

Undergraduate Topics in Computer Science

Max Bramer

# Principles of Data Mining

*Fourth Edition*



 Springer

The Springer logo consists of a stylized white chess knight piece on a black square, followed by the word "Springer" in a serif font.

# Undergraduate Topics in Computer Science

‘Undergraduate Topics in Computer Science’ (UTiCS) delivers high-quality instructional content for undergraduates studying in all areas of computing and information science. From core foundational and theoretical material to final-year topics and applications, UTiCS books take a fresh, concise, and modern approach and are ideal for self-study or for a one- or two-semester course. The texts are all authored by established experts in their fields, reviewed by an international advisory board, and contain numerous examples and problems, many of which include fully worked solutions.

More information about this series at <http://www.springer.com/series/7592>

Max Bramer

# Principles of Data Mining

Fourth Edition

 Springer

Prof. Max Bramer  
School of Computing  
University of Portsmouth  
Portsmouth, Hampshire, UK

*Series editor*  
Ian Mackie

*Advisory board*

Samson Abramsky, University of Oxford, Oxford, UK  
Chris Hankin, Imperial College London, London, UK  
Mike Hinchey, University of Limerick, Limerick, Ireland  
Dexter C. Kozen, Cornell University, Ithaca, USA  
Andrew Pitts, University of Cambridge, Cambridge, UK  
Hanne Riis Nielson, Technical University of Denmark, Kongens Lyngby, Denmark  
Steven S. Skiena, Stony Brook University, Stony Brook, USA  
Iain Stewart, University of Durham, Durham, UK

ISSN 1863-7310  
Undergraduate Topics in Computer Science  
ISBN 978-1-4471-7492-9  
DOI 10.1007/978-1-4471-7493-6

ISSN 2197-1781 (electronic)  
ISBN 978-1-4471-7493-6 (eBook)

© Springer-Verlag London Ltd., part of Springer Nature 2007, 2013, 2016, 2020

The author(s) has/have asserted their right(s) to be identified as the author(s) of this work in accordance with the Copyright, Designs and Patents Act 1988.

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by the registered company Springer-Verlag London Ltd. part of Springer Nature.

The registered company address is: The Campus, 4 Crinan Street, London, N1 9XW, United Kingdom

# *About This Book*

This book is designed to be suitable for an introductory course at either undergraduate or masters level. It can be used as a textbook for a taught unit in a degree programme on potentially any of a wide range of subjects including Computer Science, Business Studies, Marketing, Artificial Intelligence, Bioinformatics and Forensic Science. It is also suitable for use as a self-study book for those in technical or management positions who wish to gain an understanding of the subject that goes beyond the superficial. It goes well beyond the generalities of many introductory books on Data Mining but — unlike many other books — you will not need a degree and/or considerable fluency in Mathematics to understand it.

Mathematics is a language in which it is possible to express very complex and sophisticated ideas. Unfortunately it is a language in which 99% of the human race is not fluent, although many people have some basic knowledge of it from early experiences (not always pleasant ones) at school. The author is a former Mathematician who now prefers to communicate in plain English wherever possible and believes that a good example is worth a hundred mathematical symbols.

One of the author's aims in writing this book has been to eliminate mathematical formalism in the interests of clarity wherever possible. Unfortunately it has not been possible to bury mathematical notation entirely. A 'refresher' of everything you need to know to begin studying the book is given in Appendix A. It should be quite familiar to anyone who has studied Mathematics at school level. Everything else will be explained as we come to it. If you have difficulty following the notation in some places, you can usually safely ignore it, just concentrating on the results and the detailed examples given. For those who would like to pursue the mathematical underpinnings of Data Mining in greater depth, a number of additional texts are listed in Appendix C.

No introductory book on Data Mining can take you to research level in the subject — the days for that have long passed. This book will give you a good grounding in the principal techniques without attempting to show you this year’s latest fashions, which in most cases will have been superseded by the time the book gets into your hands. Once you know the basic methods, there are many sources you can use to find the latest developments in the field. Some of these are listed in Appendix C. The other appendices include information about the main datasets used in the examples in the book, many of which are of interest in their own right and are readily available for use in your own projects if you wish, and a glossary of the technical terms used in the book.

Self-assessment Exercises are included for each chapter to enable you to check your understanding. Specimen solutions are given in Appendix E.

## Note on the Fourth Edition

Since the first edition there has been a vast and ever-accelerating increase in the volume of data available for data mining. According to IBM (in 2016) 2.5 billion billion bytes of data is produced every day from sensors, mobile devices, online transactions and social networks, with 90 percent of the data in the world having been created in the last two years alone. Today the amount of healthcare data available in the world is estimated as over 2 trillion gigabytes. To reflect the growing popularity of ‘deep learning’ a new final chapter has been added which gives a detailed introduction to one of the most important types of neural net and shows how it can be applied to classification tasks.

## Acknowledgements

I would like to thank my daughter Bryony for drawing many of the more complex diagrams and for general advice on design, Dr. Frederic Stahl for advice on Chapters 21 and 22 and my wife Dawn for her very valuable comments on draft chapters and for preparing the index. The responsibility for any errors that may have crept into the final version remains with me.

Max Bramer  
Emeritus Professor of Information Technology  
University of Portsmouth, UK  
May 2020

# Contents

<b>1. Introduction to Data Mining</b>	<b>1</b>
1.1 The Data Explosion	1
1.2 Knowledge Discovery	2
1.3 Applications of Data Mining	3
1.4 Labelled and Unlabelled Data	4
1.5 Supervised Learning: Classification	5
1.6 Supervised Learning: Numerical Prediction	7
1.7 Unsupervised Learning: Association Rules	7
1.8 Unsupervised Learning: Clustering	8
<b>2. Data for Data Mining</b>	<b>9</b>
2.1 Standard Formulation	9
2.2 Types of Variable	10
2.2.1 Categorical and Continuous Attributes	12
2.3 Data Preparation	12
2.3.1 Data Cleaning	13
2.4 Missing Values	15
2.4.1 Discard Instances	15
2.4.2 Replace by Most Frequent/Average Value	15
2.5 Reducing the Number of Attributes	16
2.6 The UCI Repository of Datasets	17
2.7 Chapter Summary	18
2.8 Self-assessment Exercises for Chapter 2	18
Reference	19



<b>3. Introduction to Classification: Naïve Bayes and Nearest Neighbour</b>	21
3.1 What Is Classification?	21
3.2 Naïve Bayes Classifiers	22
3.3 Nearest Neighbour Classification	29
3.3.1 Distance Measures	32
3.3.2 Normalisation	35
3.3.3 Dealing with Categorical Attributes	36
3.4 Eager and Lazy Learning	36
3.5 Chapter Summary	37
3.6 Self-assessment Exercises for Chapter 3	37
<b>4. Using Decision Trees for Classification</b>	39
4.1 Decision Rules and Decision Trees	39
4.1.1 Decision Trees: The Golf Example	40
4.1.2 Terminology	41
4.1.3 The <i>degrees</i> Dataset	42
4.2 The TDIDT Algorithm	45
4.3 Types of Reasoning	47
4.4 Chapter Summary	48
4.5 Self-assessment Exercises for Chapter 4	48
References	48
<b>5. Decision Tree Induction: Using Entropy for Attribute Selection</b>	49
5.1 Attribute Selection: An Experiment	49
5.2 Alternative Decision Trees	50
5.2.1 The Football/Netball Example	51
5.2.2 The <i>anonymous</i> Dataset	53
5.3 Choosing Attributes to Split On: Using Entropy	54
5.3.1 The <i>lens24</i> Dataset	55
5.3.2 Entropy	57
5.3.3 Using Entropy for Attribute Selection	58
5.3.4 Maximising Information Gain	60
5.4 Chapter Summary	61
5.5 Self-assessment Exercises for Chapter 5	61
<b>6. Decision Tree Induction: Using Frequency Tables for Attribute Selection</b>	63
6.1 Calculating Entropy in Practice	63
6.1.1 Proof of Equivalence	64
6.1.2 A Note on Zeros	66

6.2	Other Attribute Selection Criteria: Gini Index of Diversity . . . .	66
6.3	The $\chi^2$ Attribute Selection Criterion . . . . .	68
6.4	Inductive Bias . . . . .	71
6.5	Using Gain Ratio for Attribute Selection . . . . .	73
6.5.1	Properties of Split Information . . . . .	74
6.5.2	Summary . . . . .	75
6.6	Number of Rules Generated by Different Attribute Selection Criteria . . . . .	75
6.7	Missing Branches . . . . .	76
6.8	Chapter Summary . . . . .	77
6.9	Self-assessment Exercises for Chapter 6 . . . . .	77
	References . . . . .	78
<b>7.</b>	<b>Estimating the Predictive Accuracy of a Classifier . . . . .</b>	<b>79</b>
7.1	Introduction . . . . .	79
7.2	Method 1: Separate Training and Test Sets . . . . .	80
7.2.1	Standard Error . . . . .	81
7.2.2	Repeated Train and Test . . . . .	82
7.3	Method 2: $k$ -fold Cross-validation . . . . .	82
7.4	Method 3: $N$ -fold Cross-validation . . . . .	83
7.5	Experimental Results I . . . . .	84
7.6	Experimental Results II: Datasets with Missing Values . . . . .	86
7.6.1	Strategy 1: Discard Instances . . . . .	87
7.6.2	Strategy 2: Replace by Most Frequent/Average Value . . . . .	87
7.6.3	Missing Classifications . . . . .	89
7.7	Confusion Matrix . . . . .	89
7.7.1	True and False Positives . . . . .	90
7.8	Chapter Summary . . . . .	91
7.9	Self-assessment Exercises for Chapter 7 . . . . .	91
	Reference . . . . .	92
<b>8.</b>	<b>Continuous Attributes . . . . .</b>	<b>93</b>
8.1	Introduction . . . . .	93
8.2	Local versus Global Discretisation . . . . .	95
8.3	Adding Local Discretisation to TDIDT . . . . .	96
8.3.1	Calculating the Information Gain of a Set of Pseudo- attributes . . . . .	97
8.3.2	Computational Efficiency . . . . .	102
8.4	Using the ChiMerge Algorithm for Global Discretisation . . . . .	105
8.4.1	Calculating the Expected Values and $\chi^2$ . . . . .	108
8.4.2	Finding the Threshold Value . . . . .	113
8.4.3	Setting <i>minIntervals</i> and <i>maxIntervals</i> . . . . .	113

8.4.4	The ChiMerge Algorithm: Summary . . . . .	115
8.4.5	The ChiMerge Algorithm: Comments . . . . .	115
8.5	Comparing Global and Local Discretisation for Tree Induction	116
8.6	Chapter Summary . . . . .	118
8.7	Self-assessment Exercises for Chapter 8 . . . . .	118
	Reference . . . . .	119
<b>9.</b>	<b>Avoiding Overfitting of Decision Trees . . . . .</b>	<b>121</b>
9.1	Dealing with Clashes in a Training Set . . . . .	122
9.1.1	Adapting TDIDT to Deal with Clashes . . . . .	122
9.2	More About Overfitting Rules to Data . . . . .	127
9.3	Pre-pruning Decision Trees . . . . .	128
9.4	Post-pruning Decision Trees . . . . .	130
9.5	Chapter Summary . . . . .	135
9.6	Self-assessment Exercise for Chapter 9 . . . . .	136
	References . . . . .	136
<b>10.</b>	<b>More About Entropy . . . . .</b>	<b>137</b>
10.1	Introduction . . . . .	137
10.2	Coding Information Using Bits . . . . .	140
10.3	Discriminating Amongst $M$ Values ( $M$ Not a Power of 2) . . . .	142
10.4	Encoding Values That Are Not Equally Likely . . . . .	143
10.5	Entropy of a Training Set . . . . .	146
10.6	Information Gain Must Be Positive or Zero . . . . .	147
10.7	Using Information Gain for Feature Reduction for Classification Tasks . . . . .	149
10.7.1	Example 1: The <i>genetics</i> Dataset . . . . .	150
10.7.2	Example 2: The <i>bcst96</i> Dataset . . . . .	154
10.8	Chapter Summary . . . . .	156
10.9	Self-assessment Exercises for Chapter 10 . . . . .	156
	References . . . . .	156
<b>11.</b>	<b>Inducing Modular Rules for Classification . . . . .</b>	<b>157</b>
11.1	Rule Post-pruning . . . . .	157
11.2	Conflict Resolution . . . . .	159
11.3	Problems with Decision Trees . . . . .	162
11.4	The Prism Algorithm . . . . .	164
11.4.1	Changes to the Basic Prism Algorithm . . . . .	171
11.4.2	Comparing Prism with TDIDT . . . . .	172
11.5	Chapter Summary . . . . .	173
11.6	Self-assessment Exercise for Chapter 11 . . . . .	173
	References . . . . .	174

<b>12. Measuring the Performance of a Classifier</b> .....	175
12.1 True and False Positives and Negatives .....	176
12.2 Performance Measures .....	178
12.3 True and False Positive Rates versus Predictive Accuracy .....	181
12.4 ROC Graphs .....	182
12.5 ROC Curves .....	184
12.6 Finding the Best Classifier .....	185
12.7 Chapter Summary .....	186
12.8 Self-assessment Exercise for Chapter 12 .....	187
<b>13. Dealing with Large Volumes of Data</b> .....	189
13.1 Introduction .....	189
13.2 Distributing Data onto Multiple Processors .....	192
13.3 Case Study: PMCRI .....	194
13.4 Evaluating the Effectiveness of a Distributed System: PMCRI ..	197
13.5 Revising a Classifier Incrementally .....	201
13.6 Chapter Summary .....	207
13.7 Self-assessment Exercises for Chapter 13 .....	207
References .....	208
<b>14. Ensemble Classification</b> .....	209
14.1 Introduction .....	209
14.2 Estimating the Performance of a Classifier .....	212
14.3 Selecting a Different Training Set for Each Classifier .....	213
14.4 Selecting a Different Set of Attributes for Each Classifier .....	214
14.5 Combining Classifications: Alternative Voting Systems .....	215
14.6 Parallel Ensemble Classifiers .....	219
14.7 Chapter Summary .....	219
14.8 Self-assessment Exercises for Chapter 14 .....	220
References .....	220
<b>15. Comparing Classifiers</b> .....	221
15.1 Introduction .....	221
15.2 The Paired t-Test .....	223
15.3 Choosing Datasets for Comparative Evaluation .....	229
15.3.1 Confidence Intervals .....	231
15.4 Sampling .....	231
15.5 How Bad Is a ‘No Significant Difference’ Result? .....	234
15.6 Chapter Summary .....	235
15.7 Self-assessment Exercises for Chapter 15 .....	235
References .....	236

<b>16. Association Rule Mining I</b>	237
16.1 Introduction	237
16.2 Measures of Rule Interestingness	239
16.2.1 The Piatetsky-Shapiro Criteria and the RI Measure	241
16.2.2 Rule Interestingness Measures Applied to the <i>chess</i> Dataset	243
16.2.3 Using Rule Interestingness Measures for Conflict Resolution	245
16.3 Association Rule Mining Tasks	245
16.4 Finding the Best $N$ Rules	246
16.4.1 The $J$ -Measure: Measuring the Information Content of a Rule	247
16.4.2 Search Strategy	248
16.5 Chapter Summary	251
16.6 Self-assessment Exercises for Chapter 16	251
References	251
<b>17. Association Rule Mining II</b>	253
17.1 Introduction	253
17.2 Transactions and Itemsets	254
17.3 Support for an Itemset	255
17.4 Association Rules	256
17.5 Generating Association Rules	258
17.6 Apriori	259
17.7 Generating Supported Itemsets: An Example	262
17.8 Generating Rules for a Supported Itemset	264
17.9 Rule Interestingness Measures: Lift and Leverage	266
17.10 Chapter Summary	268
17.11 Self-assessment Exercises for Chapter 17	269
Reference	269
<b>18. Association Rule Mining III: Frequent Pattern Trees</b>	271
18.1 Introduction: FP-Growth	271
18.2 Constructing the FP-tree	274
18.2.1 Pre-processing the Transaction Database	274
18.2.2 Initialisation	276
18.2.3 Processing Transaction 1: $f, c, a, m, p$	277
18.2.4 Processing Transaction 2: $f, c, a, b, m$	279
18.2.5 Processing Transaction 3: $f, b$	283
18.2.6 Processing Transaction 4: $c, b, p$	285
18.2.7 Processing Transaction 5: $f, c, a, m, p$	287
18.3 Finding the Frequent Itemsets from the FP-tree	288

- 18.3.1 Itemsets Ending with Item  $p$  . . . . . 291
  - 18.3.2 Itemsets Ending with Item  $m$  . . . . . 301
- 18.4 Chapter Summary . . . . . 308
- 18.5 Self-assessment Exercises for Chapter 18 . . . . . 309
- Reference . . . . . 309
- 19. Clustering** . . . . . 311
  - 19.1 Introduction . . . . . 311
  - 19.2  $k$ -Means Clustering . . . . . 314
    - 19.2.1 Example . . . . . 315
    - 19.2.2 Finding the Best Set of Clusters . . . . . 319
  - 19.3 Agglomerative Hierarchical Clustering . . . . . 320
    - 19.3.1 Recording the Distance Between Clusters . . . . . 323
    - 19.3.2 Terminating the Clustering Process . . . . . 326
  - 19.4 Chapter Summary . . . . . 327
  - 19.5 Self-assessment Exercises for Chapter 19 . . . . . 327
- 20. Text Mining** . . . . . 329
  - 20.1 Multiple Classifications . . . . . 329
  - 20.2 Representing Text Documents for Data Mining . . . . . 330
  - 20.3 Stop Words and Stemming . . . . . 332
  - 20.4 Using Information Gain for Feature Reduction . . . . . 333
  - 20.5 Representing Text Documents: Constructing a Vector Space Model . . . . . 333
  - 20.6 Normalising the Weights . . . . . 335
  - 20.7 Measuring the Distance Between Two Vectors . . . . . 336
  - 20.8 Measuring the Performance of a Text Classifier . . . . . 337
  - 20.9 Hypertext Categorisation . . . . . 338
    - 20.9.1 Classifying Web Pages . . . . . 338
    - 20.9.2 Hypertext Classification versus Text Classification . . . 339
  - 20.10 Chapter Summary . . . . . 343
  - 20.11 Self-assessment Exercises for Chapter 20 . . . . . 343
- 21. Classifying Streaming Data** . . . . . 345
  - 21.1 Introduction . . . . . 345
    - 21.1.1 Stationary v Time-dependent Data . . . . . 347
  - 21.2 Building an H-Tree: Updating Arrays . . . . . 347
    - 21.2.1 Array *currentAtts* . . . . . 348
    - 21.2.2 Array *splitAtt* . . . . . 349
    - 21.2.3 Sorting a record to the appropriate leaf node . . . . . 349
    - 21.2.4 Array *hitcount* . . . . . 350
    - 21.2.5 Array *classtotals* . . . . . 350

21.2.6	Array <i>acvCounts</i> . . . . .	350
21.2.7	Array <i>branch</i> . . . . .	352
21.3	Building an H-Tree: a Detailed Example . . . . .	352
21.3.1	Step (a): Initialise Root Node 0 . . . . .	352
21.3.2	Step (b): Begin Reading Records . . . . .	353
21.3.3	Step (c): Consider Splitting at Node 0 . . . . .	354
21.3.4	Step (d): Split on Root Node and Initialise New Leaf Nodes . . . . .	355
21.3.5	Step (e): Process the Next Set of Records . . . . .	357
21.3.6	Step (f): Consider Splitting at Node 2 . . . . .	358
21.3.7	Step (g): Process the Next Set of Records . . . . .	359
21.3.8	Outline of the H-Tree Algorithm . . . . .	360
21.4	Splitting on an Attribute: Using Information Gain . . . . .	363
21.5	Splitting on An Attribute: Using a Hoeffding Bound . . . . .	365
21.6	H-Tree Algorithm: Final Version . . . . .	370
21.7	Using an Evolving H-Tree to Make Predictions . . . . .	372
21.7.1	Evaluating the Performance of an H-Tree . . . . .	373
21.8	Experiments: H-Tree versus TDIDT . . . . .	374
21.8.1	The lens24 Dataset . . . . .	374
21.8.2	The vote Dataset . . . . .	376
21.9	Chapter Summary . . . . .	377
21.10	Self-assessment Exercises for Chapter 21 . . . . .	377
	References . . . . .	378
<b>22.</b>	<b>Classifying Streaming Data II: Time-Dependent Data . . . . .</b>	<b>379</b>
22.1	Stationary versus Time-dependent Data . . . . .	379
22.2	Summary of the H-Tree Algorithm . . . . .	381
22.2.1	Array <i>currentAtts</i> . . . . .	382
22.2.2	Array <i>splitAtt</i> . . . . .	383
22.2.3	Array <i>hitcount</i> . . . . .	383
22.2.4	Array <i>classtotals</i> . . . . .	383
22.2.5	Array <i>acvCounts</i> . . . . .	384
22.2.6	Array <i>branch</i> . . . . .	384
22.2.7	Pseudocode for the H-Tree Algorithm . . . . .	384
22.3	From H-Tree to CDH-Tree: Overview . . . . .	387
22.4	From H-Tree to CDH-Tree: Incrementing Counts . . . . .	387
22.5	The Sliding Window Method . . . . .	388
22.6	Resplitting at Nodes . . . . .	393
22.7	Identifying Suspect Nodes . . . . .	394
22.8	Creating Alternate Nodes . . . . .	396
22.9	Growing/Forgetting an Alternate Node and its Descendants . . . . .	400

22.10	Replacing an Internal Node by One of its Alternate Nodes . . . .	402
22.11	Experiment: Tracking Concept Drift . . . . .	410
22.11.1	<i>lens24</i> Data: Alternative Mode . . . . .	412
22.11.2	Introducing Concept Drift . . . . .	414
22.11.3	An Experiment with Alternating <i>lens24</i> Data . . . . .	415
22.11.4	Comments on Experiment . . . . .	423
22.12	Chapter Summary . . . . .	424
22.13	Self-assessment Exercises for Chapter 22 . . . . .	424
	Reference . . . . .	425
<b>23.</b>	<b>An Introduction to Neural Networks . . . . .</b>	<b>427</b>
23.1	Introduction . . . . .	427
23.2	Neural Nets Example 1 . . . . .	430
23.3	Neural Nets Example 2 . . . . .	434
23.3.1	Forward Propagating the Values of the Input Nodes . . .	437
23.3.2	Forward Propagation: Summary of Formulae . . . . .	442
23.4	Backpropagation . . . . .	443
23.4.1	Stochastic Gradient Descent . . . . .	443
23.4.2	Finding the Gradients . . . . .	445
23.4.3	Working backwards from the output layer to the hid- den layer . . . . .	447
23.4.4	Working backwards from the hidden layer to the input layer . . . . .	449
23.4.5	Updating the Weights . . . . .	452
23.5	Processing a Multi-instance Training Set . . . . .	455
23.6	Using a Neural Net for Classification: the <i>iris</i> Dataset . . . . .	456
23.7	Using a Neural Net for Classification: the <i>seeds</i> Dataset . . . . .	461
23.8	Neural Nets: A Note of Caution . . . . .	463
23.9	Chapter Summary . . . . .	465
23.10	Self-assessment Exercises for Chapter 23 . . . . .	466
<b>A.</b>	<b>Essential Mathematics . . . . .</b>	<b>467</b>
A.1	Subscript Notation . . . . .	467
A.1.1	Sigma Notation for Summation . . . . .	468
A.1.2	Double Subscript Notation . . . . .	469
A.1.3	Other Uses of Subscripts . . . . .	470
A.2	Trees . . . . .	470
A.2.1	Terminology . . . . .	471
A.2.2	Interpretation . . . . .	472
A.2.3	Subtrees . . . . .	473
A.3	The Logarithm Function $\log_2 X$ . . . . .	473
A.3.1	The Function $-X \log_2 X$ . . . . .	476



- A.4 Introduction to Set Theory ..... 477
  - A.4.1 Subsets ..... 479
  - A.4.2 Summary of Set Notation ..... 481
- B. Datasets ..... 483**
  - References ..... 504
- C. Sources of Further Information ..... 505**
  - Websites ..... 505
  - Books ..... 505
  - Conferences ..... 506
  - Information About Association Rule Mining ..... 507
- D. Glossary and Notation ..... 509**
- E. Solutions to Self-assessment Exercises ..... 535**
- Index ..... 565**

# Introduction to Data Mining

## 1.1 The Data Explosion

Modern computer systems are accumulating data at an almost unimaginable rate and from a very wide variety of sources: from point-of-sale machines in the high street to machines logging every bank cash withdrawal and credit card transaction, to Earth observation satellites in space, and with an ever-growing volume of information available from social media and the Internet.

Some examples will serve to give an indication of the volumes of data involved (by the time you read this, some of the numbers will have increased considerably):

- The current NASA Earth observation satellites generate a terabyte (i.e.  $10^{12}$  bytes) of data *every day*. This is more than the total amount of data ever transmitted by all previous observation satellites.
- Biologists are generating around 15 million gigabytes of gene sequence data every year.
- Many companies maintain large Data Warehouses of customer transactions. A fairly small data warehouse might contain more than a hundred million transactions.
- There are vast amounts of data recorded every day on automatic recording devices, such as credit card transaction files and web logs, as well as non-symbolic data such as CCTV recordings.
- There are estimated to be over 1.5 billion websites, some extremely large.
- There are over 2.4 billion active users of Facebook, with an estimated 350 million photographs uploaded every day.

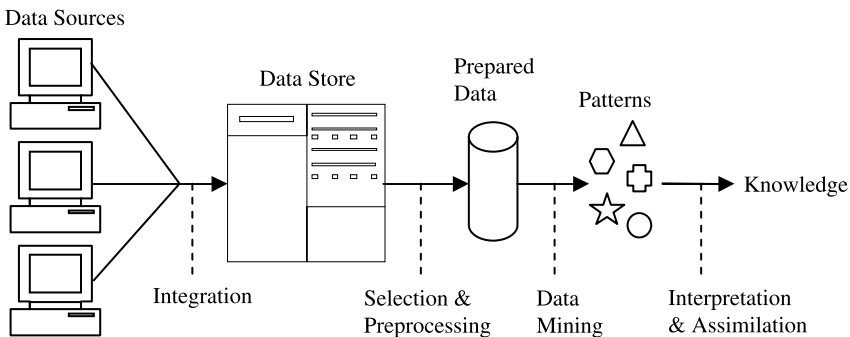
Alongside advances in storage technology, which increasingly make it possible to store such vast amounts of data at relatively low cost whether in commercial data warehouses, scientific research laboratories or elsewhere, has come a growing realisation that such data contains buried within it knowledge that can be critical to a company's growth or decline, knowledge that could lead to important discoveries in science, knowledge that could enable us accurately to predict the weather and natural disasters, knowledge that could enable us to identify the causes of and possible cures for lethal illnesses, knowledge that could literally mean the difference between life and death. Yet the huge volumes involved mean that most of this data is merely stored — never to be examined in more than the most superficial way, if at all. It has rightly been said that the world is becoming 'data rich but knowledge poor'.

As well as all the stored data, data streams of over a million records a day, potentially continuing forever, are now commonplace.

Machine learning technology, some of it very long established, has the potential to solve the problem of the tidal wave of data that is flooding around organisations, governments and individuals.

## 1.2 Knowledge Discovery

Knowledge Discovery has been defined as the 'non-trivial extraction of implicit, previously unknown and potentially useful information from data'. It is a process of which data mining forms just one part, albeit a central one.



**Figure 1.1** The Knowledge Discovery Process

Figure 1.1 shows a slightly idealised version of the complete knowledge discovery process.

Data comes in, possibly from many sources. It is integrated and placed in some common data store. Part of it is then taken and pre-processed into a standard format. This ‘prepared data’ is then passed to a data mining algorithm which produces an output in the form of rules or some other kind of ‘patterns’. These are then interpreted to give — and this is the Holy Grail for knowledge discovery — new and potentially useful knowledge.

This brief description makes it clear that although the data mining algorithms, which are the principal subject of this book, are central to knowledge discovery they are not the whole story. The pre-processing of the data and the interpretation (as opposed to the blind use) of the results are both of great importance. They are skilled tasks that are far more of an art (or a skill learnt from experience) than an exact science. Although they will both be touched on in this book, the algorithms of the data mining stage of knowledge discovery will be its prime concern.

## 1.3 Applications of Data Mining

There is a rapidly growing body of successful applications in a wide range of areas as diverse as:

- analysing satellite imagery
- analysis of organic compounds
- automatic abstracting
- bioinformatics
- credit card fraud detection
- criminal investigation
- customer relationship management
- electric load prediction
- financial forecasting
- fraud detection
- healthcare
- market basket analysis
- medical diagnosis
- predicting share of television audiences
- product design
- real estate valuation
- targeted marketing
- text summarisation
- thermal power plant optimisation
- toxic hazard analysis
- weather forecasting

and many more.

Some examples of applications (potential or actual) are:

- a supermarket chain mines its customer transactions data to optimise targeting of high value customers
- a credit card company can use its data warehouse of customer transactions for fraud detection
- a major hotel chain can use survey databases to identify attributes of a ‘high-value’ prospect
- predicting the probability of default for consumer loan applications by improving the ability to predict bad loans
- reducing fabrication flaws in VLSI chips
- data mining systems can sift through vast quantities of data collected during the semiconductor fabrication process to identify conditions that are causing yield problems
- predicting audience share for television programmes, allowing television executives to arrange show schedules to maximise market share and increase advertising revenues
- predicting the probability that a cancer patient will respond to chemotherapy, thus reducing health-care costs without affecting quality of care
- analysing motion-capture data for elderly people
- trend mining and visualisation in social networks
- analysing data from a face recognition system to locate a suspected criminal in a crowd
- analysing information about a range of drugs and natural compounds to identify significant candidates for new antibiotics
- analysing MRI images to identify possible brain tumours.

Applications can be divided into four main types: classification, numerical prediction, association and clustering. Each of these is explained briefly below. However first we need to distinguish between two types of data.

## 1.4 Labelled and Unlabelled Data

In general we have a dataset of examples (called *instances*), each of which comprises the values of a number of variables, which in data mining are often called *attributes*. There are two types of data, which are treated in radically different ways.

For the first type there is a specially designated attribute and the aim is to use the data given to predict the value of that attribute for instances that have not yet been seen. Data of this kind is called *labelled*. Data mining using labelled data is known as *supervised learning*. If the designated attribute is *categorical*, i.e. it must take one of a number of distinct values such as ‘very good’, ‘good’ or ‘poor’, or (in an object recognition application) ‘car’, ‘bicycle’, ‘person’, ‘bus’ or ‘taxi’ the task is called *classification*. If the designated attribute is numerical, e.g. the expected sale price of a house or the opening price of a share on tomorrow’s stock market, the task is called *regression*.

Data that does not have any specially designated attribute is called *unlabelled*. Data mining of unlabelled data is known as *unsupervised learning*. Here the aim is simply to extract the most information we can from the data available.

## 1.5 Supervised Learning: Classification

Classification is one of the most common applications for data mining. It corresponds to a task that occurs frequently in everyday life. For example, a hospital may want to classify medical patients into those who are at high, medium or low risk of acquiring a certain illness, an opinion polling company may wish to classify people interviewed into those who are likely to vote for each of a number of political parties or are undecided, or we may wish to classify a student project as distinction, merit, pass or fail.

This example shows a typical situation (Figure 1.2). We have a dataset in the form of a table containing students’ grades on five subjects (the values of attributes SoftEng, ARIN, HCI, CSA and Project) and their overall degree classifications. The row of dots indicates that a number of rows have been omitted in the interests of simplicity. We want to find some way of predicting the classification for other students given only their grade ‘profiles’.

There are several ways we can do this, including the following.

*Nearest Neighbour Matching.* This method relies on identifying (say) the five examples that are ‘closest’ in some sense to an unclassified one. If the five ‘nearest neighbours’ have grades Second, First, Second, Second and Second we might reasonably conclude that the new instance should be classified as ‘Second’.

*Classification Rules.* We look for rules that we can use to predict the classification of an unseen instance, for example:

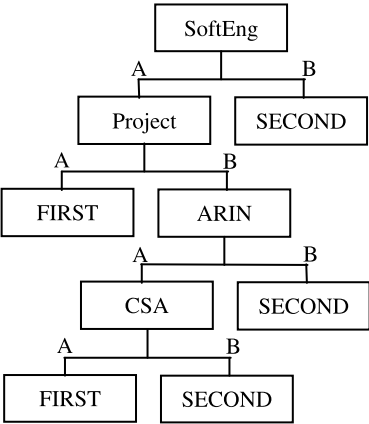
SoftEng	ARIN	HCI	CSA	Project	Class
A	B	A	B	B	Second
A	B	B	B	B	Second
B	A	A	B	A	Second
A	A	A	A	B	First
A	A	B	B	A	First
B	A	A	B	B	Second
.....	.....	.....	.....	.....	.....
A	A	B	A	B	First

**Figure 1.2** Degree Classification Data

IF SoftEng = A AND Project = A THEN Class = First  
IF SoftEng = A AND Project = B AND ARIN = B THEN Class = Second  
IF SoftEng = B THEN Class = Second

*Classification Tree.* One way of generating classification rules is via an intermediate tree-like structure called a *classification tree* or a *decision tree*.

Figure 1.3 shows a possible decision tree corresponding to the degree classification data.

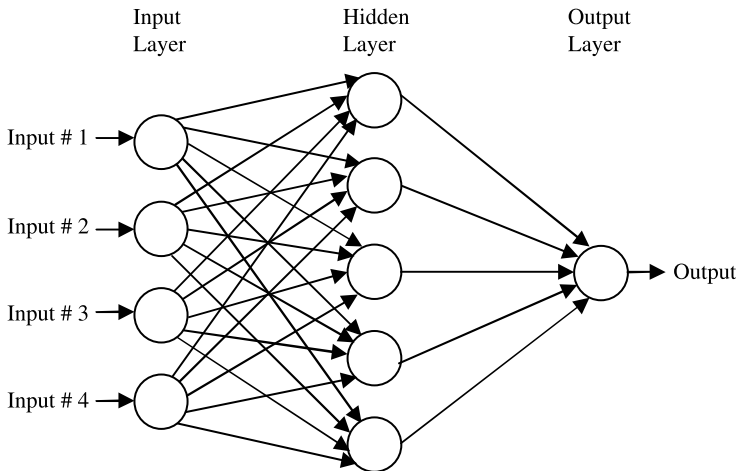


**Figure 1.3** Decision Tree for Degree Classification Data

## 1.6 Supervised Learning: Numerical Prediction

Classification is one form of prediction, where the value to be predicted is a label. Numerical prediction (often called *regression*) is another. In this case we wish to predict a numerical value, such as a company's profits or a share price.

A very popular way of doing this is to use a *Neural Network* as shown in Figure 1.4 (often called by the simplified name *Neural Net*).



**Figure 1.4** A Neural Network

This is a complex modelling technique based on a model of a human neuron. A neural net is given a set of inputs and is used to predict one or more outputs.

*One of the most widely used types of neural network is discussed in Chapter 23. However the focus is primarily on classification rather than numerical prediction.*

## 1.7 Unsupervised Learning: Association Rules

Sometimes we wish to use a training set to find any relationship that exists amongst the values of variables, generally in the form of rules known as *association rules*. There are many possible association rules derivable from any given dataset, most of them of little or no value, so it is usual for association rules to be stated with some additional information indicating how reliable they are, for example:



IF variable\_1 > 85 and switch\_6 = open  
 THEN variable\_23 < 47.5 and switch\_8 = closed (probability = 0.8)

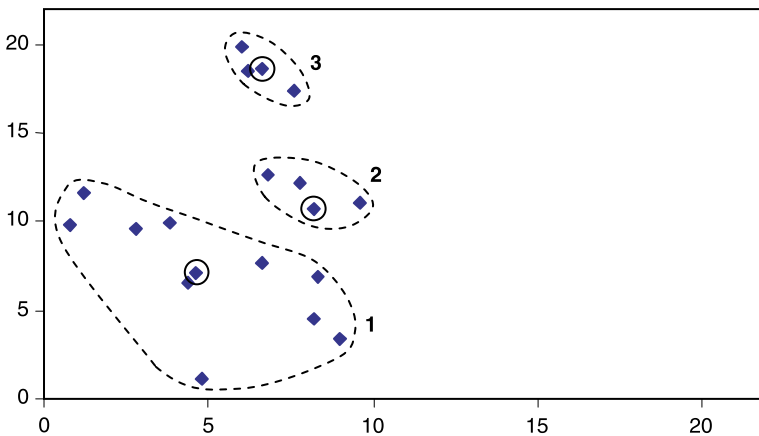
A common form of this type of application is called ‘market basket analysis’. If we know the purchases made by all the customers at a store for say a week, we may be able to find relationships that will help the store market its products more effectively in the future. For example, the rule

IF cheese AND milk THEN bread (probability = 0.7)

indicates that 70% of the customers who buy cheese and milk also buy bread, so it would be sensible to move the bread closer to the cheese and milk counter, if customer convenience were the prime concern, or to separate them to encourage impulse buying of other products if profit were more important.

## 1.8 Unsupervised Learning: Clustering

Clustering algorithms examine data to find groups of items that are similar. For example, an insurance company might group customers according to income, age, types of policy purchased or prior claims experience. In a fault diagnosis application, electrical faults might be grouped according to the values of certain key variables (Figure 1.5).



**Figure 1.5** Clustering of Data

Data for data mining comes in many forms: from computer files typed in by human operators, business information in SQL or some other standard database format, information recorded automatically by equipment such as fault logging devices, to streams of binary data transmitted from satellites. For purposes of data mining (and for the remainder of this book) we will assume that the data takes a particular standard form which is described in the next section. We will look at some of the practical problems of data preparation in Section 2.3.

## 2.1 Standard Formulation

We will assume that for any data mining application we have a *universe of objects* that are of interest. This rather grandiose term often refers to a collection of people, perhaps all human beings alive or dead, or possibly all the patients at a hospital, but may also be applied to, say, all dogs in England, or to inanimate objects such as all train journeys from London to Birmingham, all the rocks on the moon or all the pages stored in the World Wide Web.

The universe of objects is normally very large and we have only a small part of it. Usually we want to extract information from the data available to us that we hope is applicable to the large volume of data that we have not yet seen.

Each object is described by a number of *variables* that correspond to its properties. In data mining variables are often called *attributes*. We will use both terms in this book.

The set of variable values corresponding to each of the objects is called a *record* or (more commonly) an *instance*. The complete set of data available to us for an application is called a *dataset*. A dataset is often depicted as a table, with each row representing an instance. Each column contains the value of one of the variables (attributes) for each of the instances. A typical example of a dataset is the ‘degrees’ data given in the Introduction (Figure 2.1).

SoftEng	ARIN	HCI	CSA	Project	Class
A	B	A	B	B	Second
A	B	B	B	B	Second
B	A	A	B	A	Second
A	A	A	A	B	First
A	A	B	B	A	First
B	A	A	B	B	Second
.....	.....	.....	.....	.....	.....
A	A	B	A	B	First

**Figure 2.1** The Degrees Dataset

This dataset is an example of *labelled* data, where one attribute is given special significance and the aim is to predict its value. In this book we will give this attribute the standard name ‘class’. When there is no such significant attribute we call the data *unlabelled*.

## 2.2 Types of Variable

In general there are many types of variable that can be used to measure the properties of an object. A lack of understanding of the differences between the various types can lead to problems with any form of data analysis. At least six main types of variable can be distinguished.

### Nominal Variables

A variable used to put objects into categories, e.g. the name or colour of an object. A nominal variable may be numerical in form, but the numerical values have no mathematical interpretation. For example we might label 10 people as numbers 1, 2, 3, . . . , 10, but any arithmetic with such values, e.g.  $1 + 2 = 3$

would be meaningless. They are simply labels. A **classification** can be viewed as a nominal variable which has been designated as of particular importance.

## Binary Variables

A binary variable is a special case of a nominal variable that takes only two possible values: true or false, 1 or 0 etc.

## Ordinal Variables

Ordinal variables are similar to nominal variables, except that an ordinal variable has values that can be arranged in a meaningful order, e.g. small, medium, large.

## Integer Variables

Integer variables are ones that take values that are genuine integers, for example ‘number of children’. Unlike nominal variables that are numerical in form, arithmetic with integer variables is meaningful ( $1 \text{ child} + 2 \text{ children} = 3 \text{ children}$  etc.).

## Interval-scaled Variables

Interval-scaled variables are variables that take numerical values which are measured at equal intervals from a zero point or origin. However the origin does not imply a true absence of the measured characteristic. Two well-known examples of interval-scaled variables are the Fahrenheit and Celsius temperature scales. To say that one temperature measured in degrees Celsius is greater than another or greater than a constant value such as 25 is clearly meaningful, but to say that one temperature measured in degrees Celsius is twice another is meaningless. It is true that a temperature of 20 degrees is twice as far from the zero value as 10 degrees, but the zero value has been selected arbitrarily and does not imply ‘absence of temperature’. If the temperatures are converted to an equivalent scale, say degrees Fahrenheit, the ‘twice’ relationship will no longer apply.

## Ratio-scaled Variables

Ratio-scaled variables are similar to interval-scaled variables except that the zero point does reflect the absence of the measured characteristic, for example Kelvin temperature and molecular weight. In the former case the zero value corresponds to the lowest possible temperature ‘absolute zero’, so a temperature of 20 degrees Kelvin is twice one of 10 degrees Kelvin. A weight of 10 kg is twice one of 5 kg, a price of 100 dollars is twice a price of 50 dollars etc.

### 2.2.1 Categorical and Continuous Attributes

Although the distinction between different categories of variable can be important in some cases, many practical data mining systems divide attributes into just two types:

- **categorical** corresponding to nominal, binary and ordinal variables
- **continuous** corresponding to integer, interval-scaled and ratio-scaled variables.

This convention will be followed in this book. For many applications it is helpful to have a third category of attribute, the ‘ignore’ attribute, corresponding to variables that are of no significance for the application, for example the name of a patient in a hospital or the serial number of an instance, but which we do not wish to (or are unable to) delete from the dataset.

It is important to choose methods that are appropriate to the types of variable stored for a particular application. The methods described in this book are applicable to categorical and continuous attributes as defined above. There are other types of variable to which they would not be applicable without modification, for example any variable that is measured on a logarithmic scale. Two examples of logarithmic scales are the Richter scale for measuring earthquakes (an earthquake of magnitude 6 is 10 times more severe than one of magnitude 5, 100 times more severe than one of magnitude 4 etc.) and the Stellar Magnitude Scale for measuring the brightness of stars viewed by an observer on Earth.

## 2.3 Data Preparation

Although this book is about data mining not data preparation, some general comments about the latter may be helpful.

For many applications the data can simply be extracted from a database in the form described in Section 2.1, perhaps using a standard access method such as ODBC. However, for some applications the hardest task may be to get the data into a standard form in which it can be analysed. For example data values may have to be extracted from textual output generated by a fault logging system or (in a crime analysis application) extracted from transcripts of interviews with witnesses. The amount of effort required to do this may be considerable.

### 2.3.1 Data Cleaning

Even when the data is in the standard form it cannot be assumed that it is error free. In real-world datasets erroneous values can be recorded for a variety of reasons, including measurement errors, subjective judgements and malfunctioning or misuse of automatic recording equipment.

Erroneous values can be divided into those which are possible values of the attribute and those which are not. Although usage of the term *noise* varies, in this book we will take a *noisy* value to mean one that is valid for the dataset, but is incorrectly recorded. For example the number 69.72 may accidentally be entered as 6.972, or a categorical attribute value such as *brown* may accidentally be recorded as another of the possible values, such as *blue*. Noise of this kind is a perpetual problem with real-world data.

A far smaller problem arises with noisy values that are invalid for the dataset, such as 69.7X for 6.972 or *bbrown* for *brown*. We will consider these to be *invalid values*, not noise. An invalid value can easily be detected and either corrected or rejected.

It is hard to see even very ‘obvious’ errors in the values of a variable when they are ‘buried’ amongst say 100,000 other values. In attempting to ‘clean up’ data it is helpful to have a range of software tools available, especially to give an overall visual impression of the data, when some anomalous values or unexpected concentrations of values may stand out. However, in the absence of special software, even some very basic analysis of the values of variables may be helpful. Simply sorting the values into ascending order (which for fairly small datasets can be accomplished using just a standard spreadsheet) may reveal unexpected results. For example:

- A numerical variable may only take six different values, all widely separated. It would probably be best to treat this as a categorical variable rather than a continuous one.
- All the values of a variable may be identical. The variable should be treated as an ‘ignore’ attribute.