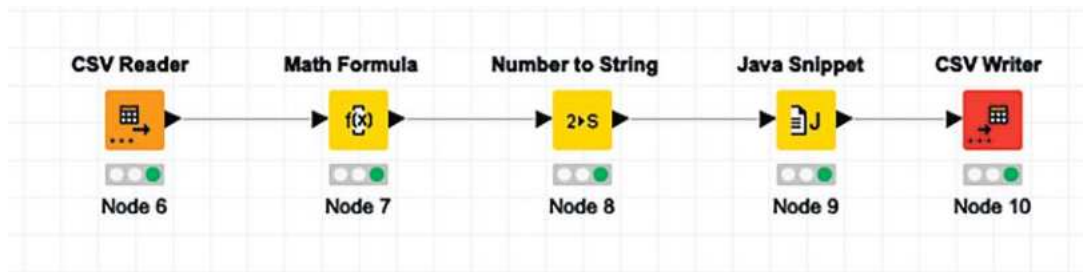


FIG. J.11 Final data cleaning workflow.

- g. After the `column("ZIP")` specification, add the string "`<5)`".
- h. Click on the **+ function** box to add a function and double-click on the **Join** function to add it to the expression.
- i. Click between the parentheses, enter a zero as "`0`" including the quotation marks, followed by a comma.
- j. Then click on the **+ Column** and double-click on the **ZIP** column.
- k. On a new line in the expression, enter "`else column("ZIP")`" (with no quotation marks).
- l. The final expression should look like [Figure J.8](#).
20. Right-click on the **Column Expressions** node and select **Output table**.
 - a. Click on ZIP header and select **Sort Ascending**.
The issue with the leading zeros was fixed, and all ZIP codes have five digits now as shown in [Fig. J.10](#).
 - b. Close the table.
21. On the **Node Repository** section, enter "**CSV**" into the search box, as you did for Tutorial H.
 - a. Double-click on the **CSV Write** node.
 - b. Connect the output triangle of the **Column Expressions** node to the left triangle of the **CSV Writer** node.
22. Right-click on the **CSV Writer** node and select **Configure**.
23. Double-click the **CSV Writer** node on the workflow.
 - a. Select the **Settings** tab in the *Configuration Dialog* screen and specify the output file location and name for **Output Location**.
 - b. Name the output file as **Tutorial-J_DataCleaning.csv** file. Make sure the file extension **.csv** is explicitly added to the file name; otherwise, KNIME will not recognize the file type.
 - c. Check the **Write column header** checkbox.
 - d. Select **overwrite** radio button in the **If exists**.
 - e. Click **OK** to return to the workflow screen.
24. Execute the **CSV Writer** node.
25. Click **File > Save** to save the workflow, which should be like as shown in [Fig. J.11](#).

K

Dummy coding category variables

Roberta Bortolotti, Senior Director, Nasdaq

All data is not created equally. Different from continuous numerical values, categorical data has its own characteristics, because they may be represented with textual values or a predefined scale with arbitrary ordering of the categories, which may not necessarily reflect any meaningful relationships among them. Typically, categorical data cannot be handled by machine learning algorithms designed to operate only with numerical data making encoding an important operation in data preparation.

To resolve for categorical variables, this tutorial walks through one technique, known as one-hot encoding, using dummy variables. A dummy variable is a binary variable coded as 1 or 0 to represent the presence or absence of a variable. The transformation of variables into dummy variables is indeed a technique used whenever modeling algorithms require numerical data.

There are other techniques used to handle categorical variables. The decision on which category to use depends on the use case. For example, both one-hot encoding and label encoding use numerical values to represent presence or absence; however, they have different properties and are applicable to different use cases.

- One-hot encoding is suitable when (1) the categories do not have any implied ordering or relationship with each other, and each category is treated as a separate entity with no relation to other categories; and (2) the number of categories is relatively small, so the number of columns does not become very large.
- On the other hand, label encoding is suitable when (1) the categories imply ordering or relationship with each other, such as with ordinal values ("small," "medium," and "large"); and (2) the number of categories is very large, as each ordinal value can reduce the dimensionality of the data.

Generally, one-hot encoding is more commonly used because of its flexibility, helping to avoid ambiguity and arbitrary ordering problems in machine learning applications.

1. Open **KNIME**.
2. Click on **File > New** on the top bar menu to create a new workflow, as you did in Tutorial I.
3. In the Wizard window, select **New KNIME Workflow** and click **Next**.
4. In the next screen, name the new workflow **Tutorial-K_Dummy-coding**. Click on **Browse** to specify a Tutorial folder, if necessary, and click **Finish**. The new workflow will display in KNIME Explorer as an active workflow marked with a green dot.
5. Add a **CSV Reader** node to the workflow space, as you did in Tutorial I.
 - a. Right-click on the **CSV Reader** node.
 - b. In the *Configuration Dialog* screen, click on **Browse** for "Input location:."
 - c. Navigate to the folder where you stored the output file for Tutorial J.
 - d. Select the **Tutorial-J_DataCleaning.csv** file.
 - e. Check the **Write column header** checkbox.
 - f. In the **Advanced Settings** tab, unclick the box to Limit data rows scanned.
 - g. Click **OK** to return to the workflow screen.
6. Execute the **CSV Reader** node.
7. Right-click on the **CSV Reader** node and select **File Table**.
8. Once the table displays, click on **Spec – Columns** tab.
 - a. Look for variable **TARGET_B** under **Columns**. Note that the data type for variables **TARGET_B** is of type **Number (integer)**, as shown in Fig. K.1.
 - b. Close the table.
9. On the **Node Repository** section, enter "Number to String" in the search box.
 - a. Double-click the **Number to String** node. The **Number to String** node converts numbers in a column (or a set of columns) into strings.
 - b. Connect the output triangle of the **CSV Reader** node to the left triangle of the **Number to String** node.

File Table - 5:10 - CSV Reader

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Columns: 220	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...	Value 0	Value 1	Value
RECNO	Number (integer)	0					1	19,049	?	?	?
TARGET_B	Number (integer)	1					0	1	?	?	?
TARGET_D	Number (double)	2					0	100	?	?	?
ODATEDW	Number (integer)	3					8,601	9,701	?	?	?
OSOURCE	String	4					?	?	?	?	?
TCODE	Number (integer)	5					0	72,002	?	?	?
STATE	String	6					?	?	IL	CA	NC
ZIP	Number (integer)	7					1,754	99,901	?	?	?
MAILCODE	String	8					?	?	B	?	?
PVASTATE	String	9					?	?	P	E	?
DOB	Number (integer)	10					0	9,508	?	?	?
NOEXCH	Number (integer)	11					0	1	?	?	?
RECINHSE	String	12					?	?	X	?	?
RECP3	String	13					?	?	X	?	?
RECPGVG	String	14					?	?	X	?	?
RECSWEEP	String	15					?	?	X	?	?
MDMAUD	String	16					?	?	XXXX	C1CM	D1CM
DOMAIN	String	17					?	?	T2	S1	R2
CLUSTER	Number (integer)	18					1	53	?	?	?
AGE	Number (integer)	19					2	98	?	?	?
AGEFLAG	String	20					?	?	E	I	?
HOMEOWNR	String	21					?	?	H	U	?
CHILD03	String	22					?	?	M	F	B
CHILD07	String	23					?	?	M	B	F
CHILD12	String	24					?	?	F	M	B
CHILD18	String	25					?	?	M	F	B
NUMCHLD	Number (integer)	26					1	7	?	?	?
INCOME	Number (integer)	27					1	?	?	?	?

FIG. K.1 Inspecting data type issue for categorical variable TARGET_B.

10. Right-click the **Number to String** node and select **Configure**. This step is similar to step 16 in Tutorial J.
 - a. In the *Configuration Dialog* screen, note that all variables are selected automatically as to be included.
 - b. Click on **Remove All**, using the «symbol.
 - c. From the **Excludes** column, select TARGET_B variables and click on the »symbol.
 - d. Note that TARGET_B variable is placed in the **Include** column.
 - e. Click OK.
11. Execute the **Number to String** node.
12. Right-click on the **Number to String** node and select **Transformed Input**.
13. Once the table displays, click on **Spec - Columns** tab.
 - a. Look for variable TARGET_B under **Columns**. Note that the data type for variables TARGET_B now shows of type **String** as shown in Fig. K.2.
 - b. Scroll down on the list to variable RFA_2A and note its values as E, G, F, and D, as shown in Fig. K.3. We will use this variable RFA_2A to derive the dummy values we need with a technique called one-hot label encoding.
 - c. Close the table.
14. On the **Node Repository** section, enter "One to Many" in the search box.
 - a. Double-click the **One to Many** node. The **One to Many** node transforms all possible values into a set of new columns where the value becomes the column name and the cell value is either 1 or 0, which is exactly what we need for the one-hot encoding of our categorical variable. If a row contains a missing value in a selected column, all corresponding new columns contain the value 0.
 - b. Connect the output triangle of the **Number to String** node to the left triangle of the **One to Many** node.
15. Right-click on the **One to Many** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, remove all variables and add RFA_2A the same way in step 11 above.
 - b. Check the **Remove included columns from output** checkbox and click OK. See Fig. K.4 for reference.
 - c. Execute the **One to Many** node.
16. Right-click on the **One to Many** node and select **Processed Data**.

Transformed input - 5:11 - Number to String

File

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Columns: 220	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...	Value 0
RECNO	Number (integer)	0					1	19,049	?
TARGET_B	String	1					?	?	0
TARGET_D	Number (double)	2					0	100	?
ODATEDW	Number (integer)	3					8,601	9,701	?
OSOURCE	String	4					?	?	?
TCODE	Number (integer)	5					0	72,002	?
STATE	String	6					?	?	IL
ZIP	Number (integer)	7					1,754	99,901	?
MAILCODE	String	8					?	?	B
PVASTATE	String	9					?	?	P
DOB	Number (integer)	10					0	9,508	?
NOEXCH	Number (integer)	11					0	1	?
RECINHSE	String	12					?	?	X
RECP3	String	13					?	?	X
RECPGVG	String	14					?	?	X
RECSWEEP	String	15					?	?	X
MDMAUD	String	16					?	?	XXXX
DOMAIN	String	17					?	?	T2
CLUSTER	Number (integer)	18					1	53	?
AGE	Number (integer)	19					2	98	?
AGEFLAG	String	20					?	?	E

FIG. K.2 Inspecting fixed data type issue for categorical variable TARGET_B.

Transformed input - 5:11 - Number to String

File

Table "default" - Rows: 19049 Spec - Columns: 220 Properties Flow Variables

Columns: 220	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...	Value 0	Value 1	Value 2	Value 3
MAXRDATE	Number (integer)	180					8,403	9,702	?	?	?	?
LASTGIFT	Number (double)	181					0	1,000	?	?	?	?
LASTDATE	Number (integer)	182					9,503	9,702	?	?	?	?
FISTDATE	Number (integer)	183					7,211	9,603	?	?	?	?
NEXTDATE	Number (integer)	184					7,211	9,702	?	?	?	?
TIMELAG	Number (integer)	185					0	90	?	?	?	?
AVGGIFT	Number (double)	186					1,574	1,000	?	?	?	?
CONTROLN	Number (integer)	187					7	191,763	?	?	?	?
HPHONE_D	Number (integer)	188					0	1	?	?	?	?
RFA_2R	String	189					?	?	L	?	?	?
RFA_2F	Number (integer)	190					1	4	?	?	?	?
RFA_2A	String	191					?	?	E	G	F	D
MDMAUD_R	String	192					?	?	X	C	D	L
MDMAUD_F	Number (integer)	193					1	5	?	?	?	?
MDMAUD_A	String	194					?	?	X	C	M	L
CLUSTER2	Number (integer)	195					1	62	?	?	?	?
GEOCODE2	String	196					?	?	C	A	D	B
ADATE_2	Number (integer)	197					9,704	9,706	?	?	?	?
ADATE_3	Number (integer)	198					9,604	9,606	?	?	?	?
ADATE_4	Number (integer)	199					9,511	9,609	?	?	?	?
ADATE_5	Number (integer)	200					9,604	9,604	?	?	?	?

FIG. K.3 Inspecting issue with values of categorical variable RFA_2A to apply dummy encoding.

17. When the table displays, click on the **Spec - Columns** tab.
 - a. Scroll down to the bottom of the list and notice that the RFA_2A values are now columns (Fig. K.5). The original column name needs to be added as a prefix in the column names so that they are more meaningful.
 - b. Close the table.
18. On the **Node Repository** section, enter "column renamer" in the search box.
 - a. Double-click the **Column Renamer** node and connect it to the **One to Many** node.

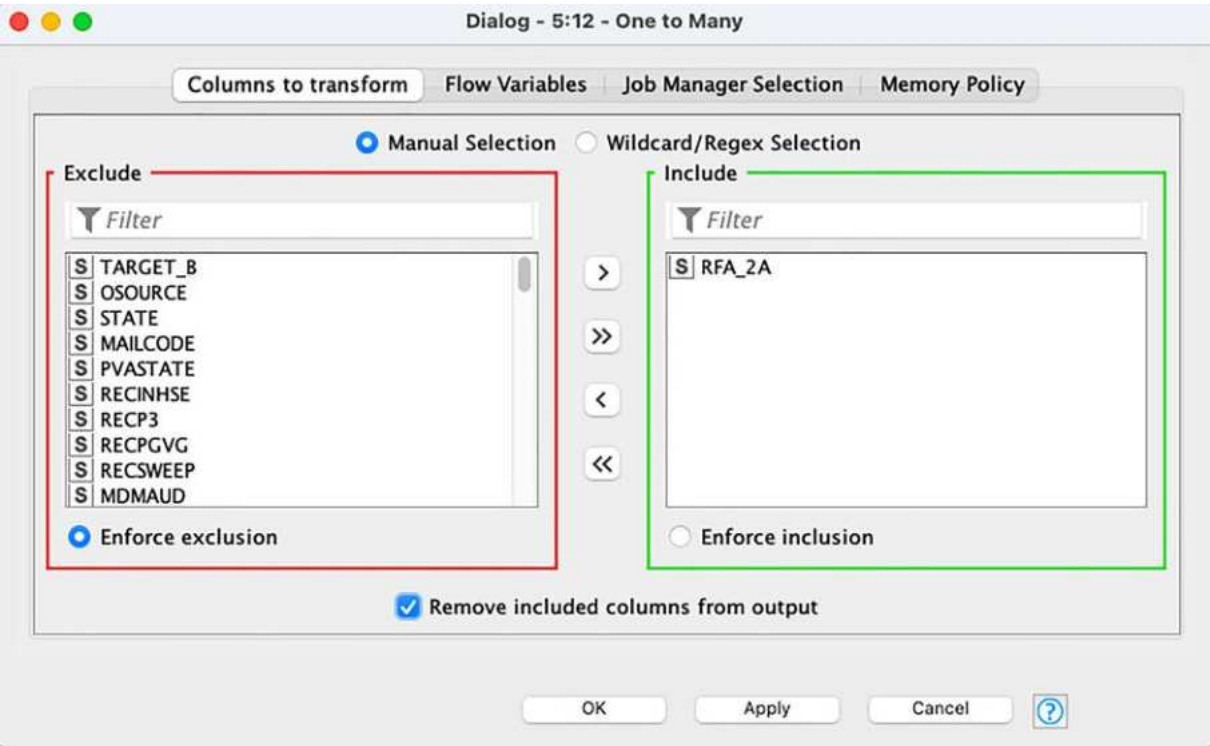


FIG. K.4 Configuring one-hot encoding for variable RFA_2A.

The screenshot shows a software window titled "Processed data - 5:12 - One to Many". It has a "File" menu and four tabs: "Table 'default' - Rows: 19049", "Spec - Columns: 223", "Properties", and "Flow Variables". The "Spec - Columns: 223" tab is active. It displays a table with 8 columns: "Columns: 223", "Column Type", "Column In...", "Color Han...", "Size Han...", "Shape Ha...", "Filter Han...", and "Lower Bo...". The table lists 22 columns, all of which are "Number (integer)" type. The first 24 rows correspond to ADATE_7 through ADATE_24, with values ranging from 201 to 218. The last four rows correspond to E, G, F, and D, with values 219, 220, 221, and 222 respectively. The "Lower Bo..." column shows values for each row, such as 9,512 for ADATE_7 and 0 for E, G, F, and D.

Columns: 223	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...
ADATE_7	Number (integer)	201					9,512
ADATE_8	Number (integer)	202					9,511
ADATE_9	Number (integer)	203					9,509
ADATE_10	Number (integer)	204					9,510
ADATE_11	Number (integer)	205					9,508
ADATE_12	Number (integer)	206					9,507
ADATE_13	Number (integer)	207					9,502
ADATE_14	Number (integer)	208					9,504
ADATE_15	Number (integer)	209					9,504
ADATE_16	Number (integer)	210					9,502
ADATE_17	Number (integer)	211					9,501
ADATE_18	Number (integer)	212					9,409
ADATE_19	Number (integer)	213					9,409
ADATE_20	Number (integer)	214					9,411
ADATE_21	Number (integer)	215					9,409
ADATE_22	Number (integer)	216					9,408
ADATE_23	Number (integer)	217					9,312
ADATE_24	Number (integer)	218					9,405
E	Number (integer)	219					0
G	Number (integer)	220					0
F	Number (integer)	221					0
D	Number (integer)	222					0

FIG. K.5 Columns derived from the values of the RFA_2A variable.

Dialog - 5:13 - Column Renamer

Column	New name
E	RFA_2A_E
G	RFA_2A_G
F	RFA_2A_F
D	RFA_2A_D

+ Add column

Cancel Ok and Execute

FIG. K.6 Renaming the new derived columns for variable RFA_2A.

19. Right-click the **Column Renamer** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, select E from the Column drop-down list.
 - b. In the New Name field, the new column name should be the prefix "RFA_2A_" + the column name. Enter "RFA_2A_E."
 - c. Repeat steps (a) and (b) above for each of the new columns (E, G, F, and D). The result should be as shown in Fig. K.6.
 - d. Click **OK and Execute**.
 - e. Execute the **Column Renamer** node.
20. Right-click on the **Column Renamer** node and select **Output table**.
 - a. Once the table displays, click on **Spec – Columns** tab.
 - b. Scroll down to the end of the list to see the re-named columns, as shown in Fig. K.7.
 - c. Scroll up to the variable RFA_2R. Note that it only has value as L, which doesn't provide any distinguishing information, and it will just confuse the model. Because of that, this variable should be deleted (Fig. K.8).
 - d. Close the table.
21. On the **Node Repository** section, enter "column filter" in the search box.
 - a. Double-click the **Column Filter** node and connect it to the **Column Renamer** node.
22. Right-click the **Column Filter** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, exclude the RFA_2R variable from the **Include** column by selecting it and using the < symbol. The Configuration should look like Fig. K.9.
 - b. Click **OK and Execute**.

Output Table - 5:13 - Column Renamer

File

Table "default" - Rows: 19049 Spec - Columns: 223 Properties Flow Variables

Columns: 223	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...
ADATE_6	Number (integer)	200					9,601	9,603
ADATE_7	Number (integer)	201					9,512	9,602
ADATE_8	Number (integer)	202					9,511	9,603
ADATE_9	Number (integer)	203					9,509	9,511
ADATE_10	Number (integer)	204					9,510	9,511
ADATE_11	Number (integer)	205					9,508	9,511
ADATE_12	Number (integer)	206					9,507	9,510
ADATE_13	Number (integer)	207					9,502	9,507
ADATE_14	Number (integer)	208					9,504	9,506
ADATE_15	Number (integer)	209					9,504	9,504
ADATE_16	Number (integer)	210					9,502	9,504
ADATE_17	Number (integer)	211					9,501	9,503
ADATE_18	Number (integer)	212					9,409	9,508
ADATE_19	Number (integer)	213					9,409	9,411
ADATE_20	Number (integer)	214					9,411	9,412
ADATE_21	Number (integer)	215					9,409	9,410
ADATE_22	Number (integer)	216					9,408	9,506
ADATE_23	Number (integer)	217					9,312	9,407
ADATE_24	Number (integer)	218					9,405	9,406
RFA_2A_E	Number (integer)	219					0	1
RFA_2A_G	Number (integer)	220					0	1
RFA_2A_F	Number (integer)	221					0	1
RFA_2A_D	Number (integer)	222					0	1

FIG. K.7 Renamed derived columns for variable RFA_2A.

Output Table - 5:13 - Column Renamer

File

Table "default" - Rows: 19049 Spec - Columns: 223 Properties Flow Variables

Columns: 223	Column Type	Column In...	Color Han...	Size Han...	Shape Ha...	Filter Han...	Lower Bo...	Upper Bo...	Value 0	Value
NGIFTALL	Number (integer)	175					1	95	?	?
CARDGIFT	Number (integer)	176					0	41	?	?
MINRAMNT	Number (double)	177					0	1,000	?	?
MINRDATE	Number (integer)	178					7,506	9,702	?	?
MAXRAMNT	Number (double)	179					5	5,000	?	?
MAXRDATE	Number (integer)	180					8,403	9,702	?	?
LASTGIFT	Number (double)	181					0	1,000	?	?
LASTDATE	Number (integer)	182					9,503	9,702	?	?
FISTDATE	Number (integer)	183					7,211	9,603	?	?
NEXTDATE	Number (integer)	184					7,211	9,702	?	?
TIMELAG	Number (integer)	185					0	90	?	?
AVGGIFT	Number (double)	186					1.574	1,000	?	?
CONTROLN	Number (integer)	187					7	191,763	?	?
HPHONE_D	Number (integer)	188					0	1	?	?
RFA_2R	String	189					?	?	L	?
RFA_2F	Number (integer)	190					1	4	?	?
MDMAUD_R	String	191					?	?	X	C
MDMAUD_F	Number (integer)	192					1	5	?	?
MDMAUD_A	String	193					?	?	X	C
CLUSTER2	Number (integer)	194					1	62	?	?
GEOCODE2	String	195					?	?	C	A
ADATE_2	Number (integer)	196					9,704	9,706	?	?
ADATE_3	Number (integer)	197					9,604	9,606	?	?

FIG. K.8 Variable with no relevant information identified for deletion.

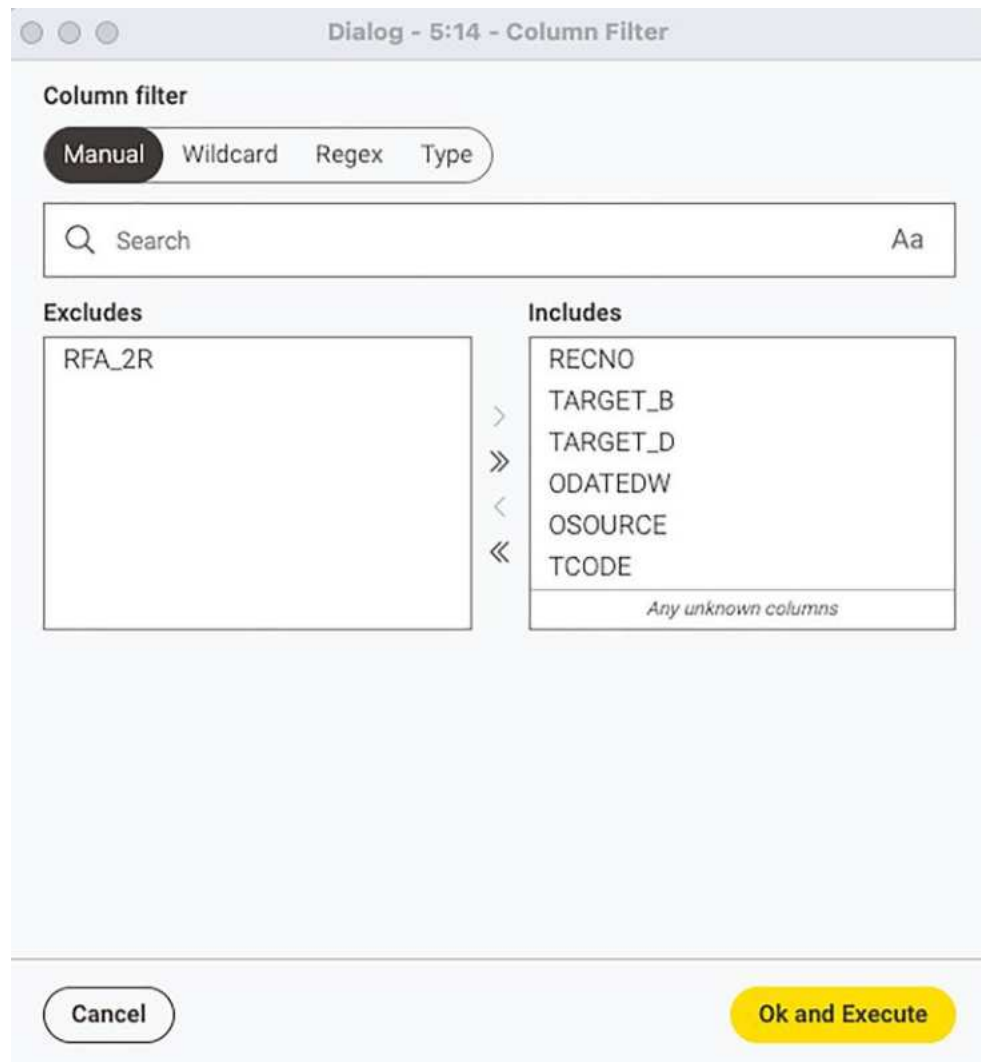


FIG. K.9 Configuration of the Column Filter node.

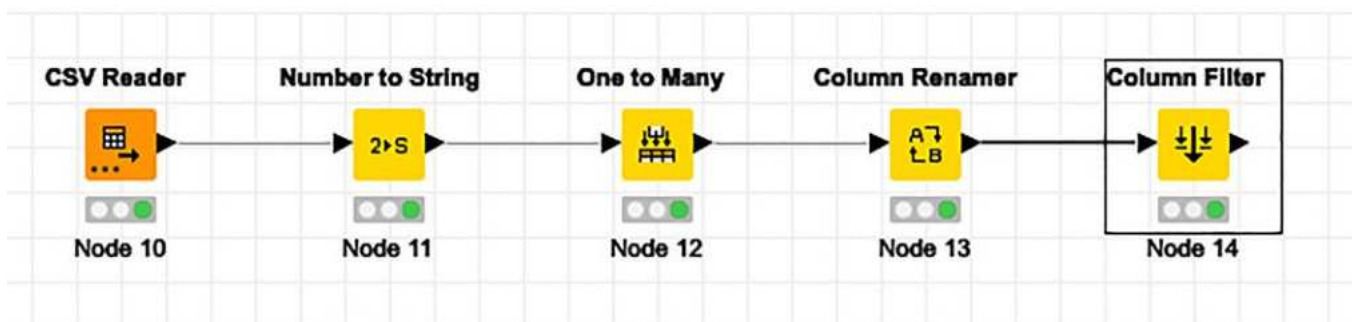


FIG. K.10 Dummy Coding workflow.

23. Right-click the **Column Filter** node and select **Filtered table**.
24. Once the table displays, click on **Spec – Columns** tab.
 - a. Scroll down the list and notice that the variable RFA_2A is not listed as one of the variables anymore. The variable was deleted from the dataset.
 - b. Close the table.
25. Click **File > Save** to save the workflow, which should be like as shown in Fig. K.10.
26. You will continue this workflow in Tutorial-L.
27. Close the KNIME application.

L

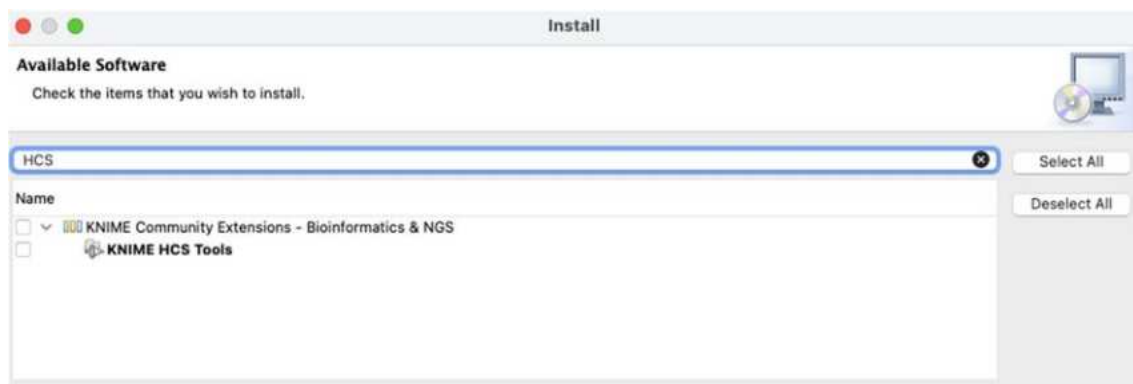
Outlier handling

Roberta Bortolotti, Senior Director, Nasdaq

Removal of outliers is the simplest type of data filtering. Even though outliers should be kept in the dataset sometimes, because they represent natural variations in the population, such as in the case of modeling of credit risk, fraud, and other rare events, there are outliers that are problematic, because they represent unnecessary or incorrect information, such as measurement errors, data entry, and processing errors or poor sampling, and

should be removed. Unnecessary outliers are noise in the data and mostly can reduce the predictability of the model; therefore their removal can not only improve the accuracy of the predictions but also produce better visualization, making it easier to identify patterns and trends.

This tutorial shows an automatic and manual way to remove outliers. We will continue building on the workflow of Tutorial K as part of the data preparation process.



1. Open the workflow of **Tutorial-K_Dummy-coding**.
2. On the **Node Repository** section, enter “**statistics**” in the search box.
 - a. Double-click the **Statistics** node. The **Statistics** node outputs the basic descriptive statistical information that will permit the determination of how the data should be handled and prepared for modeling. For example, analyzing the values of the minimum, maximum, and mean can hint at outliers in the dataset. In case a suspicious situation is present, looking at a frequency table or histogram depicts how the data values are distributed, which will confirm any outlier in the dataset.
 - b. Connect the output triangle of the **Column Filter** node to the left triangle of the **Statistics** node.
3. Right-click on the **Statistics** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, exclude all variables from the **Include** column by using the << symbol.
 - b. From the **Excludes** column, select the HIT variable, and click on the > symbol to move it to the **Include** column.
 - c. Set the “**Max no. of possible values per column (in output table):**” field to be 20,000, as shown in [Fig. L.1](#).
 - d. Click **OK - Execute**.
4. Right-click on the **Statistics** node and select **Occurrences Table**.
 - a. Select the HIT variable header and click to sort in descending order.

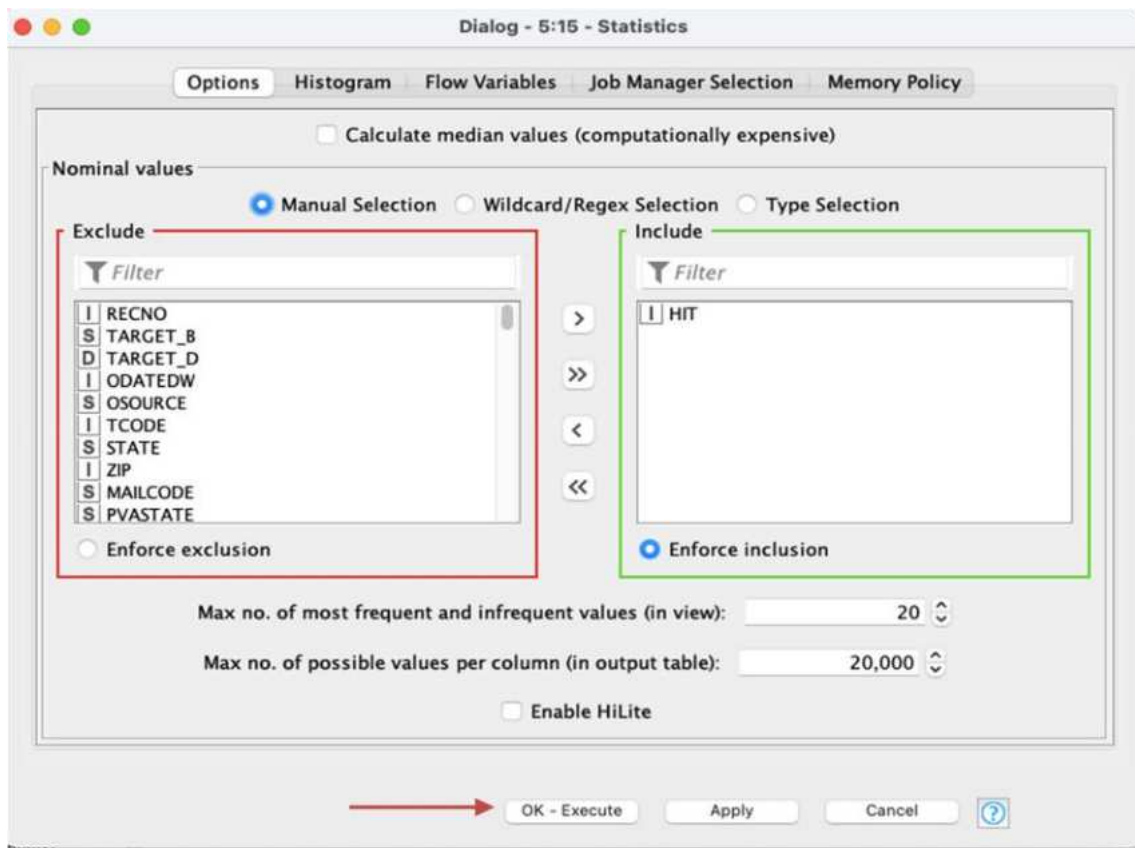
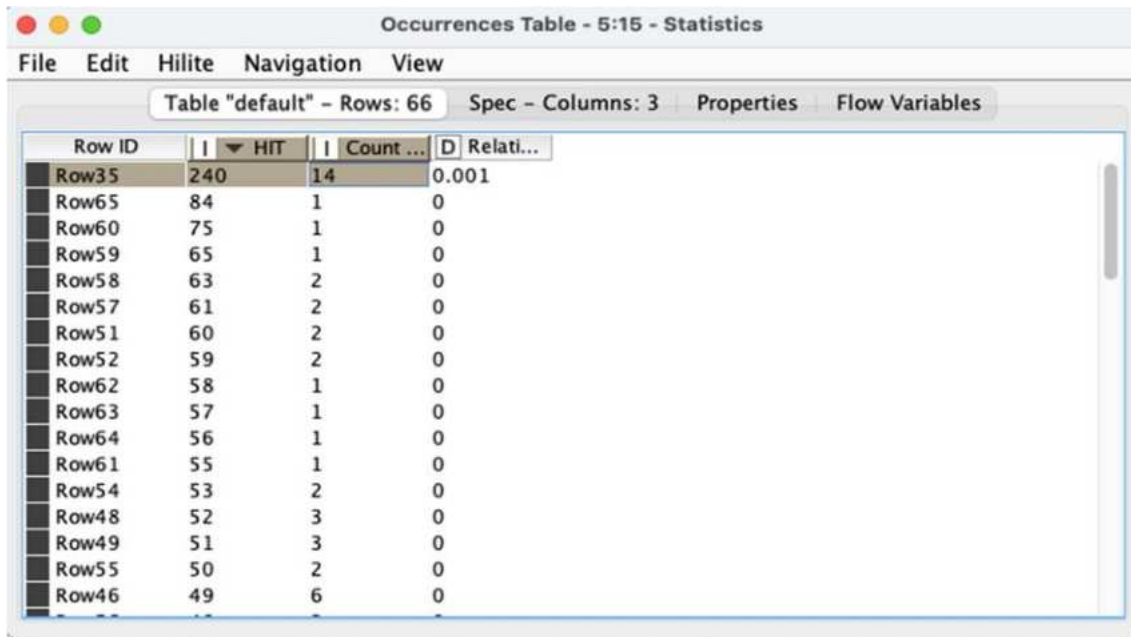


FIG. L.1 Configuration for the statistics measures for the HIT variable.

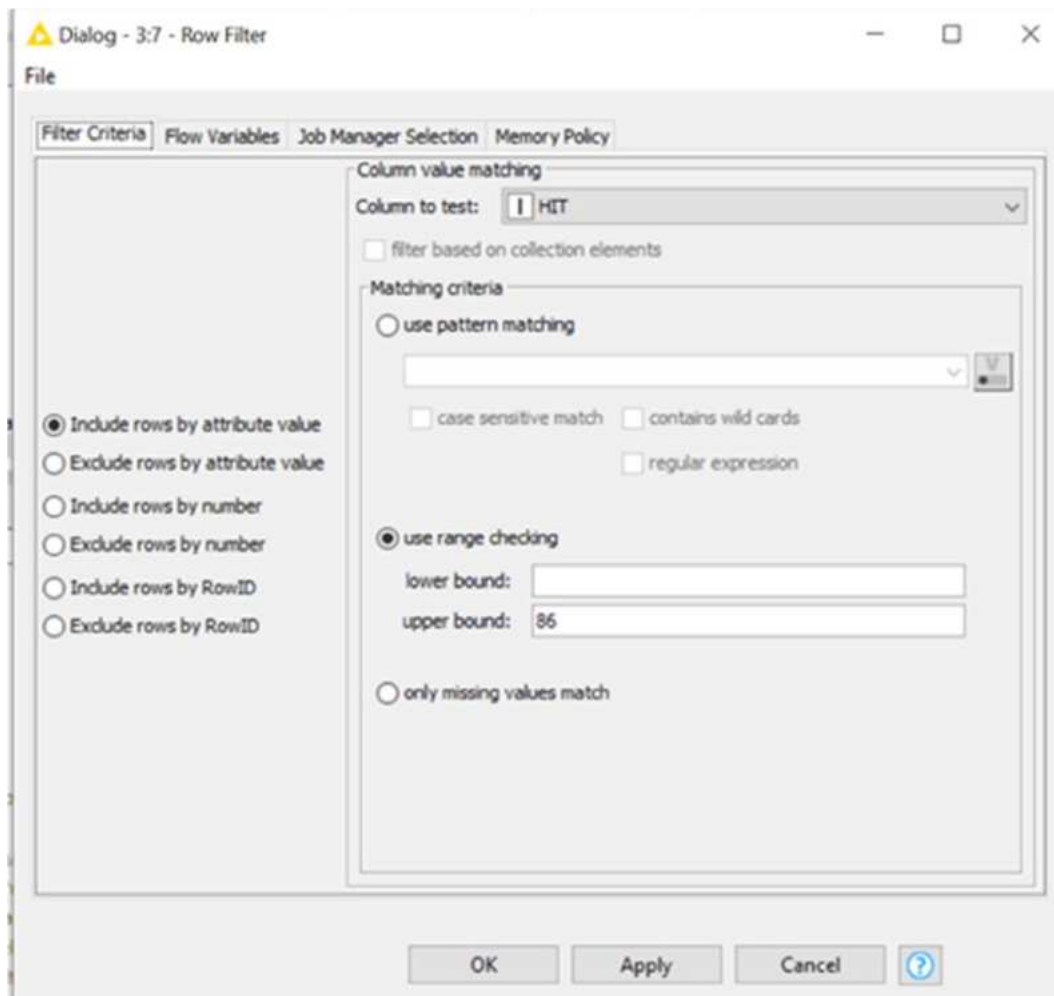
Note that 14 records have a value of 240. Since the next highest value is 84, a value of 240 is a strong indicator that it is an outlier perhaps caused by an input error. There are 14 outlier records that should be removed, as shown in Fig. L.2.

- b. Close the table.
5. On the **Node Repository** section, enter "row filter" in the search box.
 - a. Double-click the **Row Filter** node.
 - b. Connect the output triangle of the **Column Filter** node to the left triangle of the **Row Filter** node. This node will remove all rows above a specified maximum value.
6. Right-click on the **Row Filter** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, set the Column to test to the **as HIT** column.
 - b. Click the **use range** checking radio button and enter 86 in the **upper** bound box, as shown in Fig. L.3.
 - c. Click OK and execute the node.
7. Right-click on the executed **Row Filter** node and select the **Filtered** option. Sort the HIT variable in descending order to see in Fig. L.4 that all the rows with 240 in them were removed from the data set.
 - a. Close the table.
8. Now we will handle the outliers by replacing their values with a specified number. On the **Node Repository** section, enter "math" in the search box.
 - a. Double-click the **Math Formula** node.
 - b. Connect the output triangle of the **Column Filter** node to the left triangle of the **Math Formula** node.
9. Right-click on the **Math Formula** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, place the cursor in the **Expression** box and, similar to Tutorial J, Step 11, specify the expression `if( $HIT$ > 84, 84,$HIT$ )`.
 - b. This expression replaces rows with HIT values > 84 with the value = 84.
 - c. Select **Replace Column** and then select HIT from the dropdown box as shown in Fig. L.5.
 - d. Click **OK - Execute**.



Row ID	HIT	Count	Relati...
Row35	240	14	0.001
Row65	84	1	0
Row60	75	1	0
Row59	65	1	0
Row58	63	2	0
Row57	61	2	0
Row51	60	2	0
Row52	59	2	0
Row62	58	1	0
Row63	57	1	0
Row64	56	1	0
Row61	55	1	0
Row54	53	2	0
Row48	52	3	0
Row49	51	3	0
Row55	50	2	0
Row46	49	6	0

FIG. L.2 Identification of outlier records for variable HIT.



Dialog - 3:7 - Row Filter

File

Filter Criteria Flow Variables Job Manager Selection Memory Policy

Column value matching

Column to test: HIT

☐ filter based on collection elements

Matching criteria

☐ use pattern matching

☐ case sensitive match ☐ contains wild cards

☐ regular expression

☒ use range checking

lower bound:

upper bound: 85

☐ only missing values match

OK Apply Cancel ?

FIG. L.3 Use of the Row Filter node to remove outlier rows.

Table "default" - Rows: 19035 Spec - Columns: 200 Properties Flow Variables

Row ID	CHILD03	CHILD07	CHILD12	CHILD18	NUMCHLD	INCOME	GENDER	WEALTH1	HIT	MBCRAFT	MBGAR...	MBBOOKS	MBCOL...	MAGPAM	MAGFEM
Row 18220	?	?	?	?	?	5	M	3	84	2	2	9	2	4	0
Row 10421	?	?	?	?	?	4	M	8	75	3	1	9	1	3	0
Row 1003	?	?	?	?	?	5	M	9	65	0	0	9	0	2	0
Row 13397	?	?	?	?	?	5	F	3	63	1	1	9	1	3	1
Row 17944	?	?	?	?	?	6	U	8	63	4	0	9	0	5	1
Row 10013	?	?	?	?	?	5	F	3	61	2	0	4	2	2	2
Row 11473	?	?	?	?	?	7	M	6	61	1	1	7	2	0	0
Row 4	?	?	?	?	1	3	F	2	60	1	0	9	0	4	1
Row 4029	?	M	M	M	?	3	M	3	60	1	0	9	1	1	0
Row 3304	?	?	?	?	?	4	M	8	59	0	0	8	0	4	2
Row 14029	?	?	?	?	1	4	F	4	59	3	2	9	0	3	0
Row 14262	?	?	?	?	?	5	M	9	58	0	0	9	0	0	0
Row 15602	?	?	?	?	?	7	F	9	57	0	0	8	1	1	0
Row 17708	?	?	?	?	?	1	F	3	56	2	4	9	1	2	1
Row 13330	?	?	?	?	?	5	M	3	55	1	0	9	5	4	0
Row 6664	?	?	?	?	?	4	F	4	53	1	1	9	0	2	1
Row 9840	?	?	?	?	?	3	F	9	53	0	0	9	0	2	3
Row 2016	?	?	?	?	1	3	M	4	52	0	0	9	0	1	1
Row 4188	?	?	?	?	?	7	M	0	52	0	0	9	0	1	1
Row 4592	?	?	?	?	2	7	M	9	52	3	1	8	2	1	0
Row 4063	?	?	?	?	?	2	F	6	51	0	1	9	0	1	1
Row 16164	?	?	?	?	?	6	M	3	51	1	2	9	1	2	0
Row 10464	?	?	?	?	?	5	M	5	51	1	2	7	0	0	1
Row 6804	?	?	?	?	?	2	M	8	50	0	0	8	0	0	0
Row 12708	?	?	?	?	?	2	F	2	50	2	0	7	1	4	0
Row 5724	?	?	?	?	?	3	M	2	49	2	2	9	0	2	0
Row 6859	?	?	?	M	?	7	M	7	49	0	0	6	1	5	0
Row 8550	?	?	?	?	?	1	F	5	49	0	0	3	0	1	1
Row 10027	?	?	?	?	?	7	M	6	49	2	0	7	1	0	1
Row 15933	?	?	?	?	?	5	M	6	49	0	2	8	0	4	1
Row 16530	?	?	?	?	?	2	F	8	49	2	0	5	0	1	1
Row 7654	?	?	?	?	?	1	F	1	48	1	3	9	0	2	2
Row 14645	?	?	?	?	?	3	F	7	48	1	1	9	1	2	2
Row 5315	?	?	?	?	?	1	M	7	47	1	2	9	0	2	0
Row 10224	?	?	?	?	?	6	M	8	47	1	0	7	0	3	2
Row 14440	?	?	?	?	?	2	M	8	47	1	0	9	0	1	0
Row 3402	?	?	?	?	?	3	M	6	46	3	0	8	0	1	2

FIG. L.4 Outliers with HIT values = 240 are removed from the output of the Row Filter node.

Dialog - 3:17 - Math Formula

Math Expression Flow Variables Job Manager Selection Memory Policy

Column List

- TCODE
- ZIP
- DOB
- NOEXCH
- CLUSTER
- AGE
- NUMCHLD
- INCOME
- WEALTH1
- HIT
- MBCRAFT
- MBGARDEN
- MBBOOKS
- MBCOLLECT

Flow Variable List

Category: All

Description: If-Condition: if (cond, trueval, falseval)

Function

- sqrt(x)
- rand()
- mod(x, y)
- if(x, y, z)**
- round(x)
- round(x, y)
- roundHalfUp(x, y)
- roundHalfDown(x, y)
- ceil(x)
- floor(x)
- binom(x, v)

Expression

1 if(\$HIT\$>84, 84, \$HIT\$)

Append Column: ☐

Replace Column: ☒ HIT

Convert to Int: ☐

OK - Execute Apply Cancel

FIG. L.5 Configuration of the expression to remove outliers for variable HIT.

Row ID	S	GENDER	I	WEAL...	D	HIT	I	MBCR...	I	MBGA...	I	MBBO...	I	MBCO...	I	MAGF.
Row3171		F		2		84		0		0		1		?		0
Row3237		F		4		84		0		0		0		?		0
Row3617		M		1		84		1		0		9		?		2
Row4414		F		9		84		0		0		0		?		0
Row10163		F		1		84		0		0		0		?		0
Row10207		M		6		84		0		0		0		?		0
Row10824		M		7		84		0		0		3		0		1
Row11406		F		5		84		0		0		1		?		1
Row11490		F		2		84		1		0		4		?		0
Row13080		F		7		84		0		0		6		?		2
Row14670		F		9		84		0		0		1		?		1
Row15794		M		4		84		0		0		1		?		0
Row18220		M		3		84		2		2		9		2		4
Row10421		M		8		75		3		1		9		1		3
Row1003		M		9		65		0		0		9		0		2
Row13397		F		3		63		1		1		9		1		3
Row17944		U		8		63		4		0		9		0		5
Row10013		F		3		61		2		0		4		2		2
Row11473		M		6		61		1		1		7		2		0
Row4		F		2		60		1		0		9		0		4
Row4029		M		3		60		1		0		9		1		1
Row3304		M		8		59		0		0		8		0		4
Row14039		F		4		59		3		2		9		0		3
Row14263		M		9		58		0		0		9		0		0
Row15607		F		9		57		0		0		8		1		1

FIG. L.6 Removal of outliers using manual technique.

10. Right-click on the **Math Formula** node and select **Output Data**.
 - a. Scroll to the right to the HIT variable header and sort by descending order. The sorted display shown in Fig. L.6 shows that the highest HIT value now is 84. This is much better than the previous technique we used, because it doesn't remove 14 rows.
 - b. Close the table.
11. On the **Node Repository** section, enter "CSV" into the search box, as you did for Tutorial H.
 - a. Double-click on the **CSV Writer** node.
 - b. Connect the output triangle of the **Math Formula** node to the left triangle of the **CSV Writer** node.
12. Right-click on the **CSV Writer** node and select **Configure**.
 13. Double-click the **CSV Writer** node on the workflow.
 - a. Select the **Settings** tab in the *Configuration Dialog* screen and specify the output file location and name for **Output Location**.
 - b. Name the output file as **Tutorial-L_Outlier Handling.csv**. Make sure the file extension **.csv** is explicitly added to the file name; otherwise, KNIME will not recognize the file type.
 - c. Check the **Write column header** checkbox.
 - d. Select **overwrite** radio button in the **If exists:** section.
 - e. Click **OK** to return to the workflow screen.
 14. Execute the **CSV Writer** node.
 15. Click **File > Save** to save the workflow, which should be as shown in Fig. L.7.
 16. Close the KNIME application.

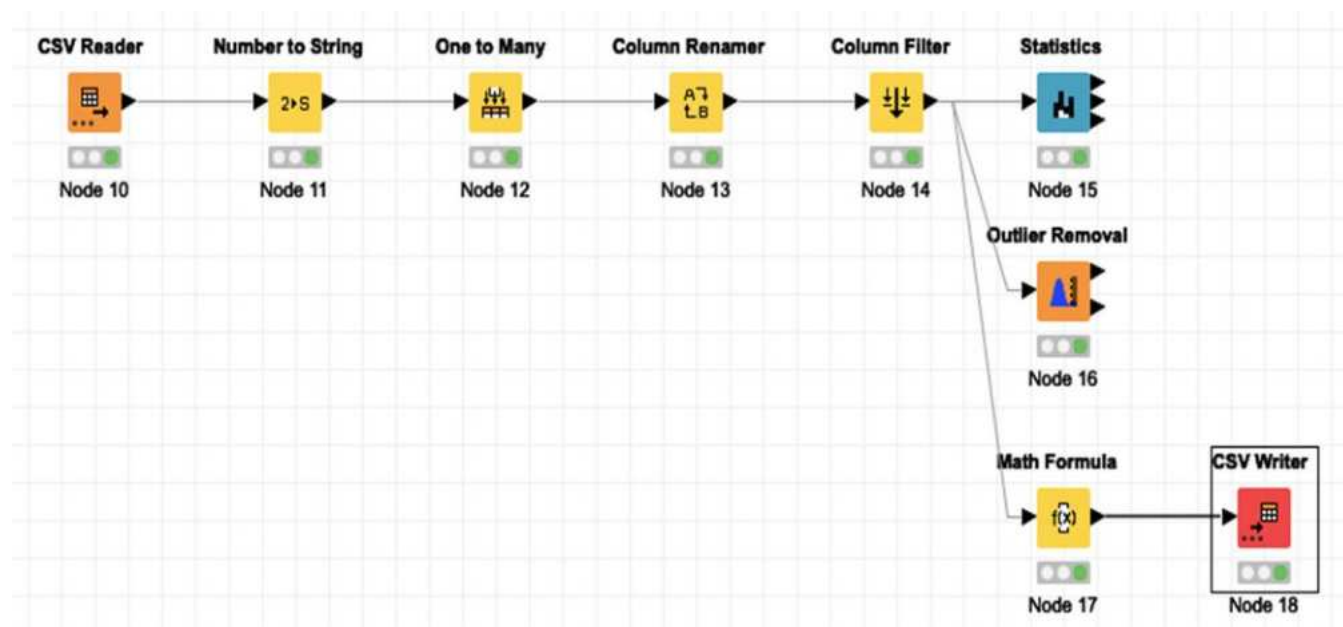


FIG. L.7 KNIME workflow for handling of outlier values.

M

Filling missing values with constants

Roberta Bortolotti, Senior Director, Nasdaq

Filling missing values with a suitable data value for a variable is an operation called data imputation. The task of deciding which value to use to fill in blanks in a record should follow rules or a set of rules defined based on assumptions about the pattern of the data. There is a drawback in this operation, though: data added to the pattern might not reflect the underlying pattern of the complete data.

Handling missing data inappropriately can lead to biased conclusions and consequently to wrong business decisions. Determining the type of missing data, (1) Missing Completely at Random (MCAR), (2) Missing at Random (MAR), and (3) Missing Not at Random (MNAR), will help to choose which imputation technique to use.

The next set of Tutorials, M through O, will walk through three imputation techniques that show data imputation operations by:

- a. Filling missing values with constants (Tutorial M).
 - b. Filling missing values with formulas (Tutorial N).
 - c. Filling missing values with a model (Tutorial O).
1. Repeat steps 1 to 6 from the previous tutorials to create a new KNIME workflow and to read the CSV file to be worked throughout this tutorial.
 - a. Name the workflow as **Tutorial-M_Filling_MV_w_Constants**.
 - b. Add a CSV Reader node to the new workflow.
 - c. Use the output file **Tutorial-L_Outlier_Handling.csv** from Tutorial L to load the data into the **CSV Reader** node.
 - d. Be sure to unclick the **Advanced Settings** option to limit the rows scanned, as you did in Tutorial-L
 - e. Execute the **CSV Reader** node.
 2. Right-click on the **CSV Reader** node and select **File table**.
 3. When the table displays, click on the **Spec – Columns** tab.
 - a. Look through the variables and note that there are records that contain missing values for a lot of variables, denoted as “?”.

- b. Scroll over to the **MBCRAFT** column, as shown in [Fig. M.1](#).

Variables from **MBCRAFT** to **PUBOPP** show the number of times different mail promotions were accepted, which is important information for our model. They will be used in this exercise to show how to fill missing values with constant values.

- c. Close the table.
4. Click on the **CSV Reader** node to highlight it.
5. On the **Node Repository** section, enter “**missing**” in the search box.
 - a. Double-click the **Missing Value** node under the **Transform** folder and connect it automatically to the **CSV Reader** node.
6. Right-click on the **Missing Value** node and select **Configure**.
 - a. On the **Default** tab, select the **Fix Value** option from the drop-down list for **Number (integer)**. The default value is 0 and it shows automatically; leave it as 0.
 - b. Select the **Fix Value** option from the drop-down list for **String**, then enter the value **U** for unknown. Select the **Fix Value** option from the dropdown list for **Number (double)**. The default value is 0.0 and it shows automatically; leave it as 0.0.
 - c. The configured node is shown in [Fig. M.2](#).
 - d. Click **OK and execute the node**.
7. Right-click on the **Missing Value** node and select **Output table**, as shown on [Fig. M.3](#).
 - a. Look at the values of variables from **MBCRAFT** to **PUBOPP**. Note that all missing values were filled with a constant and that there are no more missing values for those variables in the dataset.
 - b. Close the table.
8. Click **File > Save** to save the workflow.
9. Close the KNIME application.

File Table - 3:12 - CSV Reader

Table "default" - Rows: 19049 Spec - Columns: 222 Properties Flow Variables

Row ID	HIT	MBCK...	MBGA...	MBBO...	MBCO...	MAGF...	MAGF...	MAGM...	PUBG...	PUBC...	PUBH...	PUBD...	PUBN...	PUBP...	PUBOPP	DATA...
Row0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0
Row1	16	0	0	3	1	1	1	0	0	0	2	0	3	0	0	3
Row2	2	0	0	1	0	0	0	0	0	0	1	0	0	0	0	3
Row3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3
Row4	60	1	0	9	0	4	1	0	0	0	4	0	1	0	1	3
Row5	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row6	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Row8	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row9	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row10	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row11	3	0	0	1	0	0	0	0	0	0	1	0	0	0	0	3
Row12	16	1	1	1	0	1	0	0	0	0	0	1	0	0	0	3
Row13	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row14	17	0	0	1	0	1	0	0	0	0	1	1	5	0	1	3
Row15	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row16	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row17	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row18	6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	3
Row19	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row20	10	0	0	0	1	1	0	0	0	0	0	0	0	0	0	3
Row21	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row22	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row23	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row24	3	1	0	0	0	0	0	0	0	0	0	1	0	0	0	3
Row25	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3
Row26	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Row27	7	1	1	0	0	0	0	0	0	0	0	1	1	0	0	3
Row28	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
Row29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

FIG. M.1 List of variables with missing data.

Dialog - 3:13 - Missing Value

Default Column Settings Flow Variables Job Manager Selection

Number (integer) Fix Value Value 0

Number (double) Fix Value Value 0.0

String Fix Value Value U

Options marked with an asterisk (*) will result in non-standard PMML.

OK - Execute Apply Cancel

FIG. M.2 Configuration for missing value imputation with a constant.

Output table - 3:13 - Missing Value

File Edit Hilite Navigation View

Table "default" - Rows: 19049 Spec - Columns: 222 Properties Flow Variables

Row ID	MBCK...	MEGA...	MBBO...	MBCO...	MACF...	MACF...	MAGM...	PUBG...	PUBC...	PUBHL...	PUBD...	PUBN...	PUBP...	PUBOPP	DATA...
Row0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Row1	0	0	3	1	1	1	0	0	0	2	0	3	0	0	3
Row2	0	0	1	0	0	0	0	0	0	1	0	0	0	0	3
Row3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3
Row4	1	0	9	0	4	1	0	0	0	4	0	1	0	1	3
Row5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Row6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
Row7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Row8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
Row9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Row10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Row11	0	0	1	0	0	0	0	0	0	1	0	0	0	0	3
Row12	1	1	1	0	1	0	0	0	0	1	0	0	0	0	3
Row13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Row14	0	0	1	0	1	0	0	0	0	1	1	5	0	1	3
Row15	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1
Row16	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3
Row17	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row18	0	0	1	0	0	0	0	0	0	0	0	0	0	2	3
Row19	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row20	0	0	0	1	1	0	0	0	0	0	0	0	0	2	3
Row21	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row22	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row23	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row24	1	0	0	0	0	0	0	0	0	1	0	0	0	2	3
Row25	0	0	0	0	0	0	0	0	0	0	1	0	0	2	3
Row26	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row27	1	1	0	0	0	0	0	0	0	1	1	0	0	2	3
Row28	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row29	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3
Row30	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
Row31	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0

FIG. M.3 Fixed missing values with constant values.

N

Filling missing values with formulas

Roberta Bortolotti, Senior Director, Nasdaq

Another technique used to fill missing data can be based on a strong relationship between the missingness and the other variables. The missing information is filled by taking into consideration the information from other columns. In this case, a formula is used to impute missing values based on specific logic rules derived from the understanding of the information from one or multiple variables.

As stated in Tutorial M, Tutorial N will walk through filling missing values by using a formula to derive the values for imputation. Depending on the complexity of the formula, it can be developed as either a mathematical formula or a Java snippet code. For the purpose of this tutorial, we will use a Java snippet code.

1. From Tutorial M, open the workflow named **Tutorial-M_Filling_MV_w_Constants**.
2. Remember that we replaced all of the missing values with constants in Tutorial M. In this tutorial, we will override some of those default replacements in four columns with other values.
3. Double-click on the **Missing Value** node to configure it further and click on the Column Settings tab.
 - a. In the list of columns on the left, double-click on the CHILD03 column to move it to the right pane.
 - b. Notice that the default setting in the righthand section is **Do nothing**. Leave it that way, because we will fill it in later with a formula in this tutorial.
 - c. Double-click also on **CHILD07**, **CHILD12**, and **CHILD18** and also leave the default setting as **Do nothing**.
 - d. See [Fig. N.1](#) to see the Column Settings overrides. Click **OK** and execute the node.
 - e. Notice that the Missing Values node has a yellow exclamation point under it. Hover the cursor over the exclamation point to read the warning that columns CHILD02, CHILD07, CHILD12, and CHILD18 still have missing values, which is OK; we will fill them later.
4. On the **Node Repository** section, enter “**statistics**” in the search box.
 - a. Click on the Missing Values node to highlight it.
 - b. Double-click the **Statistics** node to connect it automatically to the **Missing Value** node.
5. Double-click on the **Statistics** node to configure it.
6. In the *Configuration Dialog* screen, exclude all variables from the **Include** column by using the << symbol.
 - a. From the **Excludes** column, select the variables from CHILD03 to CHILD18 (a total of four variables) and click on the > symbol to move them to the **Include** column.
 - b. Set the “**Max no. of possible values per column (in output table):**” field to be 50,000.
 - c. The configured screen should look like [Fig. N.2](#).
 - d. Click **OK** and execute the node.
7. After execution, right-click on the **Statistics** node and select the **Occurrences Table**.
 - a. Look at each of the variables and note that every one has four categories (see [Fig. N.3](#)):
 - U – The default missing value filler
 - M – Male
 - F – Female
 - B – Both M and F (at least two kids)
 This provides us with information on the gender and the number of kids. We will use it to derive values.
 - b. Close the table.

Move the Statistics node down out of the way, because the next step will use a **Column Expressions** node to update the NUMCHLD column with the information we just gathered.
 - c. Close the table.
8. Click on the **Missing Values** node to highlight it.
9. Enter the string “Column exp” in the search box of the Node Repository,
10. Double-click on the **Column Expressions** node to connect it automatically.

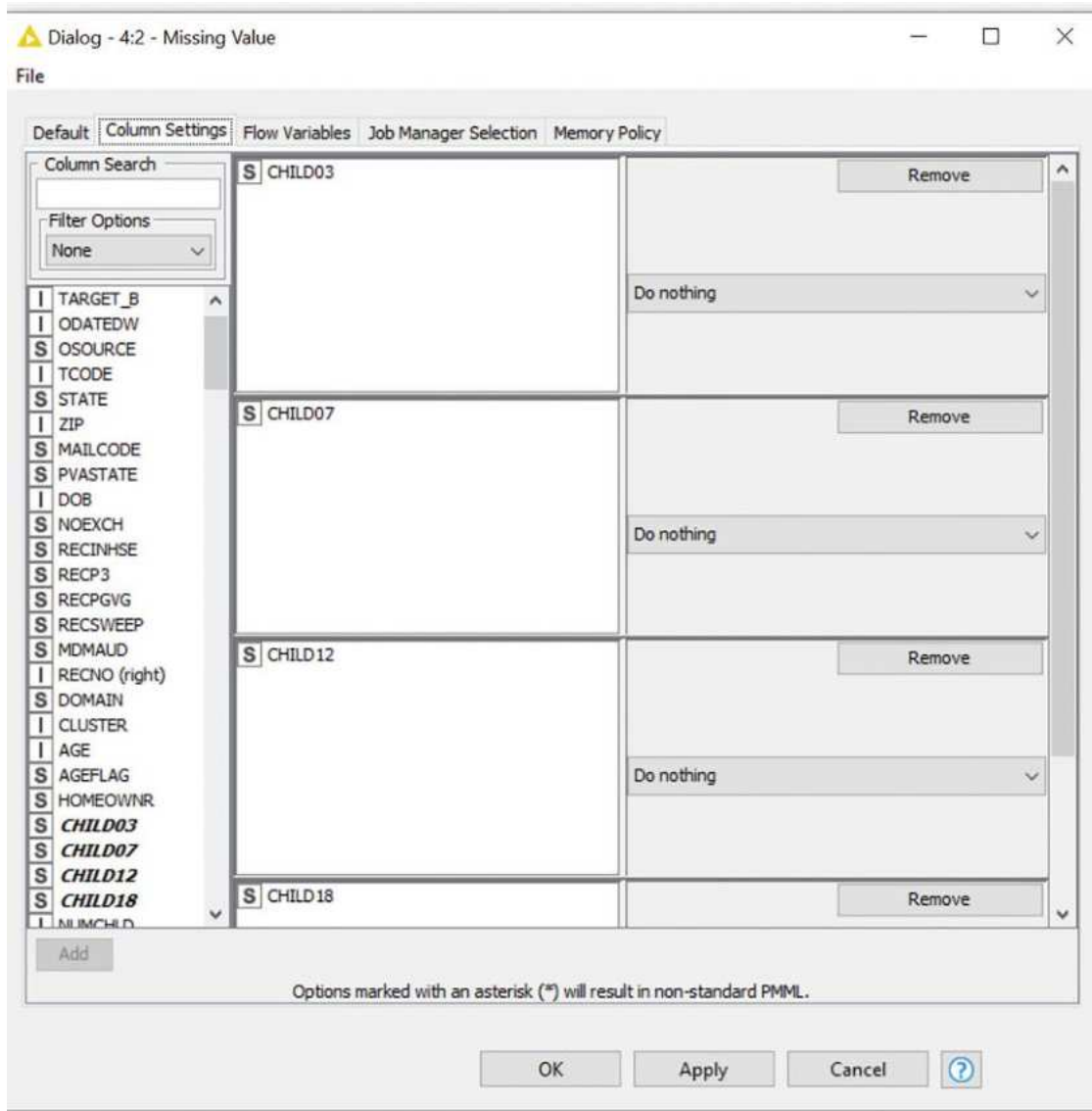


FIG. N.1 Configuration for the statistics measures of variables CHILD03 to CHILD18.

11. Double-click on the **Column Expressions** node to configure it.
12. Click on the plus sign under the Expressions tab to add an expression to the list.
13. In the Expression Editor, enter:

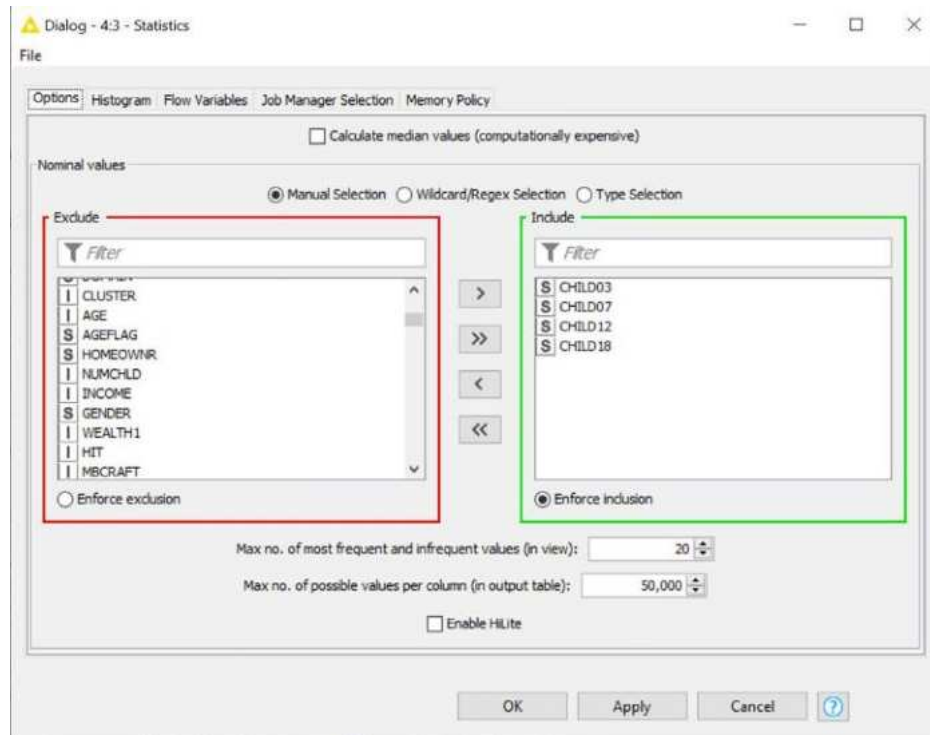

```
num = 0
if(column("CHILD03")=="F") num = num +1
if(column("CHILD07")=="F") num = num +1
if(column("CHILD12")=="F") num = num +1
if(column("CHILD18")=="F") num = num +1
if(column("CHILD03")=="M") num = num +1
if(column("CHILD07")=="M") num = num +1
if(column("CHILD12")=="M") num = num +1
if(column("CHILD18")=="M") num = num +1
if(column("CHILD03")=="B") num = num +2
```

```
if(column("CHILD07")=="B") num = num +2
if(column("CHILD12")=="B") num = num +2
if(column("CHILD18")=="B") num = num +2
if(column("NUMCHLD")==0) num
else column("NUMCHLD")
```

The easiest way to do this operation is to cut and paste this if-then-else statement into the Expression Editor of your workflow.

This statement will fill missing values (currently with zeros in them) with information from the CHILD03-CHILD18 columns.

14. Check row 10 in the Output table of the **Missing Values** node in Fig. N.4 to see that NUMCHLD = 0, even though CHILD18 = "F," which means that the

FIG. N.2. The configured **Statistics** node.

Occurrences Table - 3:14 - Statistics

File Edit Hilite Navigation View

Table "default" - Rows: 4 Spec - Columns: 12 Properties Flow Variables

Row ID	S	CHILD03	I	Count ...	D	Relati...	S	CHILD07	I	Count ...	D	Relati...	S	CHILD...	I	Count ...	D	Relati...	S	C
Row0	U			18828		0.988	U			18767		0.985	U			18694		0.981	U	
Row1	M			164		0.009	M			190		0.01	M			225		0.012	M	
Row2	F			52		0.003	F			73		0.004	F			99		0.005	F	
Row3	B			5		0	B			19		0.001	B			31		0.002	B	

FIG. N.3 Categories for variables CHILD03 to CHILD18.

- household has a female child, even though $\text{NUMCHLD} = 0$. Compare with the Output table of the **Column Expressions** node to see that NUMCHLD was updated to 1.
15. In the **Node Repository** section, enter "CSV" into the search box, as you did for Tutorial H.
 - a. Double-click on the **CSV Writer** node.
 - b. Connect the output triangle of the **Math Formula** node to the left triangle of the **CSV Writer** node.
 16. Right-click on the **CSV Writer** node and select **Configure**.
 17. Double-click the **CSV Writer** node on the workflow.
 - a. Select the **Settings** tab in the *Configuration Dialog* screen and specify the output file location and name for **Output Location**.
 - b. Name the output file as **Tutorial-N_Filling_MV_w_Formulas.csv** file. Make sure the file

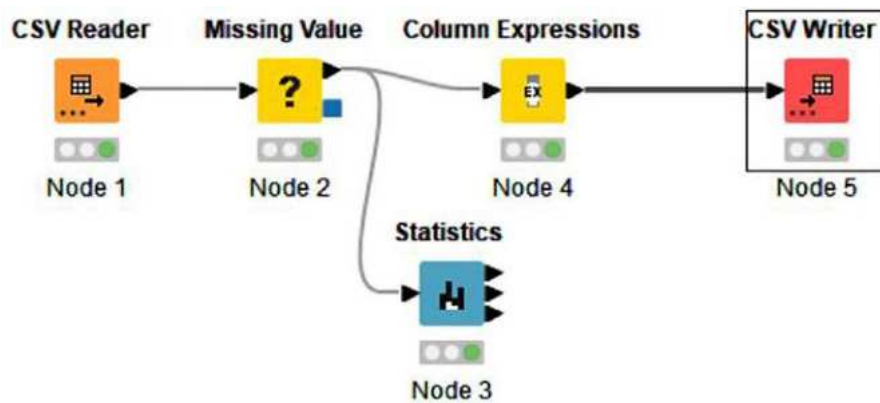


FIG. N.4 Workflow for filling missing values with constants and formulas.

- extension **.csv** is explicitly added to the file name; otherwise, KNIME will not recognize the file type.
- c. Check the **Write column header** checkbox.
 - d. Select the **overwrite** radio button in the **If exists:** section.
 - e. Click **OK** to return to the workflow screen.
 18. Execute the **CSV Writer** node.
 19. Click **File > Save** to save the workflow, which should be as shown in Fig. N.4.
 20. Close the KNIME application.

O

Filling missing values with a model

Roberta Bortolotti, Senior Director, Nasdaq

The imputation of missing values can also be done by building a simple model with a machine learning algorithm. The strategy is to have a model with as few variables as possible. It is more a model of a model to some extent, because it presents a logical tautology. The benefit of imputing missing values with a model outweighs the risk of error caused by the processing tautology.

In case the variables are all numbers, the easiest algorithm to use is linear regression (LR). The issue with using this algorithm, however, is that business data violate several important assumptions considered in the algorithmic rules of LR, which can cause significant errors in the parameter estimation and predictive accuracy. Alternatively, the use of a classification and regression tree algorithm may be more feasible, because it applies more successfully to either classification or estimation problems. The key feature of these algorithms is that variables can be nominal (text strings), ordinal, or continuous (numbers).

In this tutorial, a regression tree algorithm is used to build a model to replace missing values in the INCOME variable. It also includes steps to derive the mean absolute error (MAE) to evaluate the accuracy of the regression tree model.

1. Follow the steps in previous tutorials to create a new KNIME workflow and to read the CSV file to be worked throughout this tutorial.
 - a. Name the workflow as **Tutorial-O_Filling_MV_w_Model**.
 - b. Use the output file **Tutorial-N_Filling_MV_w_Formulas.csv** from Tutorial L to load the data into the **CSV Reader** node.
 - c. Be sure to uncheck the box in the Advanced Settings tab to limit the number of rows scanned.
 - d. We will use a new feature in KNIME version 5.0 to change the data type of the **TARGET_B** column from an integer to a string. Click on the Transformation tab of the configuration screen,

then click on the Type box and select String as the data type. The way to perform this operation in previous KNIME versions was to use the **Number to String** node.

- e. Execute the node.
2. Right-click on the **CSV Reader** node and select **File table**.
3. When the table displays, click on the **Spec – Columns** tab.
 - a. Look at the variable Income and note its missing values.
 - b. Close the table.
4. In the **Node Repository** section, enter “**partitioning**” in the search box.
 - a. Double-click the **Partitioning** node and connect it to the **CSV Reader** node.
Dataset partitioning is a classic approach to training modeling algorithms in an iterative training process. This node divides the dataset into two parts: modeling set and validation set. The training set is split internally in the regression tree algorithm into a training set and a testing set. The training set is used to train the model, and the testing set is used to calculate error between iterations of the training operation. The error level identified in the testing set is used to adjust the learning parameter in the algorithm for the next iteration. The validation set is used to evaluate the prediction accuracy of the final configuration of the regression tree algorithm.
5. Right-click on the **Partitioning** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, select **Relative[%]** and enter 60.
 - b. Select **Draw randomly**, as shown in Fig. O.1.
 - c. Click **OK**.
 - d. Execute the **Partitioning** node.

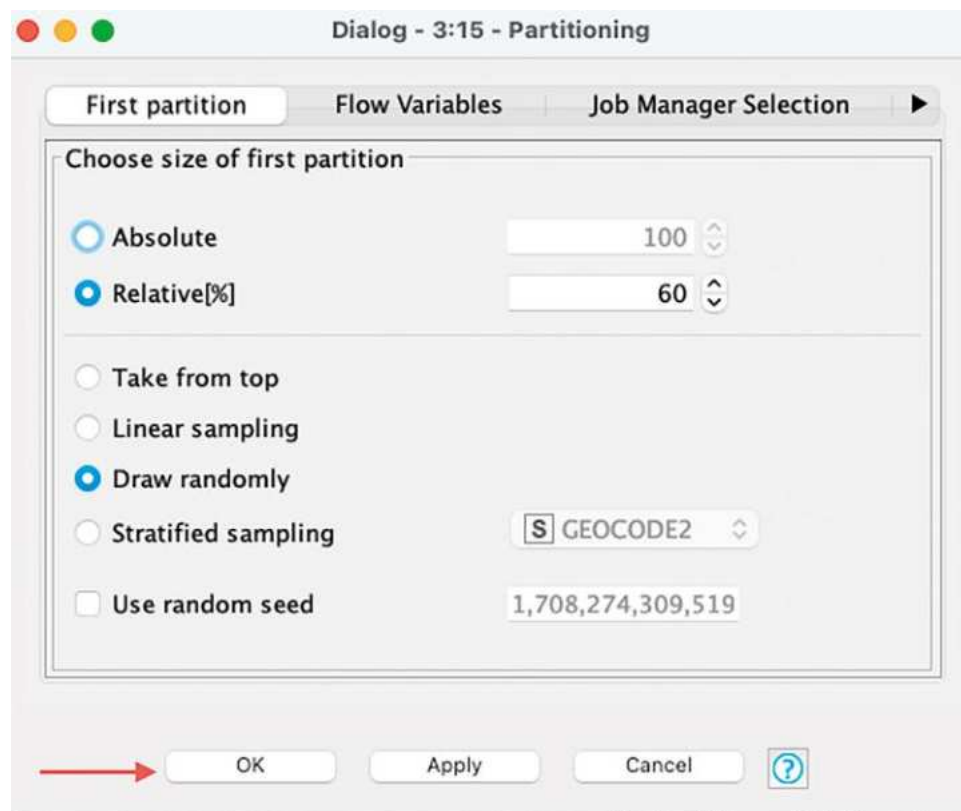


FIG. O.1 Configuration of the Partitioning node.

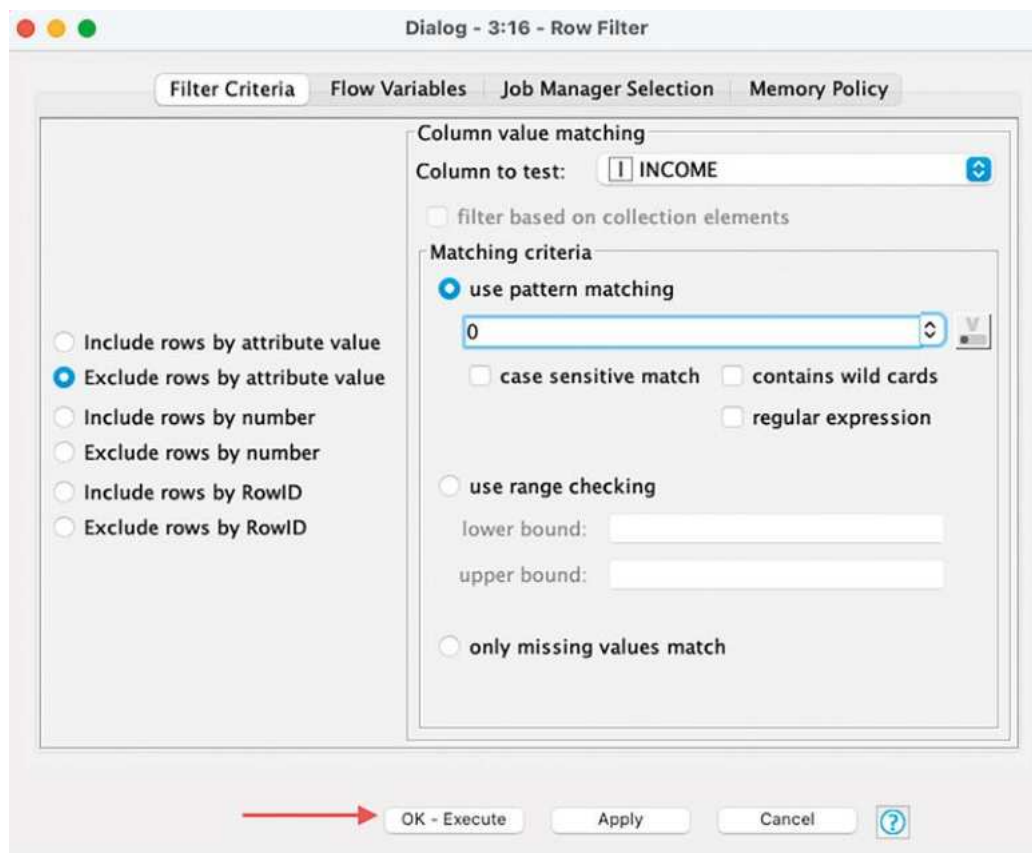


FIG. O.2 Configuration to filter out zero rows in variable INCOME.

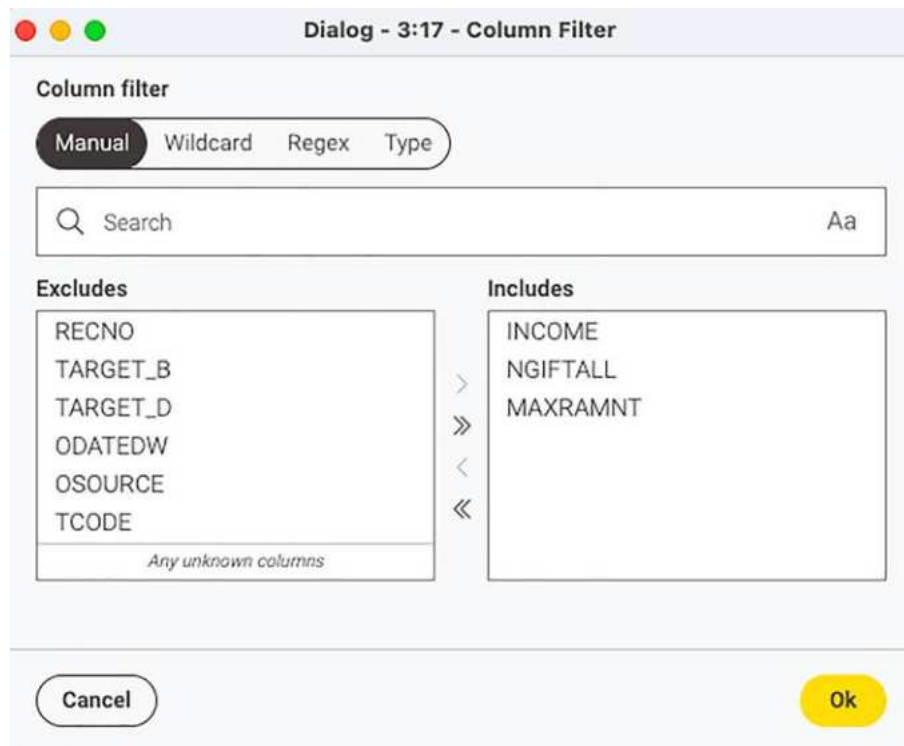


FIG. O.3 Configuration to use only variable INCOME, NGIFTALL, and MAXRAMNT.

6. In the **Node Repository** section, enter “**row filter**” in the search box.
 - a. Double-click the **Row Filter** node and connect it to the **Partitioning** node.
 Rows with value 0 for the INCOME variable were missing values filled in Tutorial M, and they need to be filtered out or the algorithm will assume that they are real numbers. This node excludes all rows with INCOME = 0. The rows that pass the **Row Filter** node have original positive values in them and will be used to train the model. Later in the tutorial, original values for INCOME = 0 are replaced with values predicted by the regression algorithm.
7. Right-click on the **Row Filter** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, select “**Exclude rows by attribute value**” on the left list.
 - b. In the **Column value matching** section, select INCOME from the drop-down list for “**Column to test**” (column 28 in the list).
 - c. Enter 0 for **use pattern matching** in the **Matching criteria** section. See Fig. O.2.
 - d. Click **OK - Execute**.
 - e. Execute the **Row Filter** node.
8. In the **Node Repository** section, enter “**column filter**” in the search box.
 - a. Double-click the **Column Filter** node and connect it to the **Row Filter** node.
 Except for those columns used in the model, all the other columns need to be filtered out. This is necessary, because the regression tree algorithm in KNIME does not do this in its configuration screen.
9. Right-click on the **Column Filter** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, exclude all variables from the **Include** column by using the << symbol.
 - b. From the **Excludes** column, select the INCOME, NGIFTALL, and MAXRAMNT columns to use as variables in the model, and click on the < symbol to add. See Fig. O.3.
 - c. Click **OK**.
 The strategy in building the model is to have the fewest number of variables possible and follow Occam’s razor concept. We are using the variable INCOME, because it is considered a predictable variable.
10. Execute the **Column Filter** node.

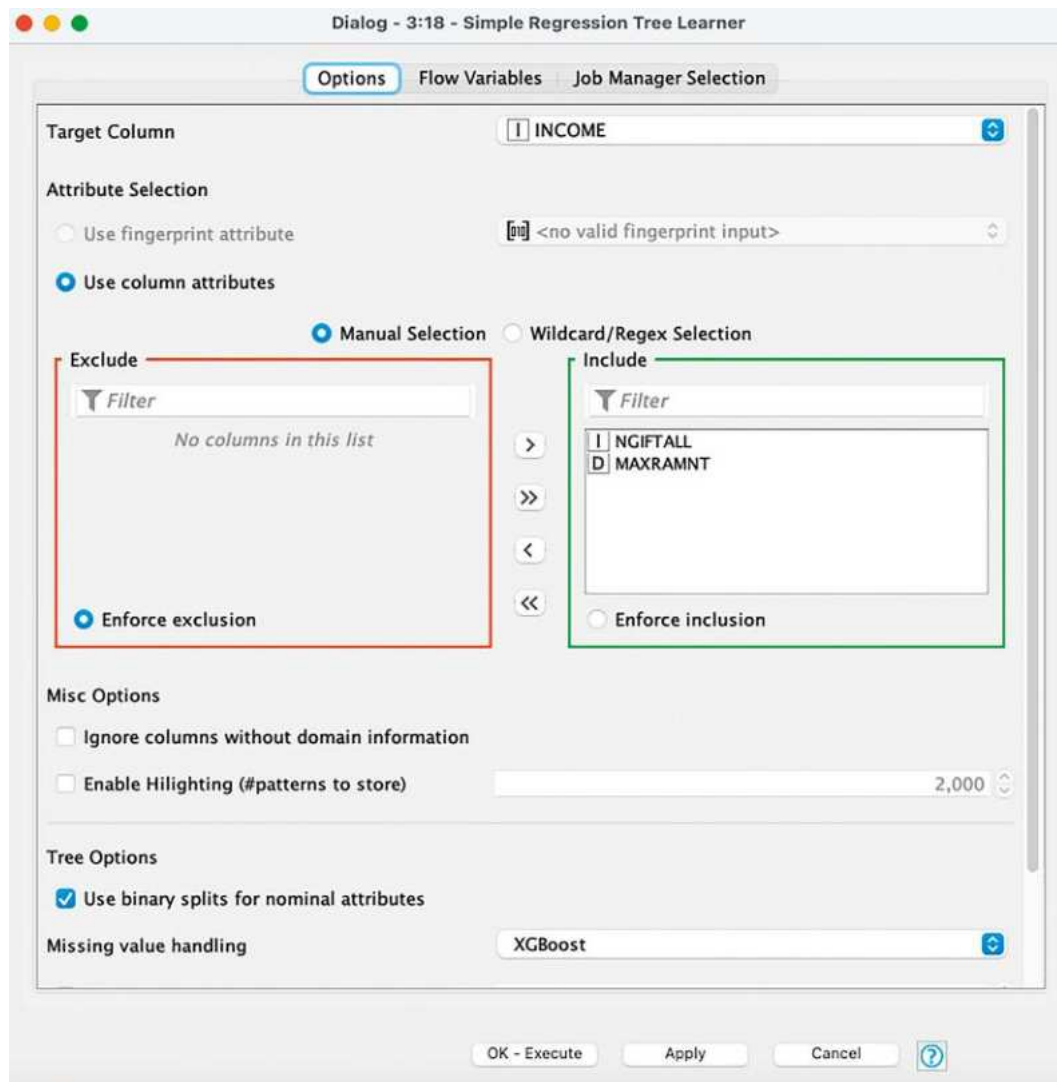


FIG. O.4 Configuration of the Simple Regression Tree Learner.

11. In the **Node Repository** section, enter “simple regression tree” in the search box.
 - a. Double-click the **Simple Regression Tree Learner** node and connect it to the **Column Filter** node.
12. Right-click on the **Simple Regression Tree Learner** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, select **INCOME** as the **Target Column**.
 - b. Make sure **MAXRAMNT** and **NGIFTALL** are in the **Include** list. See Fig. O.4.
 - c. Click **OK - Execute**.
13. In the **Node Repository** section, enter “simple regression tree” in the search box.
 - a. Double-click the **Simple Regression Tree Predictor** node.
 - b. Connect the output box of the **Simple Regression Tree Learner** node to the left box of the **Simple Regression Tree Predictor** node.
 - c. Connect the output triangle of the **CSV Reader** node to the left triangle of the **Simple Regression Tree Predictor** node (see Fig. O.11).
14. Right-click on the **Simple Regression Tree Predictor** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, check the checkbox **Change prediction column name** and enter **Prediction column name** as **INCOME2**. See Fig. O.5.
 - b. Click **OK - Execute**.
15. Execute the **Simple Regression Tree Predictor** node. Let’s make some calculations using the results of the simple regression trees.

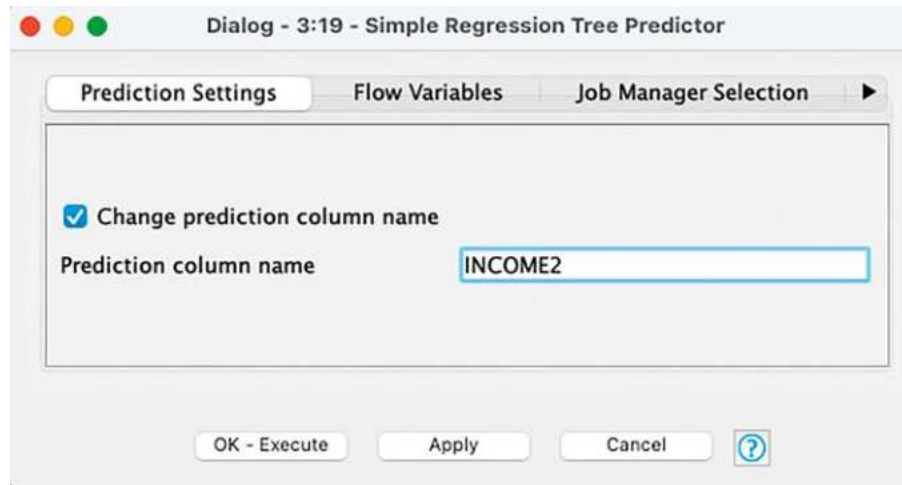


FIG. O.5 Configuration of the Simple Regression Tree Predictor.

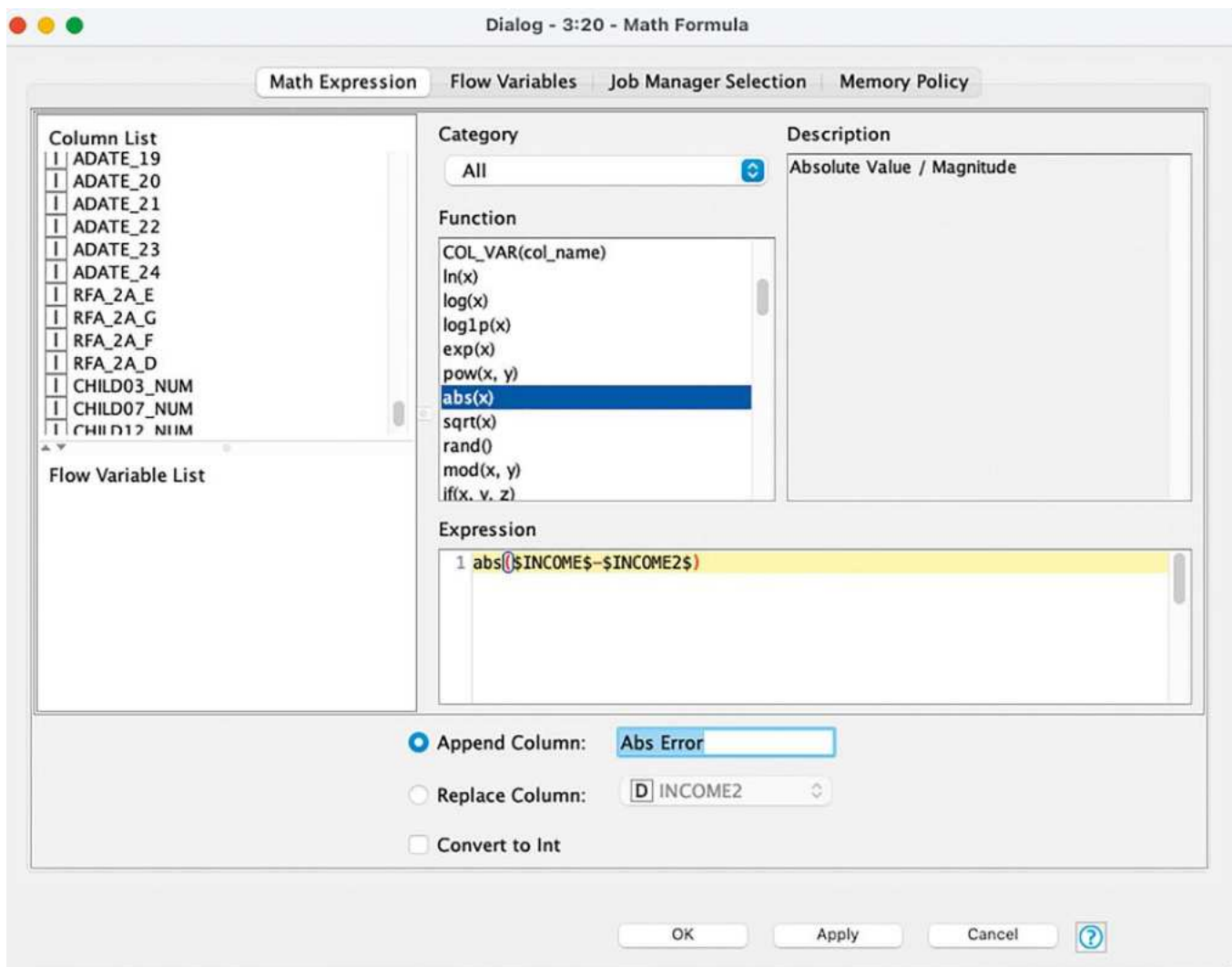


FIG. O.6 Configuration to calculate the absolute error.

Output data - 3:20 - Math Formula (Abs Error)

File Edit Hilite Navigation View

Table "default" - Rows: 19049 Spec - Columns: 230 Properties Flow Variables

Row ID	...	CHILD...	CHILD...	SUMC...	NUMC...	D INCOME2	D Abs Error
Row0	0	0	0	0	3.333	3.333	
Row1	0	1	1	1	4.341	1.659	
Row2	0	0	0	0	5	2	
Row3	0	0	0	0	3.5	2.5	
Row4	0	0	0	1	4	1	
Row5	0	0	0	0	4.037	4.037	
Row6	1	0	1	1	3.552	0.448	
Row7	0	0	0	0	3.583	1.583	
Row8	0	0	0	0	3.5	0.5	
Row9	0	0	0	0	3.889	3.889	
Row10	0	1	1	1	3.946	1.946	
Row11	0	0	0	0	3.955	2.955	
Row12	0	0	0	2	4.167	0.167	
Row13	0	0	0	0	3.333	3.333	
Row14	0	0	0	0	4.141	0.141	
Row15	0	0	0	0	3.955	2.955	
Row16	0	0	0	0	3.889	2.889	
Row17	0	0	0	1	7	0	

FIG. O.7 Calculated absolute error values.

- a. First, calculate the absolute error between the observed and predicted values of the variable INCOME.
- b. Second, calculate the MAE to evaluate the accuracy of the regression model.
- c. Last, select the original or predicted value for INCOME variable.
16. In the **Node Repository** section, enter “**math formula**” in the search box.
 - a. Double-click the **Math Formula** node and connect it to the **Simple Regression Tree Predictor** node.
17. Right-click on the **Math Formula** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, place your cursor in the **Expression Box**.
 - b. Then scroll down the list in the **Function List** window and double-click on the **abs(x)** function. This function calculates the absolute value of a calculation, in this case, the difference between INCOME and INCOME2.
 - c. Place the cursor in the x-expression position and double-click on the **INCOME** variable in the **Column List** window.
 - d. Enter the subtraction sign “-”, then double-click on the **INCOME2** variable in the **Column List** window. The INCOME2 variable should be the last one in the list. The final expression looks like this: **abs(\$INCOME\$ - \$INCOME2\$)**, as seen in Fig. O.6.
 - e. Select **Append Column** and enter **Abs Error**.
 - f. Click **OK**.
18. Execute the **Math Formula** node.
19. Double-click the node label and rename it to be **Abs Error**.
20. Right-click on the **Math Formula** node and select **Output Data**.
 - a. Scroll to the end of the table and note that the last column is **Abs Error**. See Fig. O.7.
 - b. Close the table.
21. In the **Node Repository** section, enter “**math formula**” in the search box again.
 - a. Double-click the **Math Formula** node and connect it to the first **Math Formula** node.
22. Right-click on the **Math Formula** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, place your cursor in the **Expression Box**.
 - b. Then scroll down the list in the **Function List** window and double-click on the **COL_SUM(col_name)** function.

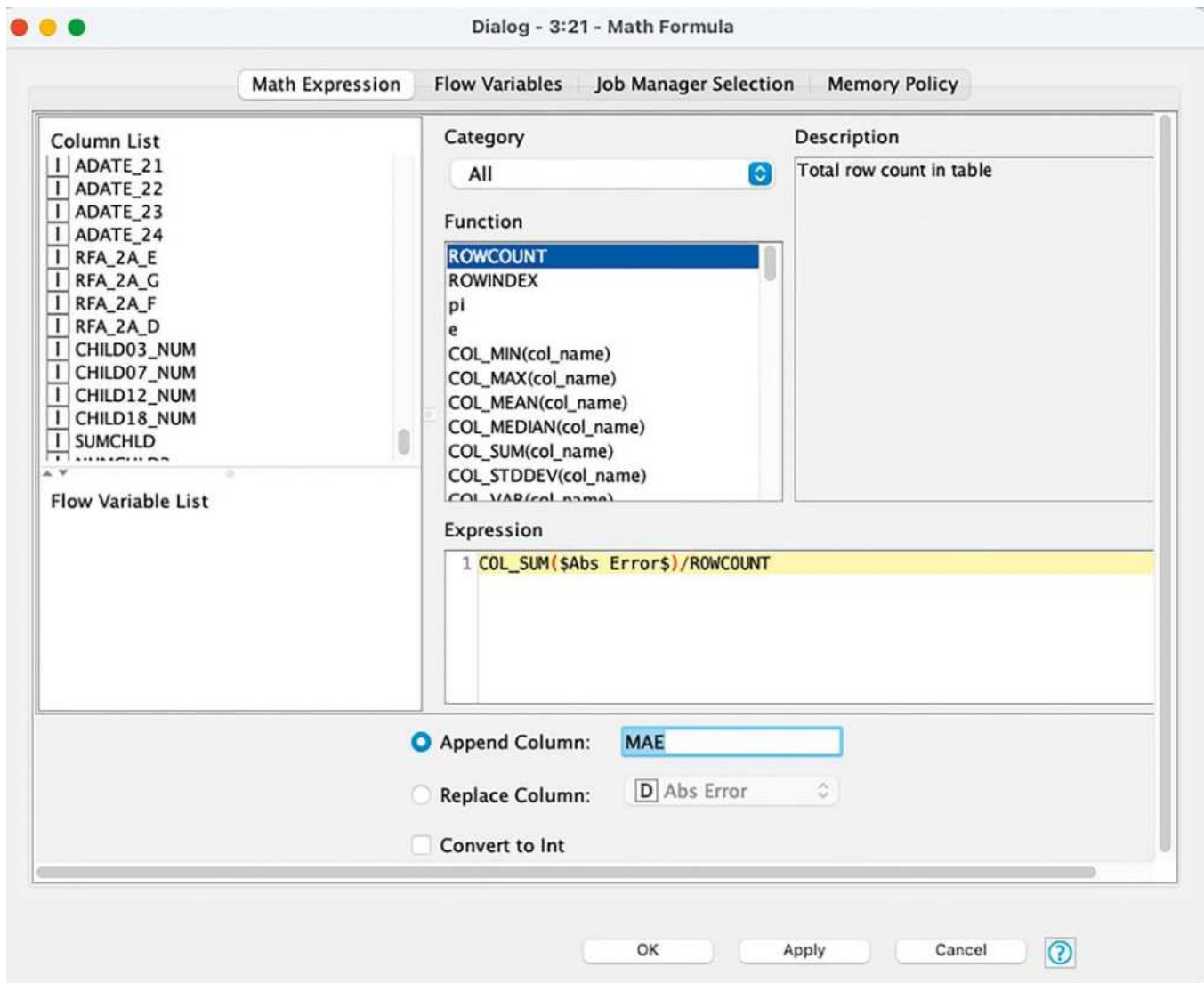


FIG. O.8 Configuration to calculate the MAE values.

- c. Place the cursor in the col_name-expression position and double-click on the new **Abs Error** variable in the **Column List** window.
- d. Enter the division sign "/" and then double-click on the **ROWCOUNT** function in the **Function List** window. The final expression looks like this: **COL_SUM(\$Abs Error\$)/\$ROWCOUNT\$**. This formula sums all values in the Abs Error column and divides the total by the number of rows.
- e. Select **Append Column** and enter **MAE**, as shown in Fig. O.8.
- f. Click **OK**.
23. Execute the **Math Formula** node.
24. Double-click the node label and rename it **MAE**.
25. In the **Node Repository** section, enter "math formula" in the search box. This should be your third **Math Formula** node.
 - a. Double-click the **Math Formula** node and connect it to the second **Math Formula** node.
26. Right-click on the **Math Formula** node and select **Configure**.
 - a. In the *Configuration Dialog* screen, place your cursor in the **Expression** Box.
 - b. Then scroll down the list in the **Function List** window and double-click the **if(x,y,z)** function. To assign original versus predicted value to the INCOME variable, an appended column INCOME_FIL is used. All variables that have values from 1 to 7 are assigned the predicted income

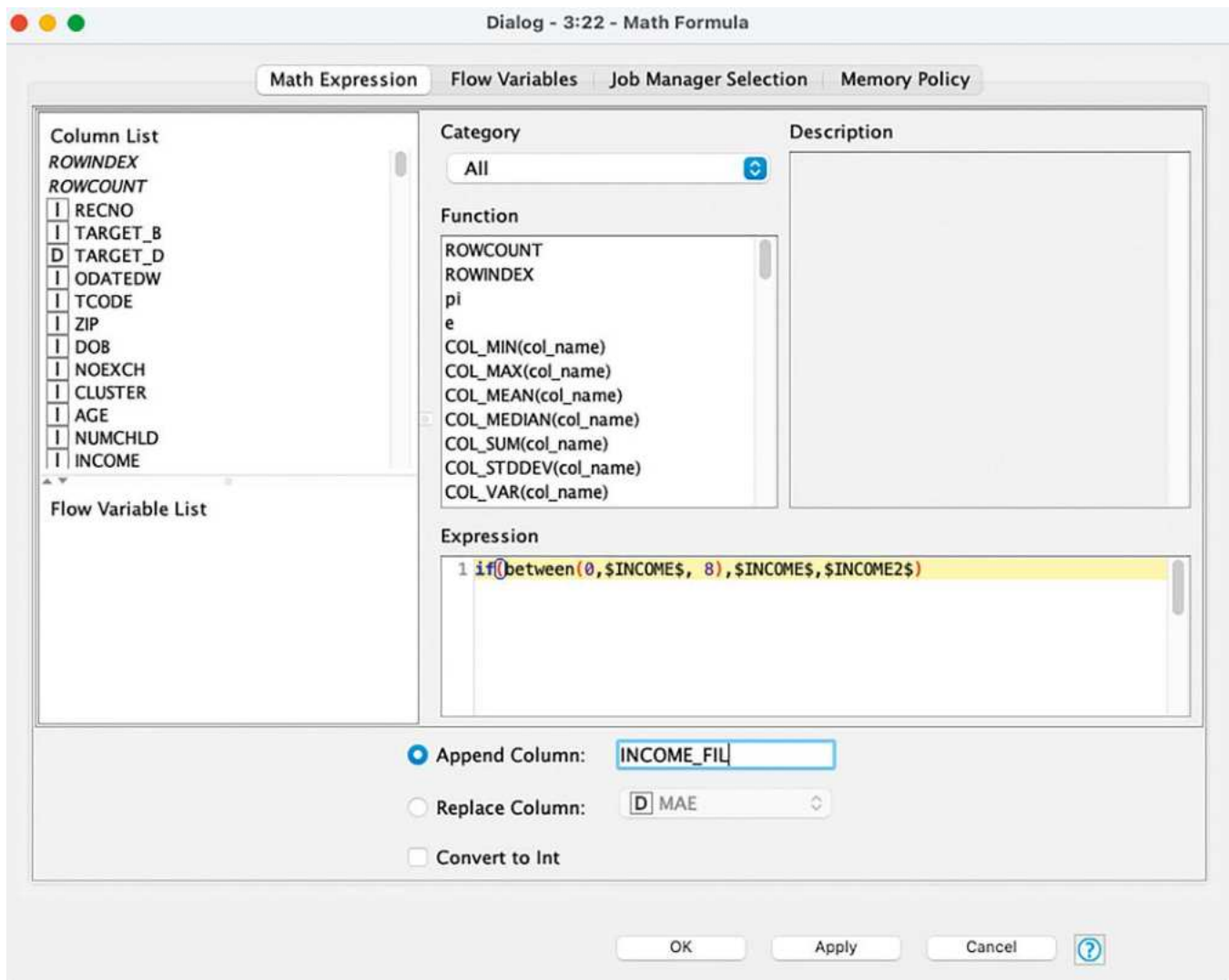
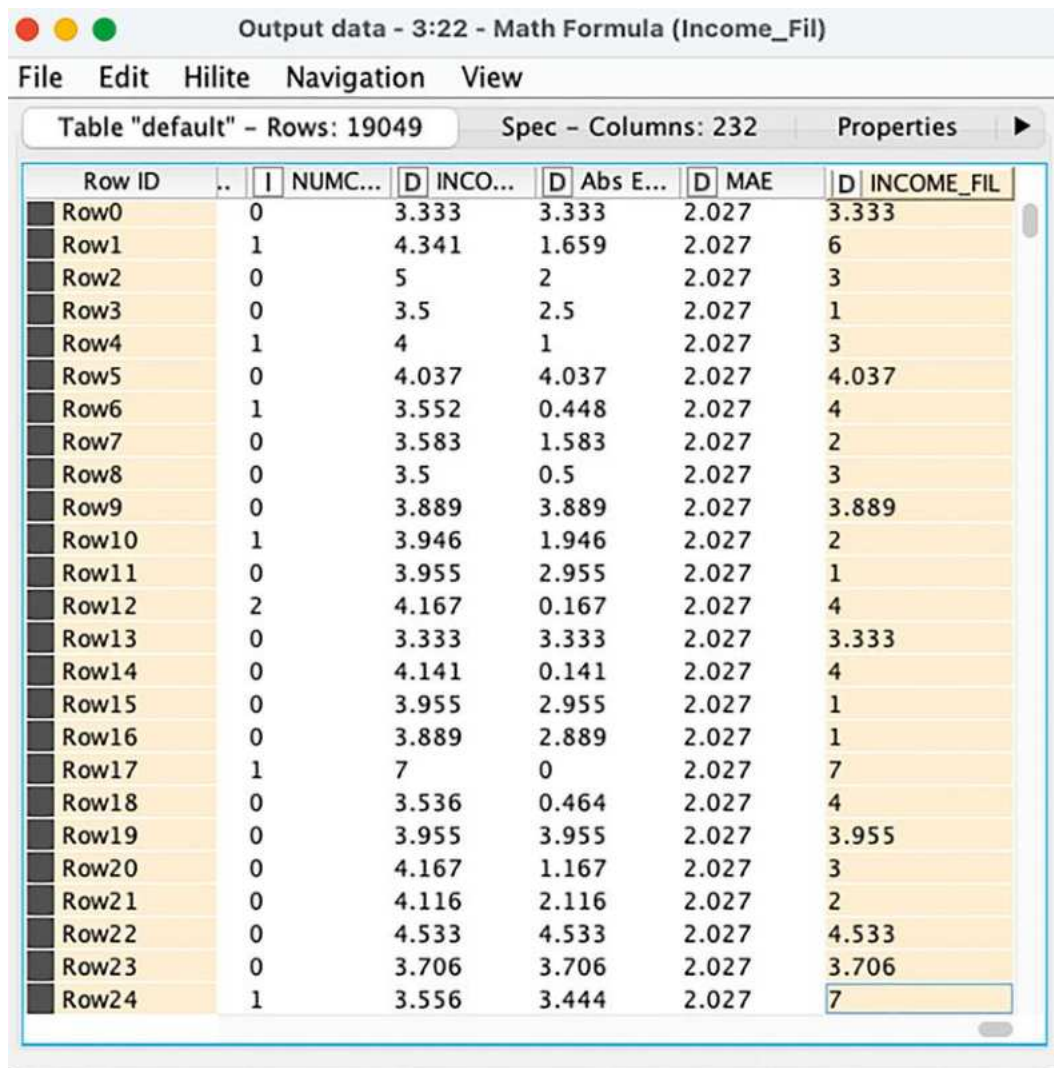


FIG. O.9 Configuration to impute calculated values into INCOME_FIL variable.

- variable (INCOME2); otherwise, the observed original value (INCOME) is the assigned value.
- c. Place the cursor in **Expression** box and enter the following if condition: `if(between(0,$INCOME$, 8),$INCOME$, $INCOME2$)`.
This syntax means if INCOME is between 0 and 8, use INCOME, else use INCOME2. Note that 0 is NOT part of the range; only values from 1 to 8 are included. However, 0 needs to be added in this condition for KNIME to consider the correct range; that is, to include value 1 as well.
- d. Select **Append Column** and enter **INCOME_FIL**, as shown in Fig. O.9.
- e. Click **OK**.
27. Execute the **Math Formula** node.
28. Double-click the node label and rename it to **Income_Fil**.
29. Right-click on the **Math Formula** node and select **Output Data**.
 - a. Scroll to the end of the table and note that the last column is **INCOME_FIL**.
 - b. Verify that all **INCOME = 0** records have been filled with the regression prediction. See Fig. O.10.
- c. Close the table.
30. In the **Node Repository** section, enter "CSV" into the search box, as you did for Tutorial H.
 - a. Double-click on the **CSV Writer** node.
 - b. Connect the output triangle of the **Math Formula** node to the left triangle of the **CSV Writer** node.
31. Right-click on the **CSV Writer** node and select **Configure**.



Row ID	NUMC...	INCO...	Abs E...	MAE	INCOME_FIL
Row0	0	3.333	3.333	2.027	3.333
Row1	1	4.341	1.659	2.027	6
Row2	0	5	2	2.027	3
Row3	0	3.5	2.5	2.027	1
Row4	1	4	1	2.027	3
Row5	0	4.037	4.037	2.027	4.037
Row6	1	3.552	0.448	2.027	4
Row7	0	3.583	1.583	2.027	2
Row8	0	3.5	0.5	2.027	3
Row9	0	3.889	3.889	2.027	3.889
Row10	1	3.946	1.946	2.027	2
Row11	0	3.955	2.955	2.027	1
Row12	2	4.167	0.167	2.027	4
Row13	0	3.333	3.333	2.027	3.333
Row14	0	4.141	0.141	2.027	4
Row15	0	3.955	2.955	2.027	1
Row16	0	3.889	2.889	2.027	1
Row17	1	7	0	2.027	7
Row18	0	3.536	0.464	2.027	4
Row19	0	3.955	3.955	2.027	3.955
Row20	0	4.167	1.167	2.027	3
Row21	0	4.116	2.116	2.027	2
Row22	0	4.533	4.533	2.027	4.533
Row23	0	3.706	3.706	2.027	3.706
Row24	1	3.556	3.444	2.027	7

FIG. O.10 Calculated values imputed to replace missing values.

32. Double-click the **CSV Writer** node on the workflow.
 - a. Select the **Settings** tab in the *Configuration Dialog* screen, and specify the output file location and name for **Output Location**.
 - b. Name the output file **Tutorial-O_Filling_MV_w_Model.csv**. Make sure the file extension **.csv** is explicitly added to the file name; otherwise, KNIME will not recognize the file type.
 - c. Check the **Write column header** checkbox.
 - d. Select **overwrite** radio button in the **If exists:** section.
 - e. Click **OK** to return to the workflow screen.
33. Execute the **CSV Writer** node.
34. Click **File > Save** to save the workflow, which should be like as shown in Fig. O.11.
35. Close the KNIME application.

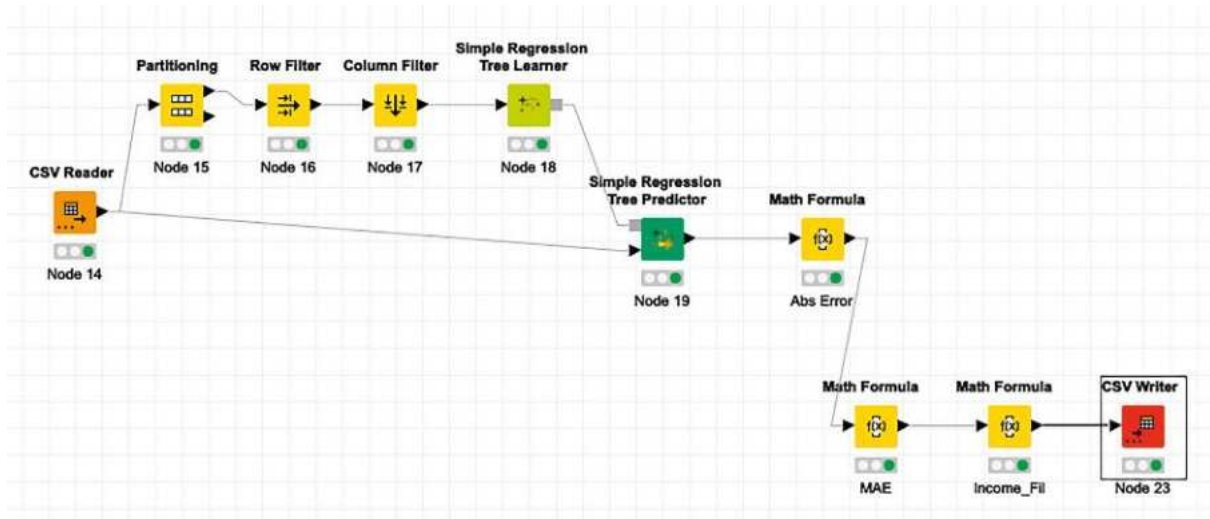


FIG. O.11 Workflow for filling missing values with a model.

APPENDIX A

Listing of tutorials and other resources on this book's companion web page

The companion web page displays a list of tutorials in the first edition and those tutorials in this third edition. The third edition tutorials are edited versions of those that appeared in the second edition.

Each tutorial can be accessed from this book's website: <https://www.elsevier.com/books-and-journals/book-companion/9780443158452>

Titles of the tutorials:

Second edition tutorials:

Note: Second edition tutorials A--O were modified and included in the third edition, and are not included in the second edition listing.

Tutorial A – Midwest Company Data – Example of Data Mining Recipes

Using Windows 10 and STATISTICA 13 - 305-318-9780 124166325

Tutorial B – Using the STATISTICA Data Mining Workspace Method for Analysis of Hurricane Data (Hurrdata.sta) -319-334-9780124166325

Tutorial C – Case Study – Using SPSS Modeler and STATISTICA to Predict Student Success at High-Stakes Nursing Examinations (NCLEX)-335-358-9780124166325

Tutorial D – Constructing a Histogram in KNIME Using MidWest Company Personality Data -359-376-9780124166325

Tutorial E – **Feature Selection in KNIME** -377-392-9780 124166325

Tutorial F – Medical/Business Tutorial

Tutorial G – A KNIME Exercise, Using Alzheimer's Training Data of Tutorial F

Tutorial H – Data Prep 1-1: Merging Data Sources

Tutorial I – Data Prep 1-2: Data Description

Tutorial J – Data Prep 2-1: Data Cleaning and Recoding

Tutorial K – Data Prep 2-2: Dummy Coding Category Variables

Tutorial L – Data Prep 2-3: Outlier Handling

Tutorial M – Data Prep 3-1: Filling Missing Values With Constants

Tutorial N – Data Prep 3-2: Filling Missing Values With Formulas

Tutorial O – Data Prep 3-3: Filling Missing Values With a Model

Tutorial P – City of Chicago Crime Map: A Case Study Predicting Certain Kinds of Crime Using STATISTICA Data Miner and Text Miner

Tutorial Q – Using Customer Churn Data to Develop and Select a Best Predictive Model for Client Defection Using STATISTICA Data Miner 13 64-bit for Windows 10

Tutorial R – Example With C&RT to Predict and Display Possible Structural Relationships

Tutorial S – Clinical Psychology: Making Decisions About Best Therapy for a Client

Tutorials AA – KK and O – Z, are all first edition tutorials:

Tutorial AA – Random Forests Classification

Tutorial BB –

Response Optimization for Data Mining Models

Tutorial CC –

Diagnostic Tooling and Data Mining: Semiconductor Industry and Data Mining Analysis

Tutorial DD –

Visual Data Mining: Titanic Survivors

(C:) > 000000000 - 3rd Ed DM HANDBOOK > 1 - 3rd Edition WEBSITE

☐ Name	Date modified
Tutorial-F-2nd-Edition	4/28/2023 3:59 PM
Tutorial-G-2nd-Edition	4/28/2023 3:59 PM
Tutorial-H-2nd-Edition	4/28/2023 3:59 PM
Tutorial-I-2nd-Edition	4/28/2023 3:59 PM
Tutorial-J-2nd-Edition	4/28/2023 3:59 PM
Tutorial-K-2nd-Edition	4/28/2023 3:59 PM
Tutorial-L-2nd-Edition	4/28/2023 3:59 PM
Tutorial-M-2nd-Edition	4/28/2023 3:59 PM
Tutorial-N-2nd-Edition	4/28/2023 3:59 PM
Tutorial-O-2nd-Edition	4/28/2023 3:59 PM
Tutorial-P-2nd-Edition	4/28/2023 3:59 PM
Tutorial-Q-2nd-Edition	4/28/2023 3:59 PM
Tutorial-R-2nd-Edition	4/28/2023 3:59 PM
Tutorial-S-2nd-Edition	4/28/2023 3:59 PM
Tutorial_AA	4/26/2023 10:30 AM
Tutorial_BB	4/26/2023 10:30 AM
Tutorial_CC	4/26/2023 10:31 AM
Tutorial_DD	4/26/2023 10:30 AM
Tutorial_EE	4/26/2023 10:31 AM
Tutorial_FF (1)	4/26/2023 10:37 AM
Tutorial_GG	4/26/2023 10:31 AM
Tutorial_HH	4/26/2023 10:31 AM
Tutorial_II	4/26/2023 10:31 AM
Tutorial_JJ	4/26/2023 10:31 AM
Tutorial_KK (3)	4/26/2023 10:31 AM
Tutorial_O (3)	4/25/2023 12:07 PM
Tutorial_P (2)	4/25/2023 12:16 PM
Tutorial_Q	4/26/2023 10:30 AM
Tutorial_R	4/26/2023 10:30 AM
Tutorial_S	4/26/2023 10:30 AM
Tutorial_T	4/26/2023 10:30 AM
Tutorial_U	4/26/2023 10:30 AM
Tutorial_V	4/26/2023 10:30 AM
Tutorial_W	4/26/2023 10:30 AM
Tutorial_X	4/26/2023 10:30 AM
Tutorial_Y	4/26/2023 10:30 AM
Tutorial_Z	4/26/2023 10:30 AM

**2nd
Edition
Tutorials**

**Tutorials AA -
KK,
and
O - Z
are all
1st Edition
Tutorials
..... (these
are all ZIPPED
folders)**

Tutorial EE –
Census Data Analysis: Basic Statistical Data Description

Tutorial FF – Linear and Logistic Regression (Ozone Data)

Linear and Logistic Regression (Ozone Data)

Tutorial GG –
R-Integration Into a Data Miner Workspace Node

R-node Competing Hazards Program Named cmprsk From the R-Library

Tutorial HH –
Social Networking Among Community Organizations: Tulsa Cornerstone Assistance Network Partners Survey of Satisfaction Derived From This Social Network by Members of Cornerstone Partners: Out of 24 Survey Questions, Which Are Important in Predicting Partner Satisfaction?

Tutorial II –
Nairobi, Kenya Baboon Project: Baboon Sociology Social Networking Among Baboon Populations in Kenya on the Laikipia Plateau

Tutorial JJ –
Jackknife and Bootstrap Data Miner Workspace and Macro for *STATISTICA* Data Miner

Tutorial KK –
Dahlia Mosaic Virus: A DNA Microarray Analysis on 10 Cultivars From a Single Source

Tutorial O –
Regression Trees Using Boston Housing Data Set

Tutorial P –
Cancer Gene

Tutorial Q – Clustering of Shoppers: Clustering Techniques for Data Mining Modeling

Tutorial R –
Credit Risk Using Discriminant Analysis in a Data Mining Model

Tutorial S –
Data Preparation and Management

Tutorial T –
Deployment of a Data Mining Model

Tutorial U – Stratified Random Sampling for Rare Medical Events – A Data Mining Method to Understand Pattern and Meaning of Infrequent Categories in Data

Tutorial V –
Heart Disease Utilizing Visual Data Mining Methods

Tutorial W –
Type II Diabetes Versus Assessing Hemoglobin A1c and LDL, Age, and Sex: Examination of the Data by Progressively Analyzing From Phase 1 (Traditional Statistics) Through Phase 4 (Advanced Bayesian and Statistical Learning Theory) Data Analysis Methods, Including Deployment of Model for Predicting Success in New Patients

Tutorial X –
Independent Component Analysis

Tutorial Y – NTSB Aircraft Accident Reports

Tutorial Z –
Obesity Control in Children Medical Tutorial Using *STATISTICA* Data Miner Recipe Childhood Obesity Intervention Attempt Domain: Medical Administration – Preventive

Third edition tutorials:

Tutorial A – How to Use Data Miner Recipe
STATISTICA Data Miner Only _2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

Tutorial B – Data Mining for Aviation Safety
Using Data Mining Recipe
“Automatized Data Mining” from
STATISTICA _2009_Handbook-of-Statistical-Analysis-and-

Tutorial C – Predicting Movie Box-Office Receipts
Using SPSS Clementine Data
Mining Software _2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Appli

Tutorial D – Detecting Unsatisfied Customers: A Case Study
Using SAS Enterprise Miner
Version 5.3 for the Analysis _2009_Handbook-of-Statistical-Analysis-

Tutorial E – Credit Scoring
Using *STATISTICA* Data
Miner _2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

Tutorial F – Churn Analysis
With SPSS-Clementine _2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

Tutorial G – Text Mining: Automobile Brand Review
Using *STATISTICA* Data
Miner and Text Miner _2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

Tutorial H – Predictive Process Control
QC-Data Mining
Using *STATISTICA* Data Miner
and QC-Miner _2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

Tutorial I – Business Administration in a Medical Industry:
Determining Possible Predictors for Days with Hospice Service for Patients with Dementia _2009

Tutorial-J-Clinical-Psychology-Making-Decisions-about-Best-Therapy-for-a-Client-Using-Data-Mining-to-Explore-the-Structure-of-a-Depression-Instrument_

Tutorial-K-Education-Leadership-Training-for-Business-and-Education-Using-C-and-R-to-Predict-and-Display-Possible-Structural-Relationships_2009_Handbook.

Tutorial-L-Dentistry-Facial-Pain-Study-Based-on-84-Predictor-Variables-Both-Categorical-and-Continuous-_2009_Handbook-of-Statistical-Analysis-and-Data

Tutorial-M-Profit-Analysis-of-the-German-Credit-Data-Using-SAS-EM-Version-5-3_2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

Tutorial-N-Predicting-Self-Reported-Health-Status-Using-Artificial-Neural-Networks_2009_Handbook-of-Statistical-Analysis-and-Data-Mining-Applications

APPENDIX B

Third edition:

The TUTORIALS and CASE STUDIES that are on this book's COMPANION WEB PAGE, use any one of the following analytics tool sets:

- STATISTICA Data Miner by Tibco
- IBM SPSS Modeler
- SAS Enterprise Miner
- KNIME Analytics Platform
- R
- R and PYTHON combined

Only the KNIME Analytics Platform is free; all other tool sets require licenses for use, which are not provided in this book. You can, however, look at the logical flow of processing operations in each tutorial, and translate them into operations performed by your analytics tool set. Most analytics tool sets support the basic processing and modeling operations contained in the tutorials.

One suggestion is to look for the availability of a free demo license from the vendor of the analytics tool set used in each tutorial.

Another suggestion is to download the free open-source KNIME Analytics Platform from <https://www.knime.com/>, and use KNIME capabilities to perform the operations required by a given tutorial.

Following is a listing of current Web page of the software sources listed above. Our suggestion is to check

these for what is “currently available” either as a free (or low-cost academic version), as what these vendors offer tends to change every-so-often, sometimes yearly. Please remember, that the specific web page offering these trial or evaluation downloads change from time to time so you, the reader, may have to search their web site a bit to find the page you need.

STATISTICA: <https://www.statistica.com/en/software/statistica-evaluation>

IBM SPSS MODELER: <https://www.ibm.com/products/spss-modeler?>

SAS ENTERPRISE MINER: https://www.sas.com/en_us/software/enterprise-miner.html

R PACKAGES for DATA MINING: <https://www.r-project.org/>

<https://www.rdatamining.com/>

<https://www.datacamp.com/tutorial/r-packages-guide>

https://cran.r-project.org/web/packages/g6_packages_by_name.html

<https://posit.co/products/open-source/rstudio/>

R and PYTHON combined: <https://smartboost.com/blog/data-mining-with-r-or-python/>

KNIME: <https://www.knime.com/>

<https://www.knime.com/getting-started-download>

Index

Note: Page numbers followed by *b* indicate boxes, *f* indicate figures and *t* indicate tables.

- A**
Academic analytics, 241–243
Access tools, data, 45
Account-centric database *versus* customer-centric database, 26–28
Account-centric format
 computer databases, 57–58
 data access, 58
Accuracy, 287, 400
 versus generality, 155–156
 global, 170, 258
Active learning, 212, 214
Adaptive boosting (AdaBoost) algorithm, 178–179, 179*f*
Affordable Care Act (ACA), 223
Aggregation, 105–107, 107*t*
 plan, 63–64
Agile modeling, 278–280
 timeliness and sufficiency in, 279–280
AI. *See* Artificial intelligence (AI)
AI Act, 209
Akaike's Information Criterion, 180
Algorithms. *See also* Classification algorithm
 AdaBoost, 178–179, 179*f*
 adding a cost matrix, 180
 clustering, 140, 145–146
 data analytics, 137
 decision tree, 140, 144, 153
 inadequate experimental design improper, 175
 logistic regression, 145
 machine-learning, 95, 109, 117, 156, 179–180, 242, 244, 427
 parameter adjustment, 179–180
 parametric modeling, 107–108
 prediction, 146
 regression tree, 449–457
 selection, 254–255
Alpha error, 10–11
American Statistical Association (ASA), 288–290
Analytical data flow, 60
Analytical data mart, 58
Analytical file, 103
Analytical modeling, 149
 errors in, 169–173
 evaluation
 classification errors, 170–171
 on predictive power, 169–171
 on random error, 171–173
 systematic errors, 174–175
 types, 150–152
Analytical project planning, 251
Analytics architectures, 21st century, 228
Analytics tool sets, 463
Annotation software, 215
ANNs. *See* Artificial neural networks (ANNs)
Area under the ROC curve (AUROC), 20, 171, 174, 246, 258
Aristotle, 9, 9*f*, 150, 150*f*
Arithmetic correctness, 89
ARMA algorithm. *See* Auto-regressive moving-average (ARMA) algorithm
Artificial intelligence (AI), 227–228
 domain experts, 218
 reinforcement learning techniques, 265
Artificial neural networks (ANNs), 10, 12*f*, 15–16, 153, 199–200
 multilayer perceptron, 16*f*
 neural network in, 15*f*
Artistic steps, in data analytics, 44
Art of data analytics, 43–44
ASA. *See* American Statistical Association (ASA)
Attention, 18–19
Attribute, 103
Audience
 business stakeholders, 75–76
 goal, 75
 internal customers, 75
 subject matter experts, 75
AUROC. *See* Area under the ROC curve (AUROC)
AutoDiscovery from Butler Scientifics, 13
Automated
 business modeler, 13
 decisioning, 227
 IBM SPSS Modeler selection procedure, 322–323, 326*f*
 modeling interface, DMRecipe, 330
 predictive analytics applications, 13–14
 statistician project, 13
Autonomous vehicles, 196, 214
Auto-regressive moving-average (ARMA) algorithm, 114
B
Back-propagation operation, 12–13, 16, 153, 156, 195
Backward feature elimination node, 122–123, 123*f*
Bagging technique, 177–179, 183, 185, 185*f*, 244–245, 245*f*, 256, 322–330, 325*f*
Balancing method, 96, 156, 244, 254, 321
Band-pass filter, 90
Bard, 19–20, 265–268
Base models. *See* Foundation models
Base rate, cancer, 5*b*
Bat signal modeling, 32, 32*f*
Bayes, Thomas, 5, 5*f*
 approach, 5, 42–43
 networks, 228
 theorem, 6
Behavior pattern, static measures *versus* historical measures, 233
Beta distribution, 159
BI. *See* Business intelligence (BI)
Bias, decision-making, 284
Bidirectional Encoder Representations from Transformers (BERT), 18–19, 266
Big data, 66, 241
Binary variables, 71, 71*t*
Binomial distribution, 159
Bioinformatics, 223–224
Bionomics: Economy As Ecosystem (Rothschild), 276
Bivariate analysis, 38, 78–79
Black box modeling, 63, 189–190, 190*f*, 204
BlenderBot-3 Meta AI, 266
Boosted Trees algorithm, 299, 388*t*, 389, 391*f*
Boosting technique, 178–179, 183, 185, 185*f*, 322–330, 324*f*
Bootstrap
 aggregation, 179
 method, 175–176
Box-and-whisker plot, 78–79, 82*f*
Business
 analytical problem, 138, 139*f*
 customer relationship management issues in, 231
 data analytics capacity, 3–4
 ecosystem, 277, 285
 customer relationship management in, 232–235
 transforming corporations into, 232
 environment, 37
 interdiction strategies, 190
 organism
 complex system, 277–278
 decision-making activities in, 276
 muscles in, 276
 stakeholders, 75–76
 understanding, 37–38
 analytic project goals, 38
Business intelligence (BI), 58–59

- C**
- CAPTCHA app, 211
 - CAR. *See* Customer analytic record (CAR)
 - CART. *See* Classification and Regression Trees (CART)
 - Categorical variables, 103
 - dummy coding, 427–433, 428f, 429f, 430f, 431f, 432f, 433f
 - Causal generalization, 98
 - Causal inference, 228
 - CE Credit Score, 260
 - Central Limit Theorem (CLT), 145, 159–163
 - CHAID. Chi-square automatic interaction detection (CHAID), 317–318, 328
 - Chatbot. *See* Bard
 - ChatGPT, 265–268, 284
 - Chinchilla, 266
 - Chi-square automatic interaction detection (CHAID), 317–318, 328
 - Classical statistical analysis, 5–6
 - Classification, 154–155
 - accuracy *versus* generality, 155–156
 - algorithm, 251
 - assumptions, 155
 - data set, 155
 - errors, 170–171
 - initial operations in, 155
 - issues with, 155
 - Classification and Regression Trees (CART), 137–138, 137t, 140–145, 299
 - advantages, 144
 - decision tree, 140–142, 141f, 142f
 - pruning trees, 143
 - recursive partitioning, 142–143
 - statisticians, 143–144
 - three-fold cross-validation operation, 143, 144f
 - uses, 144–145
 - Claude from Anthropic, 266
 - Cleaning of data, 419–426, 420f, 421f, 422f, 423f, 424f, 425f, 426f
 - Clinical decision-making, 225
 - Closing the information loop, 43
 - Cloud computing, 21, 67, 269
 - CLT. *See* Central Limit Theorem (CLT)
 - Clustering, 154
 - data clusters, 146f
 - k-means clustering, 146
 - V-fold cross-validation, 146
 - CNN. *See* Convolutional neural network (CNN)
 - Code-based integration, 51–52
 - Code checking, 88
 - Coefficient of determination, 152, 174
 - Cognitive dissonance, 283
 - Coincidence (confusion) matrix, 170, 170t
 - Collinearity problem, 8, 166
 - Colon cancer
 - incidence, 249
 - screening tests, 249
 - Column filtering operations, 104–105
 - Column Selection tab, 405
 - Comma-separated-value (CSV), 62, 86
 - Company AI, 270–271
 - Compensation, 216
 - Complex models, 178, 277–278
 - Conditional probability, 5
 - Confusion matrix, 187
 - Consistency problems, 89
 - Constant
 - filling missing values with, 441, 442f, 443f
 - value filtering, 122, 253
 - variance, 8
 - Construct validity, 98
 - Continuous variables, 8
 - Convergence, 42
 - Conversational AI. *See* Bard
 - Convolution, 16–19
 - Convolutional neural network (CNN), 16–17, 19
 - Copyright infringement, 284
 - Corporate information factory, 277–278, 277f
 - Correlation filtering, 122–123, 122f, 253
 - Cost matrix, 180
 - Counterfactuals, 208
 - COVID-19 descriptive analytics, 227
 - Credit risk models, 260
 - Credit risk score, loan application
 - feature importance methods, 262
 - model results, 261–262
 - ruleset, decision tree algorithm, 262–263
 - Creditworthiness, 259, 263
 - Credit Xpert Credit Score, 260
 - CRM. *See* Customer relationship management (CRM)
 - Cross-industry standard process for data mining (CRISP-DM), 35–37, 125, 126f
 - acceptance criteria, 126
 - access data, 127, 127f
 - analytical goals of project, 126
 - business
 - goals, 125–126
 - stakeholders, 126
 - understanding phase, 37–38, 125–127
 - characterization and description of data, 128
 - data
 - cleaning, 129
 - preparation phase, 40–41, 129–131
 - reduction, 129–130
 - understanding phase, 38–40, 127–129
 - understanding tasks, 69–74
 - derived variables, 130–131
 - enhancing and enriching data, 128
 - filtering, 130
 - handling of outliers, 131
 - missing value imputation, 130
 - modeling
 - activities, 41–43, 41f
 - algorithms, 41
 - architecture, 41
 - assumptions, 41
 - phase, 36f, 37, 41–43, 131–132
 - predictive analytics projects, 132–134
 - process, 169
 - recoding, 130
 - sampling regimes, 129
 - service level agreement, 126–127
 - standardization, 130
 - target variable, 127
 - timeline, 127
 - working relationships, 126
 - Cross-sell modeling, 231, 235
 - Cross-validation
 - deployment, 177
 - K-fold, 176, 176f
 - Python script, 176–177, 177t
 - Crowdsourcing, 215
 - CSV. *See* Comma-separated-value (CSV)
 - Cumulative gains curve, 171, 172f
 - Current probability, cancer, 5b
 - Curse of Dimensionality, 119, 353–354
 - Curvilinear Regression, 164
 - Customer
 - centric database *versus* account-centric database, 26–28
 - response modeling, 233
 - retention modeling, 231
 - Customer analytic record (CAR), 28, 60, 127
 - creation, 28
 - decision-making activities in, 276
 - Customer-centric analytical data mart
 - analytical operations, 58f, 59
 - branch reporting, 58f
 - IT personnel, 58
 - shift organization, 58–59
 - Customer relationship management (CRM), 27, 30, 113, 152, 169–170, 284
 - in business ecosystems, 232–235
 - issues in business, 231
 - model, 125, 131f
 - Cybersecurity, 284–285
- D**
- DA. *See* Data analytics (DA)
 - DAAS. *See* Data-as-a-service (DAAS)
 - Data, 393
 - abstractions, 112–114, 233–234
 - access tools, 45
 - acquisition, 39, 401–410, 402f, 403f, 404f, 405f, 406f, 407f, 408f, 409f, 410f, 411f, 412f
 - aggregation, 65
 - analysis, exploratory, 28
 - annotation, 212
 - ethics of, 216–217
 - ecosystem, 215–216
 - issues, 217
 - cleaning, 65, 129
 - classification, 85
 - and recoding, 419–426, 420f, 421f, 422f, 423f, 424f, 425f, 426f
 - types of, 85
 - cluster in clustering problem, 145, 146f
 - code uniqueness checking, 88
 - conditioning, 56, 244, 254
 - consistent data sets, 95
 - data set balancing, 96–97
 - missing values, 97–101
 - standardization and normalization, 95–96
 - consistency, 218
 - DataRobot, 13
 - description, 39–40, 65, 251, 413–416, 414f, 415f, 416f, 417f
 - distribution, 7–8, 108, 109f
 - engineering, 103–104, 250f, 250t
 - exclusions, 252
 - exploration tools, 46
 - extraction, 60–64, 61f
 - filtering, 65, 90
 - removal of outliers, 435–439, 435f, 436f, 437f, 438f, 439f, 440f
 - time-series filtering, 90
 - fit, 218

Data (*Continued*)

- format checking, 88
- imputation, 441
 - filling missing values with constants, 441, 442f, 443f
 - filling missing values with formulas, 445–448, 446f, 447f, 448f
 - filling missing values with model, 449–457, 450f, 451f, 452f, 453f, 454f, 455f, 456f, 457f, 458f
- integration, 39, 40f, 64–65, 251, 401–410, 402f, 403f, 404f, 405f, 406f, 407f, 408f, 409f, 410f, 411f, 412f
 - ODBC connectors, 45–46
- inventory, 61–62
- loading, 59, 65–66
- lookups, 104
- mart, 276–277
 - analysis outside/inside database, 66
 - dependent *versus* stand-alone, 66
 - intermediate table design, 66
 - logical, 66
 - physical, 27, 66
 - virtual, 27, 59
- medical and healthcare predictive analytics, 224
- medical informatics, process models in, 224
- modeling, 57
- motives and attitudes, 233–234
- multiomics, 224–225
- paradigm shift, 28
- preparation, 40–41, 144, 197, 419
 - CRISP-DM phase, 129–131
 - issues, 40–41
 - operations, 243–244
 - in predictive analytics, 132–134
- profiling, 65
- quality assessment, 40
- reduction, 129–130
- reformatting, 251
- science, 3–4, 25, 225
- segmentation, 132
- source
 - analysis, 250–251
 - merging, 401–410, 402f, 403f, 404f, 405f, 406f, 407f, 408f, 409f, 410f, 411f, 412f
- storage systems, 39f
- structure design, 66–67
- transformation, 59, 61f, 65, 133, 427
- type changes, 87, 104
- type checking, 86–87
 - category to number, 86
 - changing strings to numbers, 87
 - number to string type change, 86
 - string to number type change, 86
 - string variables, 87
- understanding
 - CRISP-DM, 69–74, 127–129
 - KNIME, 74, 74f
- understanding activities, 38–40
 - data acquisition, 39, 39f
 - data description, 39–40
 - data integration, 39, 40f
 - data quality assessment, 40
- visualization, 46, 227
 - Ames Housing data set, 76–84, 76f
 - audience, 75–76

Data (*Continued*)

- foundations, 75
- warehouses, 58–59, 277
- Data analytics (DA), 3–4, 241, 282–283
 - artistic steps in, 44
 - business environment, 37
 - business objectives, 37
 - challenges, 30
 - goals and objectives, 38
 - historical development, 29t
 - issues in, 30–31
 - and knowledge discovery, 25f
 - process, 24–25, 35–37
 - project
 - example, 31–32
 - requirements for success, 31
 - strengths, 26
 - theoretical framework, 25–26
 - tools for pattern discovery, 32–33
 - types, 137
- Data-as-a-service (DAAS), 21
- Databases, 62
- Data-centric AI (DCAI), 217–218
- Data Miner Recipe interface, 293, 294f, 295f
- Data mining, 45, 154–155, 177
 - attributes, 287
 - definition, 24
 - IBM SPSS Modeler, 320–322
 - solutions, 138
 - STATISTICA, 303–314, 303f, 304f, 305f, 306f, 307f, 308f, 309f, 310f, 311f, 312f, 313f, 314f, 315f, 316f
 - supervised classification problems, 154–155
 - temporal, 114–115
- Data mining recipes (DMR), 298–299
 - module, Open Midwest Data with Statistica, 293–298, 294f, 295f, 296f, 297f, 298f, 299f, 300f, 301f, 302f
 - STATISTICA, 330–333, 330f, 331f, 332f, 333f, 334f, 335f, 336f
- Data science and machine learning (DSML)
 - analytics platforms, 50–51
 - KNIME, 51, 51f
 - menu-based and multipersona analytics platforms, 51–52
- Date variables, 72
- DCAI. *See* Data-centric AI (DCAI)
- Deciles, 171
- Decision-making activities, in business organism, 276
- Decision trees, 12, 12f, 96, 140–142, 141f, 142f, 184, 203–204
 - algorithms, 95, 153, 179–180, 262–263
 - processing elements, 120–121
- Deductive method, 23, 35
- Deep learning (DL), 21, 193–194, 194t, 201, 212–213, 283
 - architectures, 16–17
 - definition, 14–15
 - human cognition, 15
 - technology, fourth generation, 14–21 *versus* traditional machine learning, 193–200, 194t
- Deep learning neural networks (DLNNs), 15–17, 19f
- Definitional data abstraction, 113, 234
- Demographic and lifestyle variables, inclusion of, 251

- Dependent *versus* stand-alone data mart, 66
- Deployment, 43, 169
 - cross-validated model, 177
- Derived variables, 44
- Descriptive data analysis, 76
- Descriptive modeling, 28, 151
- Descriptive statistics, 413
- Diagnostic models, 151–152
- Digital age, healthcare and wellness, 21st-century, 228–229
- Digital nervous system, 276
- Dimensionality reduction
 - classification, 119–120
 - feature selection *versus*, 119
- Directional hypothesis, 175
- Discovering patterns/rules, data mining activity, 28–29
- Discretization, 115
- Dispersion parameter, 165
- Disseminated intravascular coagulation, 227
- Distributed computing, 67, 67f
- Diuretics, 225
- DL. *See* Deep learning (DL)
- DLNNs. *See* Deep learning neural networks (DLNNs)
- DMR. *See* Data mining recipes (DMR)
- DMWay, 13
- Domain knowledge, 32–33
- Donor development, 241
- DSML. *See* Data science and machine learning (DSML)
- Duality, 8–10
- Dummy coding, 71
- Dummy variables, 103, 115, 156, 397–398, 427

E

- Edge AI, 268
- Educational data mining (EDM), 241
- Educational psychology, 241
- EDW. *See* Enterprise data warehouse (EDW)
- Efficiency paradigm, 275–276
- Efficient business, 232
- Efficient solution, 275
- 80:20 rule, 278
- EIM. *See* Enterprise information management (EIM)
- Embedded method, 120
 - backward feature elimination node, 122–123, 123f
 - feature selection, 122
- Embedded modeling elimination, 253–254
- Ensemble modeling, 41–42, 41f, 177–178, 177f, 201, 249, 256
 - fused structure, 185–191, 186f
 - KNIME and Python, 187–189, 188f
 - Occam's razor, 184
 - power of, 184, 184f
 - predictive power, 186–187
 - types of, 184–185
- Enterprise data warehouse (EDW), 59–60, 277
- Enterprise information management (EIM), 4
- Epigenomic data, 225
- Equal Credit Opportunity Act, 208–209
- Equilibrium propagation (E-prop), 16
- Ernie Bot, 266

- Ernie 3.0 Titan, 266
- Errors in analytical models, 169–173
- Ethical perspectives, 283–284
- Ethics
- data analytics, 282–283
 - and data science, 281–283
 - example for, 281
 - existential-motivational perspective, 282
 - generative models, 284–285
 - normative-deontological perspective, 282
 - secular, 281–282
 - situational-teleological perspective, 282
- ETL. *See* Extraction, transformation and loading (ETL)
- Exaone, 266
- Existential-motivational perspective, 282
- Experian's National Equivalency Score, 260
- Experimental bias, 128
- Explainable AI (XAI), 259–260
- defined, 201–202
 - global and local explanations, 204–205
- Exploration tools, data, 46
- Exploratory data analysis, 28
- Exponential distributions, 165
- Extended markup language (XML), 26
- External data, 62
- Extraction, transformation and loading (ETL), 39, 57
- analytical function, 59–60
 - data extraction operations, 60, 60*f*. *See also* Data, extraction
 - data loading, 65–66
 - data transformation, 65
 - process, 39, 59–66
- Extreme programming software development (XP), 278
- Eye disease, 225
- F**
- Fab Labs concept, 269–270
- Factor independency, 8
- Fair Isaac & Company (FICO), 259
- score, 259
- False-positive predictions, 170–171
- False-positive rate (FPR), 20
- Fast-Fourier Transform (FFT), 226
- Feature derivation
- ratios, 111–112
 - sums, 110–111
 - updating variables, 112
- Feature engineering, 196–197
- Feature extraction, 109–110
- Feature importance, 207, 262
- FeatureLab, 13
- Feature reduction filters, 120–121
- Feature selection. *See also* Dimensionality reduction
- concepts, 117–119
 - data engineering steps, 253–254
 - definition, 253
 - Gini coefficient, 121
 - KNIME examples, 121–122
 - modeling, 117, 122
- Fecal Immunochemical Test (FIT), 249
- Fecal Occult Blood Test (FOBT), 249
- Feedback, 212, 215
- Feed-forward design, 18
- FFT. *See* Fast-Fourier Transform (FFT)
- FICO. *See* Fair Isaac & Company (FICO)
- Fields, 85–86
- Filling missing values, 441
- with constants, 441, 442*f*, 443*f*
 - with formulas, 445–448, 446*f*, 447*f*, 448*f*
 - with model, 449–457, 450*f*, 451*f*, 452*f*, 453*f*, 454*f*, 455*f*, 456*f*, 457*f*, 458*f*
- Filtering of data, 130
- Filtering operations, 244
- Filter method
- correlation, 122–123
 - low variation, 121–122
- Fisher, Ronald, 6–7, 7*f*, 150, 157–158
- FIT. *See* Fecal Immunochemical Test (FIT)
- FIT-DNA test, 249
- Flat-file format, 62, 401–410
- FOBT. *See* Fecal Occult Blood Test (FOBT)
- Formulas, filling missing values with, 445–448, 446*f*, 447*f*, 448*f*
- Foundation models, 15, 17–21
- examples, 18–19
- FPR. *See* False-positive rate (FPR)
- Fraud attacks, 284–285
- Fraud detection, 154
- Functional magnetic resonance imaging (fMRI), 289
- Fused ensemble model, 185–191, 186*f*, 255–256
- F-value, 171
- G**
- Gain
- column, 190
 - curve, 171
- Galton, Francis, 5–6, 145, 160–163
- Gamma distribution, 159
- Gas chromatograph-mass spectrometer (GCMS), 226
- GBT. *See* Gradient Boosted Trees (GBT)
- GCMS. *See* Gas chromatograph-mass spectrometer (GCMS)
- General Data Protection Regulation (GDPR), 63, 153, 201, 209, 260, 282
- compliance, 63
- General Discriminate Analysis, 306, 313*f*
- Generalization data abstraction, 112, 234
- Generalized linear model (GLM), 10, 163–164
- Generalized regression neural networks (GRNN), 31
- Generative AI, 268, 284–285
- Generative Pre-trained Transformer 3 (GPT-3), 266
- Genetic algorithms, 138
- Genomics, 224–225
- Geometric progression, 157
- Gini coefficient (impurity), 121
- caution, 121
 - information gain, 121
- Glaucoma, 225
- GLM. *See* Generalized linear model (GLM)
- Global
- accuracy, 170, 258
 - explanations, 204–205
 - techniques, 205–207
 - minimum error, 118, 119*f*, 353–354, 354*f*
- Gold miner panning, 43, 43*f*
- Gopher, 266
- GPT-3. *See* Generative Pre-trained Transformer 3 (GPT-3)
- Gradient Boosted Trees (GBT), 20*f*, 167, 184, 186, 255, 262, 397–398
- Gradient descent method, 16, 95, 398–399
- Graphical programming, 47, 50–52
- GRNN. *See* Generalized regression neural networks (GRNN)
- Ground truth, 213–214
- Grouping, 105–107, 107*f*, 107*t*
- H**
- Hadoop, 66–67
- Handling outliers. *See* Outlier handling
- Healthcare
- administration and delivery, 227
 - data analytics in, 224
 - future of, 228–229
 - patient-directed, 224
 - predictive analytics in, 223
 - regulatory measures, 224
 - workers, 225
- Heteroscedasticity, 8
- High-order nominals, 72
- High-pass filter, 90
- Histogram in KNIME, 339–351, 340*f*, 341*f*, 342*f*, 343*f*, 344*f*, 345*f*, 346*f*, 347*f*, 348*f*, 349*f*, 350*f*, 351*f*
- HITL. *See* Human-in-the-loop (HITL)
- Homoscedasticity, 8
- Hospice management, KNIME, 393–400, 394*f*, 395*f*, 396*f*, 397*f*, 398*f*, 399*f*, 400*t*
- Household database, 27
- Human
- behavioral modeling, 231
 - cognition, 15
 - computation, 212, 216
 - nature, 233
- Human-centeredness, 218
- HUMANE, 268–269
- Human-in-the-loop (HITL)
- deep learning, 212–213
 - definition, 212
 - use cases, 213–215
 - autonomous vehicles, 214
 - LLMs, 215
 - medical imaging and radiology, 213–214
 - NLP, 214–215
- Hyperparameters, 118–119
- optimization, 167, 179–180, 180*t*
- Hypothesis space, 12–13
- I**
- IBM SPSS modeler, 74, 156, 166–167, 243, 463
- Identity link, 10
- IG. *See* Information gain (IG)
- ImageNet Large Scale Visual Recognition Challenge, 213
- Image recognition, 14
- Imbalanced data sets
- balancing, 96–97
 - ML algorithms, 156
- IML. *See* Interpretable machine learning (IML)
- Improper algorithm use, 175
- Imputation process, 40, 98
- Inadequate experimental design, 175
- In-database modeling, 66

- Independency, 159–160
 - Indexed sequential access method (ISAM)
 - database, 26
 - Index parameter, 165
 - Inductive database approach, 23, 26, 35
 - Industrial revolution, 156, 232
 - Inexact (“fuzzy”) matching, 30
 - Inference testing, 7
 - Information Age (Revolution), 265
 - Information gain (IG), 120–121
 - calculation, 121
 - Gini coefficient, 121
 - Information loop, 43
 - In-house data annotation workforce, 215
 - Insurance risk, 165, 170–171
 - Insurance score, 260
 - Intelligence Age, 265, 268
 - Intelligence revolution
 - data and predictive analytics, 270–271
 - Fab Labs, 269–270
 - HUMANE, 268–269
 - LLMs, 265–268
 - TinyML, 268
 - Intentional bias, 129
 - Interaction effects, 197–199
 - Intermediate table design, 66
 - Internal customers, 75
 - Internal validity, 98
 - Internuncial nodes, 11
 - Interpretable machine learning (IML)
 - decision trees, 203–204
 - linear regression, 202–203
 - Intrinsically linear regression models, 164
 - Irvine Colon Polyp Risk Modeling Project,
 - University of California, 249–254
 - ISAM database. *See* Indexed sequential access method (ISAM) database
- J**
- Jackknife method, 175–176
 - Java Snippet code, 445–448
 - Joined table, 407, 408f, 409
 - Joiner node, 403, 408f, 409, 409f, 410f
 - Joins, 64
 - Jurassic-1, AL21 Labs, 266
 - Just barely good enough (JBGE), 278–279
- K**
- KDD. *See* Knowledge discovery in databases (KDD)
 - Kernel learning machines, 13
 - Kernels, 13
 - K-fold cross-validation, 175–176, 176f
 - k-means clustering, 146
 - KNIME, 187–189, 188f
 - Analytical Platform, 226, 243, 463
 - cell replacer node, 104, 104f
 - Feature Elimination metanode, 354–355
 - feature selection, 121–122
 - histogram in, 339–351, 340f, 341f, 342f, 343f, 344f, 345f, 346f, 347f, 348f, 349f, 350f, 351f
 - Lag Column, 235, 235f
 - local minimum error, 353, 354f
 - metanodes, 185
 - missing values, 99–100
 - modeling workflow, 243, 243f, 255, 255f
 - Occam’s razor, 353
 - parameter optimization loop, 167, 168f
 - KNIME (*Continued*)
 - predictive analytics in, 353–365
 - Random Forest algorithm, 203, 203t, 261
 - RProp MLP algorithm, 261
 - select features, 353
 - strategies for reduction of dimensionality in, 354–365
 - string replacer node, 104
 - temporal analysis functions, 115
 - workflow, 226, 226f
 - XGBoosted Trees algorithm, 203, 261
 - Knowledge discovery, 24, 35, 36f
 - and data analytics, 25f
 - Knowledge discovery in databases (KDD),
 - 3–4, 24, 28, 35
 - Kolmogorov-Smirnov (KS) test
 - caveats, 174
 - statistic, 174
- L**
- Lag variables, 113–114, 233–235, 235f
 - LaMDA, LLMs, 265
 - Large language models (LLMs), 14, 18–21,
 - 201, 212, 223, 265–268
 - Laser projection system, 269
 - Law of parsimony, 44, 118
 - Law of succinctness, 44
 - Learning predictive modeling, 393
 - Learning rate, 118–119
 - Learning style groups, 197
 - Least Squares method, 164
 - LeNet-5 model, 195
 - Level of measurement, 69–72
 - Lex parsimonae, 44, 118
 - Lift curve, 171, 173f
 - LIME. *See* Local Interpretable Model-Agnostic Explanations (LIME)
 - Linear additivity, 8
 - Linear correlation, 100
 - Linearity, 108, 159
 - Linear regression (LR), 95, 112f, 153–154,
 - 160–163, 202–203, 449
 - collinearity among variables in, 166
 - piecewise, 165–166
 - response surface, 160–163, 161f
 - variable interactions in, 166
 - Linear response analysis, 157, 157f
 - Link functions, 10
 - List-wise deletion, 91, 97–98
 - LLMs. *See* Large language models (LLMs)
 - Local explanations, 204–205
 - techniques, 207–208
 - Local Interpretable Model-Agnostic Explanations (LIME), 208
 - Local minimum error, KNIME, 118, 119f, 353, 354f
 - Logistic function, 18, 18f
 - Logistic regression (LR), 95–96, 120, 145, 153,
 - 164, 202t, 252–254
 - Logit
 - model, 10
 - regression, 164
 - Log transform
 - normal distribution, 108
 - parametric statistical algorithms
 - machine learning, 109
 - normality and linearity assumptions, 108
 - problem, 108–109
 - Long short-term memory (LSTM), 17
 - Loom systems, 13
 - Low-level programmers, 20–21
 - Low-order nominals, 71–72
 - Low-pass filter, 90
 - Low variation filtering, 121–122, 253
 - LSTM. *See* Long short-term memory (LSTM)
 - Lung cancer, 226–227
- M**
- Machine learning (ML), 4, 95, 109, 117, 153,
 - 179–180, 212–213, 227, 242, 244, 250, 252–253, 256, 260, 284, 290
 - analyzing imbalanced data sets with, 156
 - automation, 13–14
 - numerical prediction, 166
 - preliminary operations, 166–167
 - problems in, 137
 - technology, 24, 227, 250
 - third generation, 11–14
 - tools, 176
 - MAE. *See* Mean absolute error (MAE)
 - Mapping, 13
 - MAR. *See* Missing at random (MAR)
 - Maximum Likelihood method, 164
 - MCAR. *See* Missing completely at random (MCAR)
 - MD. *See* Missing data (MD)
 - MDR. *See* Multifactorial dimensionality reduction (MDR)
 - Mean absolute error (MAE), 173–174,
 - 449–457
 - Mean squared error (MSE), 173
 - Measurement bias, 128–129
 - MECE. *See* Mutually exclusive and categorically exhaustive (MECE)
 - Mechanical-algorithmic processes, 269
 - Medical/business tutorial, 367–371, 368f,
 - 369f, 370f, 371f, 372f, 373f, 374f, 375–376, 375t, 376f, 377f, 378f, 379f, 380f, 380t, 381, 381f, 382f, 383f, 384f, 385f, 386, 386f, 386t, 387f, 387t, 388f, 388t, 389, 389f, 390f, 391f
 - Medical data analytics, 223, 227
 - Medical informatics, 249–258
 - data and process models, 224
 - Medicare, 367
 - Megatron-Turing NLG, 266
 - Memory, 17
 - Menu-based procedural platforms, 47–50
 - Merge operation, 64
 - Metabolomics, 225
 - lung cancer, 226–227
 - pancreatic cancer, 226–227
 - Meta-modeling, 177–178, 189–190, 190f
 - Microeconomic approach, 26
 - MidWest Company Personality Data,
 - 339–351, 340f, 341f, 342f, 343f, 344f, 345f, 346f, 347f, 348f, 349f, 350f, 351f
 - Miscellaneous model enhancement options, 180
 - Missing at random (MAR), 99, 441
 - linear correlation, 100, 100f
 - regression, 100
 - Missing completely at random (MCAR),
 - 99–101, 441
 - Missing data (MD)
 - classification, 98–99

- Missing data (MD) (*Continued*)
 consequence, 98
 construct validity, 98
 correlation analysis, 100
 diagnosis, 99
 filling missing values. *See* Filling missing values
 internal validity, 98
 KNIME, 99–100
 linear correlation, MAR, 100, 100f
 missing values, 156, 181, 244
 reasons for, 97–98
 regression, MAR, 100
- Missing not at random (MNAR), 99, 441
- ML. *See* Machine learning (ML)
- MLP. *See* Multilayer perceptron (MLP)
- MLP-neural network model, 235, 236t, 237, 237t, 261, 262t
- MLR. *See* Multiple linear regression (MLR)
- MNAR. *See* Missing not at random (MNAR)
- Model(ing)
 accuracy, 256–258
 activities, 41–43
 CRISP-DM, 41–43, 41f
 algorithm, 427
 architecture, 255–256
 enhancement
 checklist, 180–181
 techniques, 177–180
 filling missing values with, 449–457, 450f, 451f, 452f, 453f, 454f, 455f, 456f, 457f, 458f
 issues, 149–150
 operations, 149
 process, evaluation and enhancement, 169, 170f
 theoretic for data analytics, 25
- Model Management and Operations (ModelOps), 228
- Modern statistical analysis, second generation, 6–11
- Modular robotic systems, 269
- Morality, 282
- Moral reasoning, 283
- MSE. *See* Mean squared error (MSE)
- mT5, 266
- Multicollinearity problem, 8, 166, 253
- Multifactorial dimensionality reduction (MDR), 120
- Multilayer perceptron (MLP), 193–194
- Multionics, 224–225
- Multiple curvilinear regression, 10
- Multiple linear regression (MLR), 96, 145, 163
- Multiple regression, 370–371, 374f
- Mutual improvement model, 218
- Mutually exclusive and categorically exhaustive (MECE), 156
- N**
- Natural language processing (NLP), 14, 196, 213, 266b
 use cases, 214–215
- National Council Licensure Examination (NCLEX), 317–322, 319t, 320f, 321f, 322f, 323f, 324f, 325f, 326f, 327f, 328f, 329f, 330–337, 330f, 331f, 332f, 333f, 334f, 335f, 336f
 case study, 318
- NCLEX examination (*Continued*)
 dataset and expected strength of predictors, 319–320, 319t
 decision management, 318
 literature review, 318–319
 research question, 318
- Nearest-neighbor method, 156
- Neocognition, 195
- Neural networks (NN), 17, 41, 96, 190, 193, 197, 213, 299
- NLEs. *See* Nonlinear events (NLEs)
- NLP. *See* Natural language processing (NLP)
- Nonlinear estimation techniques, 164–166
 logit regression, 164
 piecewise linear regression, 165–166
 Poisson regression, 165
 probit regression, 164
- Nonlinear events (NLEs), 12, 233
- Nonlinearity, 15, 160
- Nonlinear regression, 164–166
- Nonlinear relationships analyzing methods, 164, 164f
- Nonnormal distribution, 160
- Nonparametric analysis, 6
- Normal distribution, 158, 158f
- Normality, 108, 158–159
 assumption, 158
 Central Limit Theorem, 159–163
- Normalization, 95–96, 203
- Normative-deontological perspective, 282
- Null hypothesis, testing of, 287–288
- Numerical prediction, 157
- Numerical variables, 8, 103, 156
- O**
- Occam's razor, 117–118, 184, 253
 KNIME, 353
- OLTP systems. *See* Online transactional processing (OLTP) systems
- OMOP. *See* Operational Medical Outcomes Partnership (OMOP)
- One-hot coding, 71
- One-hot encoding, 397–398, 427–433
- One-hot variable processing, 244
- Online database connectivity (ODBC)
 driver, 58
 configuration, 63
 connectors, 45–46
- Online transactional processing (OLTP) systems, 62
- Open Midwest Data with STATISTICA, 293–299, 294f, 295f, 296f, 297f, 298f, 299f, 300f, 301f, 302f
- Operational Medical Outcomes Partnership (OMOP), 250
 code frequency analysis, 253
- Operations research (OR), 138
- OPT-IML, Meta AI, 266
- Ordinal variables, 72
- Outlier handling, 90–93, 435–439, 435f, 436f, 437f, 438f, 439f, 440f
 challenges, 91
 definition, 91
 detection algorithms, 91
 procedures, 92f
 ways to, 91–92
- Oversampling, 97, 97t, 254
- P**
- PA. *See* Predictive analytics (PA)
- Pair-wise deletion method, 97–98
- PaLM, LLMs, 266
- Pancreatic cancer, 226–227
- PanGu-Alpha, 266
- Parameter adjustment, 179–180
- Parameter optimization loop, KNIME, 167, 168f
- Parametric model, 7, 42–43
 assumptions, 7–8
 independency, 159
 linearity, 159
 linear regression, 160–163
 normality, 158–159
- Parametric predictive systems, 8
- Parametric statistical analysis, 95–96, 107, 157–158, 254, 289, 397–398
 machine learning, 109
 normality and linearity assumptions, 108
 problem, 108–109
- Pareto's principle, 278
- Partial dependence plots (PDPs), 205–206, 206f
- Partial joins, 64
- Partitioning, 256
 node, 398, 399f
 operation, 244
- Patient-directed healthcare, 224
- Patient level, grouping to, 251
- Pattern-seeking procedure, 375
- PAW. *See* Predictive Analytics World (PAW)
- PCA. *See* Principal components analysis (PCA)
- PDPs. *See* Partial dependence plots (PDPs)
- Pearson, Karl, 5–6
- Pearson product-moment correlation
 coefficient. *See* Simple correlation coefficient
- Pediatric ICU admissions, 227
- Perceptron, 11, 11f
- Permutation feature importance, 207
- Personal AI, 270–271
- Personalized medicine. *See* Precision medicine
- Personalized_precision healthcare, 227
- Physical data mart, 27
- Physical *versus* logical data mart, 66
- Piecewise linear regression, 165–166, 165f
- Plagiarism in schools, 284
- Plato, 9–10, 9f, 150, 150f
- PMML. *See* Predictive modeling markup language (PMML)
- Pneumonia, 252
- Poisson distribution, 165
- Poisson regression, 10
- Polynomial regression analysis, 100, 101f, 164
- Pooling, 16–17
- Positive semi-definite matrix, 275
- Post hoc explanations, 204
- Precancerous colon polyps, 152, 183, 225
- Precision medicine, 224
- Precision model, 170–171
- Prediction. *See also* Numerical prediction
 accuracies, 250t, 261, 261f, 262t
 fusion node, 186
 implications, 8

- Prediction (*Continued*)
 tool development, 225
- Predictive analytics (PA), 28, 287, 290
 activities, 28–29
 applications, 13–14, 30
 challenges, 30
 common mistakes made in PA projects, 132–134
 data knowledge, 57
 development in, current trends of, 21
 feature selection, 118
 forecasts, 228
 history of, 4–21
 in KNIME, 353–365
 in medical and healthcare, 223–224
 model-theoretic for, 26
 multiomics data, 224–225
 patient's diagnosis and treatment, 225
 pediatric ICU admissions, 227
 perceptions, 275
 as Personal AI or Company AI, 270–271
 SAP, 13–14
 terminology, 3–4
 tool selection, 4
- Predictive Analytics World (PAW), 3–4
- Predictive modeling markup language (PMML), 26, 397–398
- Predictive models, 152
 classes, 153–154
versus prescriptive models, 228
 types of, 153–168
 classification-type models, 154–155
 regression-type models, 156–158
- Predictive power, 169–171
- Predictors, 103
 different sets of, 180
 variables, 44, 103, 180, 190, 191*t*, 235–237, 245, 245*f*
- Prescriptive modeling, 227–228
- Pretrained models. *See* Foundation models
- Principal components analysis (PCA), 139
- Probit
 model, 10
 regression, 164
- Problem solving, 35
- Product-centric database systems, 57
- Professional development, 52–53
- Proteomic data, 225
- Pruning trees, 143
- Pure coding languages
 Python, 47
 R, 46–47
 R *versus* Python, 47
 SAS, 46
 SAS *versus* R, 46–47
- P-value
 principles, 288–289
 statistical analysis, 287–288
 problem of significance, 287–288
- Python, 47, 176–177, 177*t*, 187–189, 188*f*, 267, 267*b*
- Q**
- Qualitative data abstraction, 112, 234
- Quality analysis, 251
- Quantitative information, 76
- Quasi-Newton method, 164
- R**
- R, 463
 language, 46–47
versus Python, 47
 Python combined, 463
- Radial basis function (RBF), 41, 138
- RAE. *See* Relative absolute error (RAE)
- Random error, 169, 171–173
 assessment, 173–177
 evaluation of models, 171–173
- Random Forest (RF), 255–256, 256*f*, 261, 261*f*, 267, 267*b*, 362–365
- Random walk, 118
- RBF. *See* Radial basis function (RBF)
- RDBMS. *See* Relational database management systems (RDBMS)
- Recall, 171
- Receiver operating curve (ROC), 171, 172*f*, 246, 246*f*, 257*f*, 258, 261, 261*f*
- Recoding of data, 65, 88–89, 130, 419–426, 420*f*, 421*f*, 422*f*, 423*f*, 424*f*, 425*f*, 426*f*
- Recoding outliers, 92
- Record (case), 103
- Record order, 63, 64*f*
- Recurrent neural networks (RNN), 17
- Recursive partitioning, 142–143
- Referral group, 386
- Regression to the mean, 145, 160–163
- Regression-type models, 153, 156–158, 171
- Regulatory measures, 224
- Reinforcement learning, 14, 215
 human feedback, 212
- Relational database management systems (RDBMS), 26
- Relative absolute error (RAE), 174
- Relative squared error (RSE), 174
- Replication/reproducibility analysis, 288–289
- Representational learning (RL), 16
- Reproducibility crisis, 11
- The Reproducibility Project (Errington), 289
- Resampling/techniques, 175–177
- Residential dwelling, 156
- Residuals, 161, 162*f*
- Response surface, linear regression, 160–163, 161*f*
- Retrieval by content, data mining activity, 29
- RF. *See* Random Forest (RF)
- RL. *See* Representational learning (RL)
- RMSE. Root mean squared error (RMSE)
- Robot work force, 269
- Robustly Optimized BERT (RoBERTa), 18–19
- Robustness, 155
- ROC. *See* Receiver operating curve (ROC)
- Root mean squared error (RMSE), 173
- Row filtering operations, 105, 105*f*
- Row-wise deletion, 91, 97–98
- RSE. *See* Relative squared error (RSE)
- R² value, 174
- S**
- Sampling/data analysis, 228
- Sampling errors, 175
 assessment technique
 bootstrap method, 176
 Jackknife, 175–176
 K-fold cross-validation, 176, 176*f*
- SAP predictive analytics, 13–14
- SAS. *See* Statistical analysis system (SAS)
- Scale variables, 72
- Scientific method, 23–24, 288
- Sculptural theory of subtraction, 117
- SDM. *See* STATISTICA data miner (SDM)
- Secular ethics, 281–282
- Semiautomated data mining recipes, 47
- Sensitivity, 171, 256, 258, 400
- Service level agreement (SLA), 126–127
- SES. *See* Socioeconomic status (SES)
- Sex, salary and education level, 198*f*, 198*t*, 199*t*
- Shallow neural network, 199, 199*f*
- Shapley Additive exPlanations (SHAP), 207–208, 207*t*
- Short-list, 115, 119, 253
- Signal processing, 90
- Simple correlation coefficient, 174
- Single estimator model, 183–184
- Single-family dwelling, 156
- Single nucleotide polymorphisms (SNPs), 120
- Situational-teleological perspective, 282
- Skytree machine-learning software, 14
- SLA. *See* Service level agreement (SLA)
- SMEs. *See* Subject matter experts (SMEs)
- SMOTE. *See* Synthetic Minority Over-sampling Technique (SMOTE)
- Snapshot dependency theory, 235
- SNPs. *See* Single nucleotide polymorphisms (SNPs)
- Socioeconomic status (SES), 72
- Sociotechnical system, 218
- Software routines, 69
- Sonnet 151, 117
- Source data, 103
- Sparrow, LLMs, 266
- Specificity, 171, 256, 258, 400
- Split plot ANOVA technique, 6
- Spreadsheets, 61
- SPSS
 Clementine, 193
 modeler, 317–322, 319*t*, 320*f*, 321*f*, 322*f*, 323*f*, 324*f*, 325*f*, 326*f*, 327*f*, 328*f*, 329*f*, 330–337, 330*f*, 331*f*, 332*f*, 333*f*, 334*f*, 335*f*, 336*f*
 bagging, 322–330, 325*f*
 boosting, 322–330, 324*f*
 interpreting model output, 324–330
 modeling workflow, 320–322
 selection procedure and evaluation, 322–323
- SQL. *See* Structured query language (SQL)
- Standardization of data, 95–96, 130, 145, 160, 254
- Star-schema database structure, 27, 27*f*
- STATISTICA
 data mining, 303–314, 303*f*, 304*f*, 305*f*, 306*f*, 307*f*, 308*f*, 309*f*, 310*f*, 311*f*, 312*f*, 313*f*, 314*f*, 315*f*, 316*f*, 317–322, 319*t*, 320*f*, 321*f*, 322*f*, 323*f*, 324*f*, 325*f*, 326*f*, 327*f*, 328*f*, 329*f*, 330–337, 330*f*, 331*f*, 332*f*, 333*f*, 334*f*, 335*f*, 336*f*
 recipe, 330–333
 model output and evaluation, 333
 rules creation, 333–335
- STATISTICA data miner (SDM), 13–14, 98, 166–167

- Statistica data miner (SDM) (*Continued*)
 machine learning modeling, 47, 48f
 misclassification rates, 47, 50f
 Random Forest model, 47, 49f
 recipe, 330–333, 330f, 331f, 332f, 333f, 334f, 335f, 336f
 Tibco, 47, 463
 variables selection screen, 47, 50f
 Statistical analysis system (SAS), 46
 Statistical learning theory, 12–14
 Statistical modeling, 24, 152–153
 Statistical quality control systems, 224
 Statistics
 history, 4–21
 modern, 6
 node, 435
 Stepwise regression, 181
 Stopping functions, 120
 String manipulation, 89–90
 Structured query language (SQL), 30, 45, 58
 Student recruitment, 241
 Student retention, 241–243
 Subjective priors, 5–6, 157–158
 Subject matter experts (SMEs), 215–216
 Sufficiency paradigm, 276
 Supervised classification, 154
 Support vector machines (SVMs), 117–118, 138, 312–314, 315f
 Surrogate
 models, 204, 206–207
 variables, 103, 174, 180
 Sustainable agriculture, 275–276
 SVMs. *See* Support vector machines (SVMs)
 Synthetic Minority Over-sampling Technique (SMOTE), 96–97, 97t, 156, 226, 244, 254
 Systematic error assessment, 169, 174–175
 inadequate experimental design, 175
 sampling errors, 175
- T**
 Target
 classes, 254
 variable, 103, 156, 211–212
 Tautology, 65, 449
 TDNNs. *See* Time-delated neural networks (TDNNs)
 Team modeling, 215
 Technical data correction, 85–89
 Temporal data, 17
 abstraction, 233–234, 234f
 mining, 114–115
 Testing/holdout data set, 367
 Text files, 62
 Text mining, 252
 Textual variables, 72, 252–253
 Theory of Forms, 150
 Time-delated neural networks (TDNNs), 17
 Time-series
 analyses, 237–238
 filtering, 90
 traditional, 114–115
 TinyML, 268
- Tool selection, 52–53
 Top-down *versus* bottom-up solution
 architecture, 277
 TPOT. *See* Tree-based Pipeline Optimization Tool (TPOT)
 TPR. *See* True positive rate (TPR)
 Traditional machine learning, 193–200, 194t, 211
 Traditional statistical analysis, 23
 Training data set, 367
 Transcriptomic data, 225
 Transformers, 18–19, 265
 Transparency, 201, 209
 TransRisk, 260
 Tree-based Pipeline Optimization Tool (TPOT), 14
 True positive rate (TPR), 20
 Trustworthy AI, 209
 Turing test, 6–7, 212–213
 Tutorials
 boosted trees, 388t, 389, 389f, 391f
 cleaning and recoding of data, 419–426, 420f, 421f, 422f, 423f, 424f, 425f, 426f
 companion web page, 461–462
 data description, 413–416, 414f, 415f, 416f, 417f
 data sources, merging, 401–410, 402f, 403f, 404f, 405f, 406f, 407f, 408f, 409f, 410f, 411f, 412f
 dummy coding category variables, 427–433, 428f, 429f, 430f, 431f, 432f, 433f
 filling missing values with
 constants, 441, 442f, 443f
 formulas, 445–448, 446f, 447f, 448f
 model, 449–457, 450f, 451f, 452f, 453f, 454f, 455f, 456f, 457f, 458f
 KNIME, 353–365, 354f, 355f, 356f, 357f, 358f, 359f, 360f, 361f, 362f, 363f, 364f, 365f, 393–400, 394f, 395f, 396f, 397f, 398f, 399f, 400t
 medical/business tutorial, 367–371, 368f, 369f, 370f, 371f, 372f, 373f, 374f, 375–376, 375t, 376f, 377f, 378f, 379f, 380f, 380t, 381, 381f, 382f, 383f, 384f, 385f, 386, 386f, 386t, 387f, 387t, 388f, 388t, 389, 389f, 390f, 391f
 MidWest Company Personality Data, 339–351, 340f, 341f, 342f, 343f, 344f, 345f, 346f, 347f, 348f, 349f, 350f, 351f
 Open Midwest Data with Statistica, 2
 93–299, 294f, 295f, 296f, 297f, 298f, 299f, 300f, 301f, 302f
 removal of outliers, 435–439, 435f, 436f, 437f, 438f, 439f, 440f
 STATISTICA data mining, 303–314, 303f, 304f, 305f, 306f, 307f, 308f, 309f, 310f, 311f, 312f, 313f, 314f, 315f, 316f
 Type 1 detection rate, 258
 Type 1 precancerous colon polyps
 data engineering steps, 250t
 incidence, 249
 Irvine Colon Polyp Risk Modeling Project, 249–254
 modeling, 254–256
 results, 256–258
 screening tests, 249
- U**
 UCI Colonoscopy Quality Database (UCICQD), 251
 Undersampling, 96–97, 97t, 254
 Understanding and problem solving, 35
 Unique-concatenate aggregation function, 106–107, 109–110
 Univariate analysis, 38, 73, 73t
 Unsupervised classification approach, 154
 Unsupervised learning technique, 146
 Up-sell modeling, 231, 235
 Utilitarianism, 282–283
- V**
 Validity checking, 85–86
 arithmetic correctness, 89
 Value chain, 85, 86f
 Value Counter node, 413
 Vanishing gradient problem, 17
 Vantage score, 260
 Variables, 85–86, 103
 derivation, 252–253
 exclusion, 254, 254f
 features *versus*, 117–118
 filtering operations, 104–105
 importance, 203, 262
 interaction in linear regression, 166
 selection screen, 297f, 301f
 selection methods, 119–120, 144
 transformations, 107–109
 types
 binary, 71, 71t
 dates, 72
 high-order nominals, 72
 low-order nominals, 71–72
 one value, 71
 ordinal, 72
 pairs, 71
 scale, 72
 text, 72
 unique identifiers and IDs, 70–71
 V-fold cross-validation, 143, 146
 Virtual ATI (VATI), 317, 319
 Virtual data marts. *See* Data, mart – virtual
 Visual data sensemaking, 76
 Volatile organic compounds (VOCs), 226
- W**
 Waterfall process, 35, 36f
 Wearable multimedia device, 269
 Western Seminary, retention history of, 242–243
 Wrapper program, 120
- X**
 XAI. *See* Explainable AI (XAI)
 XGBoosted Trees (XGB), 179–180, 180t, 186, 201, 253, 255, 261, 262t, 393, 398–399, 400t
 XML. *See* Extended markup language (XML)
 Xpanse Analytics, 14
- Z**
 Zoster (Shingles) vaccination, 252