

Mining of Massive Datasets

Jure Leskovec
Stanford Univ.

Anand Rajaraman
Milliway Labs

Jeffrey D. Ullman
Stanford Univ.

Preface

This book evolved from material developed over several years by Anand Rajaraman and Jeff Ullman for a one-quarter course at Stanford. The course CS345A, titled “Web Mining,” was designed as an advanced graduate course, although it has become accessible and interesting to advanced undergraduates. When Jure Leskovec joined the Stanford faculty, we reorganized the material considerably. He introduced a new course CS224W on network analysis and added material to CS345A, which was renumbered CS246. The three authors also introduced a large-scale data-mining project course, CS341. The book now contains material taught in all three courses.

What the Book Is About

At the highest level of description, this book is about data mining. However, it focuses on data mining of very large amounts of data, that is, data so large it does not fit in main memory. Because of the emphasis on size, many of our examples are about the Web or data derived from the Web. Further, the book takes an algorithmic point of view: data mining is about applying algorithms to data, rather than using data to “train” a machine-learning engine of some sort. The principal topics covered are:

1. Distributed file systems and map-reduce as a tool for creating parallel algorithms that succeed on very large amounts of data.
2. Similarity search, including the key techniques of minhashing and locality-sensitive hashing.
3. Data-stream processing and specialized algorithms for dealing with data that arrives so fast it must be processed immediately or lost.
4. The technology of search engines, including Google’s PageRank, link-spam detection, and the hubs-and-authorities approach.
5. Frequent-itemset mining, including association rules, market-baskets, the A-Priori Algorithm and its improvements.
6. Algorithms for clustering very large, high-dimensional datasets.

7. Two key problems for Web applications: managing advertising and recommendation systems.
8. Algorithms for analyzing and mining the structure of very large graphs, especially social-network graphs.
9. Techniques for obtaining the important properties of a large dataset by dimensionality reduction, including singular-value decomposition and latent semantic indexing.
10. Machine-learning algorithms that can be applied to very large data, such as perceptrons, support-vector machines, and gradient descent.

Prerequisites

To appreciate fully the material in this book, we recommend the following prerequisites:

1. An introduction to database systems, covering SQL and related programming systems.
2. A sophomore-level course in data structures, algorithms, and discrete math.
3. A sophomore-level course in software systems, software engineering, and programming languages.

Exercises

The book contains extensive exercises, with some for almost every section. We indicate harder exercises or parts of exercises with an exclamation point. The hardest exercises have a double exclamation point.

Support on the Web

Go to <http://www.mmds.org> for slides, homework assignments, project requirements, and exams from courses related to this book.

Gradiance Automated Homework

There are automated exercises based on this book, using the Gradiance root-question technology, available at www.gradiance.com/services. Students may enter a public class by creating an account at that site and entering the class with code 1EDD8A1D. Instructors may use the site by making an account there

and then emailing `support` at `gradiance dot com` with their login name, the name of their school, and a request to use the MMDS materials.

Acknowledgements

Cover art is by Scott Ullman.

We would like to thank Foto Afrati, Arun Marathe, and Rok Sosic for critical readings of a draft of this manuscript.

Errors were also reported by Rajiv Abraham, Ruslan Aduk, Apoorv Agarwal, Aris Anagnostopoulos, Yokila Arora, Stefanie Anna Baby, Atilla Soner Balkir, Arnaud Belletoile, Robin Bennett, Susan Biancani, Amitabh Chaudhary, Leland Chen, Hua Feng, Marcus Gemeinder, Anastasios Gounaris, Clark Grubb, Shrey Gupta, Waleed Hameid, Saman Haratizadeh, Julien Hoachuck, Przemyslaw Horban, Jeff Hwang, Rafi Kamal, Lachlan Kang, Ed Knorr, Hae-woon Kwak, Ellis Lau, Greg Lee, David Z. Liu, Ethan Lozano, Yunan Luo, Michael Mahoney, Justin Meyer, Bryant Moscon, Brad Penoff, John Phillips, Philips Kokoh Prasetyo, Qi Ge, Harizo Rajaona, Timon Ruban, Rich Seiter, Hitesh Shetty, Angad Singh, Sandeep Sripada, Dennis Sidharta, Krzysztof Stencel, Mark Storus, Roshan Sumbaly, Zack Taylor, Tim Triche Jr., Wang Bin, Weng Zhen-Bin, Robert West, Steven Euijong Whang, Oscar Wu, Xie Ke, Christopher T.-R. Yeh, Nicolas Zhao, and Zhou Jingbo, The remaining errors are ours, of course.

J. L.
A. R.
J. D. U.
Palo Alto, CA
March, 2014

Contents

1	Data Mining	1
1.1	What is Data Mining?	1
1.1.1	Statistical Modeling	1
1.1.2	Machine Learning	2
1.1.3	Computational Approaches to Modeling	2
1.1.4	Summarization	3
1.1.5	Feature Extraction	4
1.2	Statistical Limits on Data Mining	4
1.2.1	Total Information Awareness	5
1.2.2	Bonferroni's Principle	5
1.2.3	An Example of Bonferroni's Principle	6
1.2.4	Exercises for Section 1.2	7
1.3	Things Useful to Know	7
1.3.1	Importance of Words in Documents	8
1.3.2	Hash Functions	9
1.3.3	Indexes	10
1.3.4	Secondary Storage	11
1.3.5	The Base of Natural Logarithms	12
1.3.6	Power Laws	13
1.3.7	Exercises for Section 1.3	15
1.4	Outline of the Book	15
1.5	Summary of Chapter 1	17
1.6	References for Chapter 1	18
2	MapReduce and the New Software Stack	21
2.1	Distributed File Systems	22
2.1.1	Physical Organization of Compute Nodes	22
2.1.2	Large-Scale File-System Organization	23
2.2	MapReduce	24
2.2.1	The Map Tasks	25
2.2.2	Grouping by Key	26
2.2.3	The Reduce Tasks	27
2.2.4	Combiners	27

2.2.5	Details of MapReduce Execution	28
2.2.6	Coping With Node Failures	29
2.2.7	Exercises for Section 2.2	30
2.3	Algorithms Using MapReduce	30
2.3.1	Matrix-Vector Multiplication by MapReduce	31
2.3.2	If the Vector v Cannot Fit in Main Memory	31
2.3.3	Relational-Algebra Operations	32
2.3.4	Computing Selections by MapReduce	35
2.3.5	Computing Projections by MapReduce	36
2.3.6	Union, Intersection, and Difference by MapReduce	36
2.3.7	Computing Natural Join by MapReduce	37
2.3.8	Grouping and Aggregation by MapReduce	37
2.3.9	Matrix Multiplication	38
2.3.10	Matrix Multiplication with One MapReduce Step	39
2.3.11	Exercises for Section 2.3	40
2.4	Extensions to MapReduce	41
2.4.1	Workflow Systems	41
2.4.2	Recursive Extensions to MapReduce	42
2.4.3	Pregel	45
2.4.4	Exercises for Section 2.4	46
2.5	The Communication Cost Model	46
2.5.1	Communication-Cost for Task Networks	47
2.5.2	Wall-Clock Time	49
2.5.3	Multiway Joins	49
2.5.4	Exercises for Section 2.5	52
2.6	Complexity Theory for MapReduce	54
2.6.1	Reducer Size and Replication Rate	54
2.6.2	An Example: Similarity Joins	55
2.6.3	A Graph Model for MapReduce Problems	57
2.6.4	Mapping Schemas	58
2.6.5	When Not All Inputs Are Present	60
2.6.6	Lower Bounds on Replication Rate	61
2.6.7	Case Study: Matrix Multiplication	62
2.6.8	Exercises for Section 2.6	66
2.7	Summary of Chapter 2	67
2.8	References for Chapter 2	69
3	Finding Similar Items	73
3.1	Applications of Near-Neighbor Search	73
3.1.1	Jaccard Similarity of Sets	74
3.1.2	Similarity of Documents	74
3.1.3	Collaborative Filtering as a Similar-Sets Problem	75
3.1.4	Exercises for Section 3.1	77
3.2	Shingling of Documents	77
3.2.1	k -Shingles	77

3.2.2	Choosing the Shingle Size	78
3.2.3	Hashing Shingles	79
3.2.4	Shingles Built from Words	79
3.2.5	Exercises for Section 3.2	80
3.3	Similarity-Preserving Summaries of Sets	80
3.3.1	Matrix Representation of Sets	81
3.3.2	Minhashing	81
3.3.3	Minhashing and Jaccard Similarity	82
3.3.4	Minhash Signatures	83
3.3.5	Computing Minhash Signatures	83
3.3.6	Exercises for Section 3.3	86
3.4	Locality-Sensitive Hashing for Documents	87
3.4.1	LSH for Minhash Signatures	88
3.4.2	Analysis of the Banding Technique	89
3.4.3	Combining the Techniques	91
3.4.4	Exercises for Section 3.4	91
3.5	Distance Measures	92
3.5.1	Definition of a Distance Measure	92
3.5.2	Euclidean Distances	93
3.5.3	Jaccard Distance	94
3.5.4	Cosine Distance	95
3.5.5	Edit Distance	95
3.5.6	Hamming Distance	96
3.5.7	Exercises for Section 3.5	97
3.6	The Theory of Locality-Sensitive Functions	99
3.6.1	Locality-Sensitive Functions	99
3.6.2	Locality-Sensitive Families for Jaccard Distance	100
3.6.3	Amplifying a Locality-Sensitive Family	101
3.6.4	Exercises for Section 3.6	103
3.7	LSH Families for Other Distance Measures	104
3.7.1	LSH Families for Hamming Distance	104
3.7.2	Random Hyperplanes and the Cosine Distance	105
3.7.3	Sketches	106
3.7.4	LSH Families for Euclidean Distance	107
3.7.5	More LSH Families for Euclidean Spaces	108
3.7.6	Exercises for Section 3.7	109
3.8	Applications of Locality-Sensitive Hashing	110
3.8.1	Entity Resolution	110
3.8.2	An Entity-Resolution Example	111
3.8.3	Validating Record Matches	112
3.8.4	Matching Fingerprints	113
3.8.5	A LSH Family for Fingerprint Matching	114
3.8.6	Similar News Articles	115
3.8.7	Exercises for Section 3.8	117
3.9	Methods for High Degrees of Similarity	118

3.9.1	Finding Identical Items	118
3.9.2	Representing Sets as Strings	118
3.9.3	Length-Based Filtering	119
3.9.4	Prefix Indexing	119
3.9.5	Using Position Information	121
3.9.6	Using Position and Length in Indexes	122
3.9.7	Exercises for Section 3.9	125
3.10	Summary of Chapter 3	126
3.11	References for Chapter 3	128
4	Mining Data Streams	131
4.1	The Stream Data Model	131
4.1.1	A Data-Stream-Management System	132
4.1.2	Examples of Stream Sources	133
4.1.3	Stream Queries	134
4.1.4	Issues in Stream Processing	135
4.2	Sampling Data in a Stream	136
4.2.1	A Motivating Example	136
4.2.2	Obtaining a Representative Sample	137
4.2.3	The General Sampling Problem	137
4.2.4	Varying the Sample Size	138
4.2.5	Exercises for Section 4.2	138
4.3	Filtering Streams	139
4.3.1	A Motivating Example	139
4.3.2	The Bloom Filter	140
4.3.3	Analysis of Bloom Filtering	140
4.3.4	Exercises for Section 4.3	141
4.4	Counting Distinct Elements in a Stream	142
4.4.1	The Count-Distinct Problem	142
4.4.2	The Flajolet-Martin Algorithm	143
4.4.3	Combining Estimates	144
4.4.4	Space Requirements	144
4.4.5	Exercises for Section 4.4	145
4.5	Estimating Moments	145
4.5.1	Definition of Moments	145
4.5.2	The Alon-Matias-Szegedy Algorithm for Second Moments	146
4.5.3	Why the Alon-Matias-Szegedy Algorithm Works	147
4.5.4	Higher-Order Moments	148
4.5.5	Dealing With Infinite Streams	148
4.5.6	Exercises for Section 4.5	149
4.6	Counting Ones in a Window	150
4.6.1	The Cost of Exact Counts	151
4.6.2	The Datar-Gionis-Indyk-Motwani Algorithm	151
4.6.3	Storage Requirements for the DGIM Algorithm	153

4.6.4	Query Answering in the DGIM Algorithm	153
4.6.5	Maintaining the DGIM Conditions	154
4.6.6	Reducing the Error	155
4.6.7	Extensions to the Counting of Ones	156
4.6.8	Exercises for Section 4.6	157
4.7	Decaying Windows	157
4.7.1	The Problem of Most-Common Elements	157
4.7.2	Definition of the Decaying Window	158
4.7.3	Finding the Most Popular Elements	159
4.8	Summary of Chapter 4	160
4.9	References for Chapter 4	161
5	Link Analysis	163
5.1	PageRank	163
5.1.1	Early Search Engines and Term Spam	164
5.1.2	Definition of PageRank	165
5.1.3	Structure of the Web	169
5.1.4	Avoiding Dead Ends	170
5.1.5	Spider Traps and Taxation	173
5.1.6	Using PageRank in a Search Engine	175
5.1.7	Exercises for Section 5.1	175
5.2	Efficient Computation of PageRank	177
5.2.1	Representing Transition Matrices	178
5.2.2	PageRank Iteration Using MapReduce	179
5.2.3	Use of Combiners to Consolidate the Result Vector	179
5.2.4	Representing Blocks of the Transition Matrix	180
5.2.5	Other Efficient Approaches to PageRank Iteration	181
5.2.6	Exercises for Section 5.2	183
5.3	Topic-Sensitive PageRank	183
5.3.1	Motivation for Topic-Sensitive Page Rank	183
5.3.2	Biased Random Walks	184
5.3.3	Using Topic-Sensitive PageRank	185
5.3.4	Inferring Topics from Words	186
5.3.5	Exercises for Section 5.3	187
5.4	Link Spam	187
5.4.1	Architecture of a Spam Farm	187
5.4.2	Analysis of a Spam Farm	189
5.4.3	Combating Link Spam	190
5.4.4	TrustRank	190
5.4.5	Spam Mass	191
5.4.6	Exercises for Section 5.4	191
5.5	Hubs and Authorities	192
5.5.1	The Intuition Behind HITS	192
5.5.2	Formalizing Hubbiness and Authority	193
5.5.3	Exercises for Section 5.5	196

5.6	Summary of Chapter 5	196
5.7	References for Chapter 5	200
6	Frequent Itemsets	201
6.1	The Market-Basket Model	202
6.1.1	Definition of Frequent Itemsets	202
6.1.2	Applications of Frequent Itemsets	204
6.1.3	Association Rules	205
6.1.4	Finding Association Rules with High Confidence	207
6.1.5	Exercises for Section 6.1	207
6.2	Market Baskets and the A-Priori Algorithm	209
6.2.1	Representation of Market-Basket Data	209
6.2.2	Use of Main Memory for Itemset Counting	210
6.2.3	Monotonicity of Itemsets	212
6.2.4	Tyranny of Counting Pairs	213
6.2.5	The A-Priori Algorithm	213
6.2.6	A-Priori for All Frequent Itemsets	214
6.2.7	Exercises for Section 6.2	217
6.3	Handling Larger Datasets in Main Memory	218
6.3.1	The Algorithm of Park, Chen, and Yu	218
6.3.2	The Multistage Algorithm	220
6.3.3	The Multihash Algorithm	222
6.3.4	Exercises for Section 6.3	224
6.4	Limited-Pass Algorithms	226
6.4.1	The Simple, Randomized Algorithm	226
6.4.2	Avoiding Errors in Sampling Algorithms	227
6.4.3	The Algorithm of Savasere, Omiecinski, and Navathe	228
6.4.4	The SON Algorithm and MapReduce	229
6.4.5	Toivonen's Algorithm	230
6.4.6	Why Toivonen's Algorithm Works	231
6.4.7	Exercises for Section 6.4	232
6.5	Counting Frequent Items in a Stream	232
6.5.1	Sampling Methods for Streams	233
6.5.2	Frequent Itemsets in Decaying Windows	234
6.5.3	Hybrid Methods	235
6.5.4	Exercises for Section 6.5	235
6.6	Summary of Chapter 6	236
6.7	References for Chapter 6	238
7	Clustering	241
7.1	Introduction to Clustering Techniques	241
7.1.1	Points, Spaces, and Distances	241
7.1.2	Clustering Strategies	243
7.1.3	The Curse of Dimensionality	244

7.1.4	Exercises for Section 7.1	245
7.2	Hierarchical Clustering	245
7.2.1	Hierarchical Clustering in a Euclidean Space	246
7.2.2	Efficiency of Hierarchical Clustering	248
7.2.3	Alternative Rules for Controlling Hierarchical Clustering	249
7.2.4	Hierarchical Clustering in Non-Euclidean Spaces	252
7.2.5	Exercises for Section 7.2	253
7.3	K-means Algorithms	254
7.3.1	K-Means Basics	255
7.3.2	Initializing Clusters for K-Means	255
7.3.3	Picking the Right Value of k	256
7.3.4	The Algorithm of Bradley, Fayyad, and Reina	257
7.3.5	Processing Data in the BFR Algorithm	259
7.3.6	Exercises for Section 7.3	262
7.4	The CURE Algorithm	262
7.4.1	Initialization in CURE	263
7.4.2	Completion of the CURE Algorithm	264
7.4.3	Exercises for Section 7.4	265
7.5	Clustering in Non-Euclidean Spaces	266
7.5.1	Representing Clusters in the GRGPF Algorithm	266
7.5.2	Initializing the Cluster Tree	267
7.5.3	Adding Points in the GRGPF Algorithm	268
7.5.4	Splitting and Merging Clusters	269
7.5.5	Exercises for Section 7.5	270
7.6	Clustering for Streams and Parallelism	270
7.6.1	The Stream-Computing Model	271
7.6.2	A Stream-Clustering Algorithm	271
7.6.3	Initializing Buckets	272
7.6.4	Merging Buckets	272
7.6.5	Answering Queries	275
7.6.6	Clustering in a Parallel Environment	275
7.6.7	Exercises for Section 7.6	276
7.7	Summary of Chapter 7	276
7.8	References for Chapter 7	280
8	Advertising on the Web	281
8.1	Issues in On-Line Advertising	281
8.1.1	Advertising Opportunities	281
8.1.2	Direct Placement of Ads	282
8.1.3	Issues for Display Ads	283
8.2	On-Line Algorithms	284
8.2.1	On-Line and Off-Line Algorithms	284
8.2.2	Greedy Algorithms	285
8.2.3	The Competitive Ratio	286

8.2.4	Exercises for Section 8.2	286
8.3	The Matching Problem	287
8.3.1	Matches and Perfect Matches	287
8.3.2	The Greedy Algorithm for Maximal Matching	288
8.3.3	Competitive Ratio for Greedy Matching	289
8.3.4	Exercises for Section 8.3	290
8.4	The Adwords Problem	290
8.4.1	History of Search Advertising	291
8.4.2	Definition of the Adwords Problem	291
8.4.3	The Greedy Approach to the Adwords Problem	292
8.4.4	The Balance Algorithm	293
8.4.5	A Lower Bound on Competitive Ratio for Balance	294
8.4.6	The Balance Algorithm with Many Bidders	296
8.4.7	The Generalized Balance Algorithm	297
8.4.8	Final Observations About the Adwords Problem	298
8.4.9	Exercises for Section 8.4	299
8.5	Adwords Implementation	299
8.5.1	Matching Bids and Search Queries	300
8.5.2	More Complex Matching Problems	300
8.5.3	A Matching Algorithm for Documents and Bids	301
8.6	Summary of Chapter 8	303
8.7	References for Chapter 8	305
9	Recommendation Systems	307
9.1	A Model for Recommendation Systems	307
9.1.1	The Utility Matrix	308
9.1.2	The Long Tail	309
9.1.3	Applications of Recommendation Systems	309
9.1.4	Populating the Utility Matrix	311
9.2	Content-Based Recommendations	312
9.2.1	Item Profiles	312
9.2.2	Discovering Features of Documents	313
9.2.3	Obtaining Item Features From Tags	314
9.2.4	Representing Item Profiles	315
9.2.5	User Profiles	316
9.2.6	Recommending Items to Users Based on Content	317
9.2.7	Classification Algorithms	318
9.2.8	Exercises for Section 9.2	320
9.3	Collaborative Filtering	321
9.3.1	Measuring Similarity	322
9.3.2	The Duality of Similarity	324
9.3.3	Clustering Users and Items	325
9.3.4	Exercises for Section 9.3	327
9.4	Dimensionality Reduction	328
9.4.1	UV-Decomposition	328

9.4.2	Root-Mean-Square Error	329
9.4.3	Incremental Computation of a UV-Decomposition	330
9.4.4	Optimizing an Arbitrary Element	332
9.4.5	Building a Complete UV-Decomposition Algorithm	334
9.4.6	Exercises for Section 9.4	336
9.5	The Netflix Challenge	337
9.6	Summary of Chapter 9	338
9.7	References for Chapter 9	340
10	Mining Social-Network Graphs	343
10.1	Social Networks as Graphs	343
10.1.1	What is a Social Network?	344
10.1.2	Social Networks as Graphs	344
10.1.3	Varieties of Social Networks	346
10.1.4	Graphs With Several Node Types	347
10.1.5	Exercises for Section 10.1	348
10.2	Clustering of Social-Network Graphs	349
10.2.1	Distance Measures for Social-Network Graphs	349
10.2.2	Applying Standard Clustering Methods	349
10.2.3	Betweenness	351
10.2.4	The Girvan-Newman Algorithm	351
10.2.5	Using Betweenness to Find Communities	354
10.2.6	Exercises for Section 10.2	356
10.3	Direct Discovery of Communities	357
10.3.1	Finding Cliques	357
10.3.2	Complete Bipartite Graphs	357
10.3.3	Finding Complete Bipartite Subgraphs	358
10.3.4	Why Complete Bipartite Graphs Must Exist	359
10.3.5	Exercises for Section 10.3	361
10.4	Partitioning of Graphs	361
10.4.1	What Makes a Good Partition?	362
10.4.2	Normalized Cuts	362
10.4.3	Some Matrices That Describe Graphs	363
10.4.4	Eigenvalues of the Laplacian Matrix	364
10.4.5	Alternative Partitioning Methods	367
10.4.6	Exercises for Section 10.4	368
10.5	Finding Overlapping Communities	369
10.5.1	The Nature of Communities	369
10.5.2	Maximum-Likelihood Estimation	369
10.5.3	The Affiliation-Graph Model	371
10.5.4	Avoiding the Use of Discrete Membership Changes	374
10.5.5	Exercises for Section 10.5	375
10.6	Simrank	376
10.6.1	Random Walkers on a Social Graph	376
10.6.2	Random Walks with Restart	377

10.6.3 Exercises for Section 10.6	380
10.7 Counting Triangles	380
10.7.1 Why Count Triangles?	380
10.7.2 An Algorithm for Finding Triangles	381
10.7.3 Optimality of the Triangle-Finding Algorithm	382
10.7.4 Finding Triangles Using MapReduce	383
10.7.5 Using Fewer Reduce Tasks	384
10.7.6 Exercises for Section 10.7	385
10.8 Neighborhood Properties of Graphs	386
10.8.1 Directed Graphs and Neighborhoods	386
10.8.2 The Diameter of a Graph	388
10.8.3 Transitive Closure and Reachability	389
10.8.4 Transitive Closure Via MapReduce	390
10.8.5 Smart Transitive Closure	392
10.8.6 Transitive Closure by Graph Reduction	393
10.8.7 Approximating the Sizes of Neighborhoods	395
10.8.8 Exercises for Section 10.8	397
10.9 Summary of Chapter 10	398
10.10References for Chapter 10	402
11 Dimensionality Reduction	405
11.1 Eigenvalues and Eigenvectors of Symmetric Matrices	406
11.1.1 Definitions	406
11.1.2 Computing Eigenvalues and Eigenvectors	407
11.1.3 Finding Eigenpairs by Power Iteration	408
11.1.4 The Matrix of Eigenvectors	411
11.1.5 Exercises for Section 11.1	411
11.2 Principal-Component Analysis	412
11.2.1 An Illustrative Example	413
11.2.2 Using Eigenvectors for Dimensionality Reduction	416
11.2.3 The Matrix of Distances	417
11.2.4 Exercises for Section 11.2	418
11.3 Singular-Value Decomposition	418
11.3.1 Definition of SVD	418
11.3.2 Interpretation of SVD	420
11.3.3 Dimensionality Reduction Using SVD	422
11.3.4 Why Zeroing Low Singular Values Works	423
11.3.5 Querying Using Concepts	425
11.3.6 Computing the SVD of a Matrix	426
11.3.7 Exercises for Section 11.3	427
11.4 CUR Decomposition	428
11.4.1 Definition of CUR	429
11.4.2 Choosing Rows and Columns Properly	430
11.4.3 Constructing the Middle Matrix	431
11.4.4 The Complete CUR Decomposition	432

11.4.5	Eliminating Duplicate Rows and Columns	433
11.4.6	Exercises for Section 11.4	434
11.5	Summary of Chapter 11	434
11.6	References for Chapter 11	436
12	Large-Scale Machine Learning	439
12.1	The Machine-Learning Model	440
12.1.1	Training Sets	440
12.1.2	Some Illustrative Examples	440
12.1.3	Approaches to Machine Learning	443
12.1.4	Machine-Learning Architecture	444
12.1.5	Exercises for Section 12.1	447
12.2	Perceptrons	447
12.2.1	Training a Perceptron with Zero Threshold	447
12.2.2	Convergence of Perceptrons	451
12.2.3	The Winnow Algorithm	451
12.2.4	Allowing the Threshold to Vary	453
12.2.5	Multiclass Perceptrons	455
12.2.6	Transforming the Training Set	456
12.2.7	Problems With Perceptrons	457
12.2.8	Parallel Implementation of Perceptrons	458
12.2.9	Exercises for Section 12.2	459
12.3	Support-Vector Machines	461
12.3.1	The Mechanics of an SVM	461
12.3.2	Normalizing the Hyperplane	462
12.3.3	Finding Optimal Approximate Separators	464
12.3.4	SVM Solutions by Gradient Descent	467
12.3.5	Stochastic Gradient Descent	470
12.3.6	Parallel Implementation of SVM	471
12.3.7	Exercises for Section 12.3	472
12.4	Learning from Nearest Neighbors	472
12.4.1	The Framework for Nearest-Neighbor Calculations	473
12.4.2	Learning with One Nearest Neighbor	473
12.4.3	Learning One-Dimensional Functions	474
12.4.4	Kernel Regression	477
12.4.5	Dealing with High-Dimensional Euclidean Data	477
12.4.6	Dealing with Non-Euclidean Distances	479
12.4.7	Exercises for Section 12.4	479
12.5	Comparison of Learning Methods	480
12.6	Summary of Chapter 12	481
12.7	References for Chapter 12	483