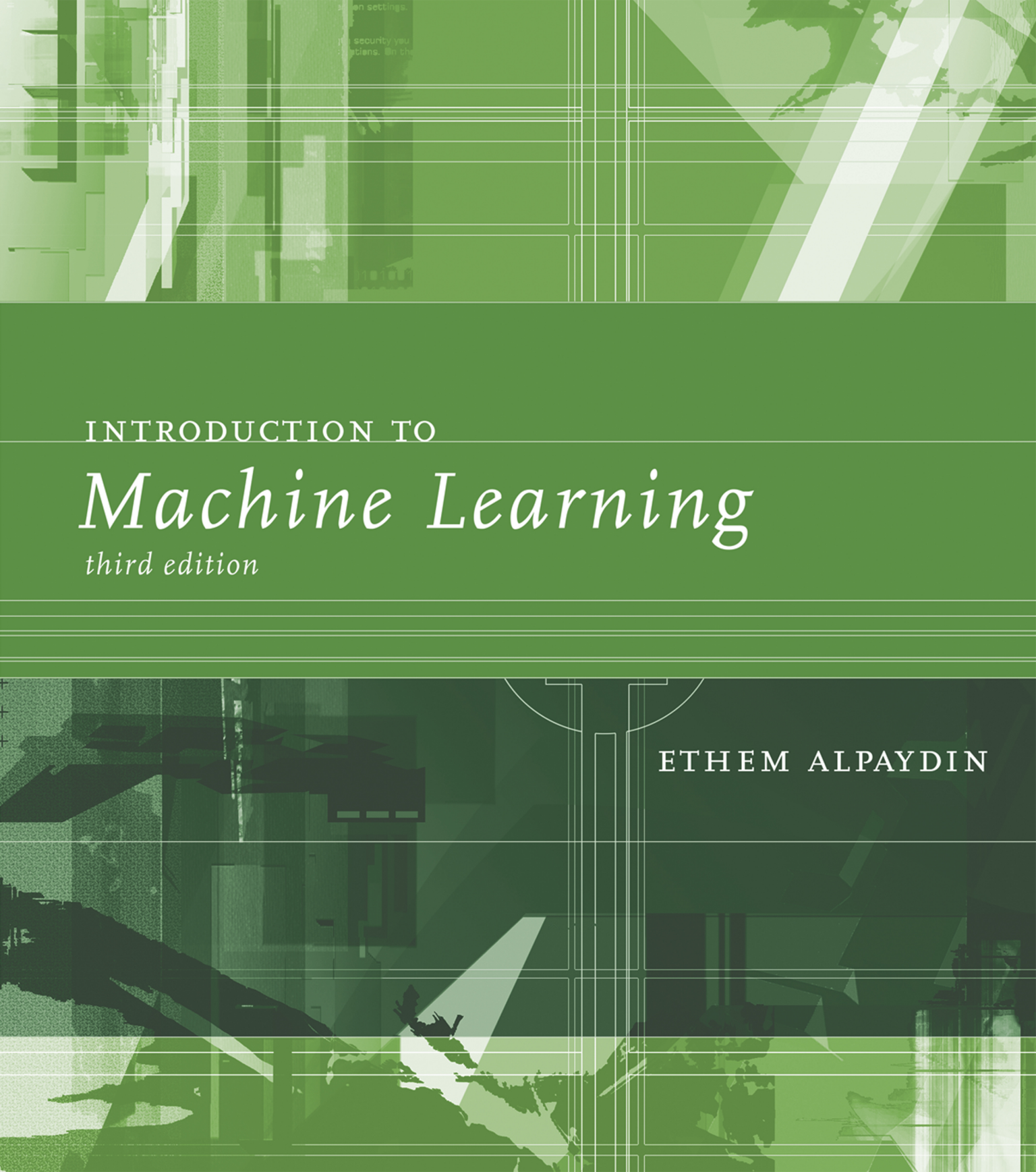

INTRODUCTION TO

Machine Learning

third edition

ETHEM ALPAYDIN

Introduction to Machine Learning

Third Edition

Adaptive Computation and Machine Learning

Thomas Dietterich, Editor
Christopher Bishop, David Heckerman, Michael Jordan, and Michael Kearns, Associate Editors

A complete list of books published in The Adaptive Computation and Machine Learning series appears at the back of this book.

Introduction to Machine Learning

Third Edition

Ethem Alpaydın

The MIT Press
Cambridge, Massachusetts
London, England

Typeset in 10/13 Lucida Bright by the author using LaTeX 2$_\varepsilon$.
Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Information

Alpaydin, Ethem.
Introduction to machine learning / Ethem Alpaydin—3rd ed.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-262-02818-9 (hardcover : alk. paper)
1. Machine learning. I. Title
Q325.5.A46 2014
006.3'1—dc23 2014007214
CIP

10 9 8 7 6 5 4 3 2 1

5.11 References 113

6 *Dimensionality Reduction* 115

6.1 Introduction 115
6.2 Subset Selection 116
6.3 Principal Component Analysis 120
6.4 Feature Embedding 127
6.5 Factor Analysis 130
6.6 Singular Value Decomposition and Matrix Factorization 135
6.7 Multidimensional Scaling 136
6.8 Linear Discriminant Analysis 140
6.9 Canonical Correlation Analysis 145
6.10 Isomap 148
6.11 Locally Linear Embedding 150
6.12 Laplacian Eigenmaps 153
6.13 Notes 155
6.14 Exercises 157
6.15 References 158

7 *Clustering* 161

7.1 Introduction 161
7.2 Mixture Densities 162
7.3 *k*-Means Clustering 163
7.4 Expectation-Maximization Algorithm 167
7.5 Mixtures of Latent Variable Models 172
7.6 Supervised Learning after Clustering 173
7.7 Spectral Clustering 175
7.8 Hierarchical Clustering 176
7.9 Choosing the Number of Clusters 178
7.10 Notes 179
7.11 Exercises 180
7.12 References 182

8 *Nonparametric Methods* 185

8.1 Introduction 185
8.2 Nonparametric Density Estimation 186
8.2.1 Histogram Estimator 187
8.2.2 Kernel Estimator 188
8.2.3 *k*-Nearest Neighbor Estimator 190
8.3 Generalization to Multivariate Data 192

Brief Contents

1 Introduction 1
2 Supervised Learning 21
3 Bayesian Decision Theory 49
4 Parametric Methods 65
5 Multivariate Methods 93
6 Dimensionality Reduction 115
7 Clustering 161
8 Nonparametric Methods 185
9 Decision Trees 213
10 Linear Discrimination 239
11 Multilayer Perceptrons 267
12 Local Models 317
13 Kernel Machines 349
14 Graphical Models 387
15 Hidden Markov Models 417
16 Bayesian Estimation 445
17 Combining Multiple Learners 487
18 Reinforcement Learning 517
19 Design and Analysis of Machine Learning Experiments 547
A Probability 593

Contents

Preface **xvii**

Notations **xxi**

1 ***Introduction*** **1**

1.1 What Is Machine Learning? 1
1.2 Examples of Machine Learning Applications 4
1.2.1 Learning Associations 4
1.2.2 Classification 5
1.2.3 Regression 9
1.2.4 Unsupervised Learning 11
1.2.5 Reinforcement Learning 13
1.3 Notes 14
1.4 Relevant Resources 17
1.5 Exercises 18
1.6 References 20

2 ***Supervised Learning*** **21**

2.1 Learning a Class from Examples 21
2.2 Vapnik-Chervonenkis Dimension 27
2.3 Probably Approximately Correct Learning 29
2.4 Noise 30
2.5 Learning Multiple Classes 32
2.6 Regression 34
2.7 Model Selection and Generalization 37
2.8 Dimensions of a Supervised Machine Learning Algorithm 41
2.9 Notes 42

8.4 Nonparametric Classification 193
8.5 Condensed Nearest Neighbor 194
8.6 Distance-Based Classification 196
8.7 Outlier Detection 199
8.8 Nonparametric Regression: Smoothing Models 201
8.8.1 Running Mean Smoother 201
8.8.2 Kernel Smoother 203
8.8.3 Running Line Smoother 204
8.9 How to Choose the Smoothing Parameter 204
8.10 Notes 205
8.11 Exercises 208
8.12 References 210

9 *Decision Trees* 213

9.1 Introduction 213
9.2 Univariate Trees 215
9.2.1 Classification Trees 216
9.2.2 Regression Trees 220
9.3 Pruning 222
9.4 Rule Extraction from Trees 225
9.5 Learning Rules from Data 226
9.6 Multivariate Trees 230
9.7 Notes 232
9.8 Exercises 235
9.9 References 237

10 *Linear Discrimination* 239

10.1 Introduction 239
10.2 Generalizing the Linear Model 241
10.3 Geometry of the Linear Discriminant 242
10.3.1 Two Classes 242
10.3.2 Multiple Classes 244
10.4 Pairwise Separation 246
10.5 Parametric Discrimination Revisited 247
10.6 Gradient Descent 248
10.7 Logistic Discrimination 250
10.7.1 Two Classes 250
10.7.2 Multiple Classes 254
10.8 Discrimination by Regression 257

10.9 Learning to Rank 260
10.10 Notes 263
10.11 Exercises 263
10.12 References 266

11 *Multilayer Perceptrons* 267

11.1 Introduction 267
11.1.1 Understanding the Brain 268
11.1.2 Neural Networks as a Paradigm for Parallel Processing 269
11.2 The Perceptron 271
11.3 Training a Perceptron 274
11.4 Learning Boolean Functions 277
11.5 Multilayer Perceptrons 279
11.6 MLP as a Universal Approximator 281
11.7 Backpropagation Algorithm 283
11.7.1 Nonlinear Regression 284
11.7.2 Two-Class Discrimination 286
11.7.3 Multiclass Discrimination 288
11.7.4 Multiple Hidden Layers 290
11.8 Training Procedures 290
11.8.1 Improving Convergence 290
11.8.2 Overtraining 291
11.8.3 Structuring the Network 292
11.8.4 Hints 295
11.9 Tuning the Network Size 297
11.10 Bayesian View of Learning 300
11.11 Dimensionality Reduction 301
11.12 Learning Time 304
11.12.1 Time Delay Neural Networks 304
11.12.2 Recurrent Networks 305
11.13 Deep Learning 306
11.14 Notes 309
11.15 Exercises 311
11.16 References 313

12 *Local Models* 317

12.1 Introduction 317
12.2 Competitive Learning 318

12.2.1 Online k-Means 318
12.2.2 Adaptive Resonance Theory 323
12.2.3 Self-Organizing Maps 324
12.3 Radial Basis Functions 326
12.4 Incorporating Rule-Based Knowledge 332
12.5 Normalized Basis Functions 333
12.6 Competitive Basis Functions 335
12.7 Learning Vector Quantization 338
12.8 The Mixture of Experts 338
12.8.1 Cooperative Experts 341
12.8.2 Competitive Experts 342
12.9 Hierarchical Mixture of Experts 342
12.10 Notes 343
12.11 Exercises 344
12.12 References 347

13 *Kernel Machines* 349

13.1 Introduction 349
13.2 Optimal Separating Hyperplane 351
13.3 The Nonseparable Case: Soft Margin Hyperplane 355
13.4 ν-SVM 358
13.5 Kernel Trick 359
13.6 Vectorial Kernels 361
13.7 Defining Kernels 364
13.8 Multiple Kernel Learning 365
13.9 Multiclass Kernel Machines 367
13.10 Kernel Machines for Regression 368
13.11 Kernel Machines for Ranking 373
13.12 One-Class Kernel Machines 374
13.13 Large Margin Nearest Neighbor Classifier 377
13.14 Kernel Dimensionality Reduction 379
13.15 Notes 380
13.16 Exercises 382
13.17 References 383

14 *Graphical Models* 387

14.1 Introduction 387
14.2 Canonical Cases for Conditional Independence 389
14.3 Generative Models 396

14.4 d-Separation 399
14.5 Belief Propagation 399
14.5.1 Chains 400
14.5.2 Trees 402
14.5.3 Polytrees 404
14.5.4 Junction Trees 406
14.6 Undirected Graphs: Markov Random Fields 407
14.7 Learning the Structure of a Graphical Model 410
14.8 Influence Diagrams 411
14.9 Notes 412
14.10 Exercises 413
14.11 References 415

15 *Hidden Markov Models* 417

15.1 Introduction 417
15.2 Discrete Markov Processes 418
15.3 Hidden Markov Models 421
15.4 Three Basic Problems of HMMs 423
15.5 Evaluation Problem 423
15.6 Finding the State Sequence 427
15.7 Learning Model Parameters 429
15.8 Continuous Observations 432
15.9 The HMM as a Graphical Model 433
15.10 Model Selection in HMMs 436
15.11 Notes 438
15.12 Exercises 440
15.13 References 443

16 *Bayesian Estimation* 445

16.1 Introduction 445
16.2 Bayesian Estimation of the Parameters of a Discrete Distribution 449
16.2.1 $K > 2$ States: Dirichlet Distribution 449
16.2.2 $K = 2$ States: Beta Distribution 450
16.3 Bayesian Estimation of the Parameters of a Gaussian Distribution 451
16.3.1 Univariate Case: Unknown Mean, Known Variance 451

16.3.2 Univariate Case: Unknown Mean, Unknown Variance 453
16.3.3 Multivariate Case: Unknown Mean, Unknown Covariance 455
16.4 Bayesian Estimation of the Parameters of a Function 456
16.4.1 Regression 456
16.4.2 Regression with Prior on Noise Precision 460
16.4.3 The Use of Basis/Kernel Functions 461
16.4.4 Bayesian Classification 463
16.5 Choosing a Prior 466
16.6 Bayesian Model Comparison 467
16.7 Bayesian Estimation of a Mixture Model 470
16.8 Nonparametric Bayesian Modeling 473
16.9 Gaussian Processes 474
16.10 Dirichlet Processes and Chinese Restaurants 478
16.11 Latent Dirichlet Allocation 480
16.12 Beta Processes and Indian Buffets 482
16.13 Notes 483
16.14 Exercises 484
16.15 References 485

17 *Combining Multiple Learners* 487

17.1 Rationale 487
17.2 Generating Diverse Learners 488
17.3 Model Combination Schemes 491
17.4 Voting 492
17.5 Error-Correcting Output Codes 496
17.6 Bagging 498
17.7 Boosting 499
17.8 The Mixture of Experts Revisited 502
17.9 Stacked Generalization 504
17.10 Fine-Tuning an Ensemble 505
17.10.1 Choosing a Subset of the Ensemble 506
17.10.2 Constructing Metalearners 506
17.11 Cascading 507
17.12 Notes 509
17.13 Exercises 511
17.14 References 513

who send me words of appreciation, criticism, or errata, or who provide feedback in any other way. Please keep them coming. My email address is `alpaydin@boun.edu.tr`. The book's web site is `http://www.cmpe.boun.edu.tr/~ethem/i2ml3e`.

It has been a pleasure to work with the MIT Press again on this third edition, and I thank Marie Lufkin Lee, Marc Lowenthal, and Kathleen Caruso for all their help and support.

18 *Reinforcement Learning* 517

18.1 Introduction 517
18.2 Single State Case: K-Armed Bandit 519
18.3 Elements of Reinforcement Learning 520
18.4 Model-Based Learning 523
18.4.1 Value Iteration 523
18.4.2 Policy Iteration 524
18.5 Temporal Difference Learning 525
18.5.1 Exploration Strategies 525
18.5.2 Deterministic Rewards and Actions 526
18.5.3 Nondeterministic Rewards and Actions 527
18.5.4 Eligibility Traces 530
18.6 Generalization 531
18.7 Partially Observable States 534
18.7.1 The Setting 534
18.7.2 Example: The Tiger Problem 536
18.8 Notes 541
18.9 Exercises 542
18.10 References 544

19 *Design and Analysis of Machine Learning Experiments* 547

19.1 Introduction 547
19.2 Factors, Response, and Strategy of Experimentation 550
19.3 Response Surface Design 553
19.4 Randomization, Replication, and Blocking 554
19.5 Guidelines for Machine Learning Experiments 555
19.6 Cross-Validation and Resampling Methods 558
19.6.1 K-Fold Cross-Validation 559
19.6.2 5×2 Cross-Validation 560
19.6.3 Bootstrapping 561
19.7 Measuring Classifier Performance 561
19.8 Interval Estimation 564
19.9 Hypothesis Testing 568
19.10 Assessing a Classification Algorithm's Performance 570
19.10.1 Binomial Test 571
19.10.2 Approximate Normal Test 572
19.10.3 t Test 572
19.11 Comparing Two Classification Algorithms 573
19.11.1 McNemar's Test 573

19.11.2 K-Fold Cross-Validated Paired t Test 573
19.11.3 5×2 cv Paired t Test 574
19.11.4 5×2 cv Paired F Test 575
19.12 Comparing Multiple Algorithms: Analysis of Variance 576
19.13 Comparison over Multiple Datasets 580
19.13.1 Comparing Two Algorithms 581
19.13.2 Multiple Algorithms 583
19.14 Multivariate Tests 584
19.14.1 Comparing Two Algorithms 585
19.14.2 Comparing Multiple Algorithms 586
19.15 Notes 587
19.16 Exercises 588
19.17 References 590

A *Probability* 593

A.1 Elements of Probability 593
A.1.1 Axioms of Probability 594
A.1.2 Conditional Probability 594
A.2 Random Variables 595
A.2.1 Probability Distribution and Density Functions 595
A.2.2 Joint Distribution and Density Functions 596
A.2.3 Conditional Distributions 596
A.2.4 Bayes' Rule 597
A.2.5 Expectation 597
A.2.6 Variance 598
A.2.7 Weak Law of Large Numbers 599
A.3 Special Random Variables 599
A.3.1 Bernoulli Distribution 599
A.3.2 Binomial Distribution 600
A.3.3 Multinomial Distribution 600
A.3.4 Uniform Distribution 600
A.3.5 Normal (Gaussian) Distribution 601
A.3.6 Chi-Square Distribution 602
A.3.7 t Distribution 603
A.3.8 F Distribution 603
A.4 References 603

***Index* 605**

Preface

Machine learning must be one of the fastest growing fields in computer science. It is not only that the data is continuously getting "bigger," but also the theory to process it and turn it into knowledge. In various fields of science, from astronomy to biology, but also in everyday life, as digital technology increasingly infiltrates our daily existence, as our digital footprint deepens, more data is continuously generated and collected. Whether scientific or personal, data that just lies dormant passively is not of any use, and smart people have been finding ever new ways to make use of that data and turn it into a useful product or service. In this transformation, machine learning plays a larger and larger role.

This data evolution has been continuing even stronger since the second edition appeared in 2010. Every year, datasets are getting larger. Not only has the number of observations grown, but the number of observed attributes has also increased significantly. There is more structure to the data: It is not just numbers and character strings any more but images, video, audio, documents, web pages, click logs, graphs, and so on. More and more, the data moves away from the parametric assumptions we used to make—for example, normality. Frequently, the data is dynamic and so there is a time dimension. Sometimes, our observations are multi-view—for the same object or event, we have multiple sources of information from different sensors and modalities.

Our belief is that behind all this seemingly complex and voluminous data, there lies a simple explanation. That although the data is big, it can be explained in terms of a relatively simple model with a small number of hidden factors and their interaction. Think about millions of customers who each day buy thousands of products online or from their local supermarket. This implies a very large database of transactions, but there is a

pattern to this data. People do not shop at random. A person throwing a party buys a certain subset of products, and a person who has a baby at home buys a different subset; there are hidden factors that explain customer behavior.

This is one of the areas where significant research has been done in recent years—namely, to infer this hidden model from observed data. Most of the revisions in this new edition are related to these advances. Chapter 6 contains new sections on feature embedding, singular value decomposition and matrix factorization, canonical correlation analysis, and Laplacian eigenmaps.

There are new sections on distance estimation in chapter 8 and on kernel machines in chapter 13: Dimensionality reduction, feature extraction, and distance estimation are three names for the same devil—the ideal distance measure is defined in the space of the ideal hidden features, and they are fewer in number than the values we observe.

Chapter 16 is rewritten and significantly extended to cover such generative models. We discuss the Bayesian approach for all major machine learning models, namely, classification, regression, mixture models, and dimensionality reduction. Nonparametric Bayesian modeling, which has become increasingly popular during these last few years, is especially interesting because it allows us to adjust the complexity of the model to the complexity of data.

New sections have been added here and there, mostly to highlight different recent applications of the same or very similar methods. There is a new section on outlier detection in chapter 8. Two new sections in chapters 10 and 13 discuss ranking for linear models and kernel machines, respectively. Having added Laplacian eigenmaps to chapter 6, I also include a new section on spectral clustering in chapter 7. Given the recent resurgence of deep neural networks, it became necessary to include a new section on deep learning in chapter 11. Chapter 19 contains a new section on multivariate tests for comparison of methods.

Since the first edition, I have received many requests for the solutions to exercises from readers who use the book for self-study. In this new edition, I have included the solutions to some of the more didactic exercises. Sometimes they are complete solutions, and sometimes they give just a hint or offer only one of several possible solutions.

I would like to thank all the instructors and students who have used the previous two editions, as well as their translations into German, Chinese, and Turkish, and their reprints in India. I am always grateful to those

Notations

x	Scalar value
$\boldsymbol{x}$	Vector
$\mathbf{X}$	Matrix
$\boldsymbol{x}^T$	Transpose
$\mathbf{X}^{-1}$	Inverse
X	Random variable
$P(X)$	Probability mass function when X is discrete
$p(X)$	Probability density function when X is continuous
$P(X\|Y)$	Conditional probability of X given Y
$E[X]$	Expected value of the random variable X
$\text{Var}(X)$	Variance of X
$\text{Cov}(X, Y)$	Covariance of X and Y
$\text{Corr}(X, Y)$	Correlation of X and Y
μ	Mean
σ^2	Variance
$\boldsymbol{\Sigma}$	Covariance matrix
m	Estimator to the mean
s^2	Estimator to the variance
$\mathbf{S}$	Estimator to the covariance matrix

$\mathcal{N}(\mu, \sigma^2)$	Univariate normal distribution with mean μ and variance σ^2
$\mathcal{Z}$	Unit normal distribution: $\mathcal{N}(0, 1)$
$\mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	d-variate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
x	Input
d	Number of inputs (input dimensionality)
y	Output
r	Required output
K	Number of outputs (classes)
N	Number of training instances
z	Hidden value, intrinsic dimension, latent factor
k	Number of hidden dimensions, latent factors
C_i	Class i
$\mathcal{X}$	Training sample
$\{x^t\}_{t=1}^N$	Set of x with index t ranging from 1 to N
$\{x^t, r^t\}_t$	Set of ordered pairs of input and desired output with index t
$g(x\|\theta)$	Function of x defined up to a set of parameters θ
$\arg\max_\theta g(x\|\theta)$	The argument θ for which g has its maximum value
$\arg\min_\theta g(x\|\theta)$	The argument θ for which g has its minimum value
$E(\theta\|\mathcal{X})$	Error function with parameters θ on the sample $\mathcal{X}$
$l(\theta\|\mathcal{X})$	Likelihood of parameters θ on the sample $\mathcal{X}$
$\mathcal{L}(\theta\|\mathcal{X})$	Log likelihood of parameters θ on the sample $\mathcal{X}$
$1(c)$	1 if c is true, 0 otherwise
$\#\{c\}$	Number of elements for which c is true
δ_{ij}	Kronecker delta: 1 if $i = j$, 0 otherwise

1 *Introduction*

1.1 What Is Machine Learning?

THIS IS the age of "big data." Once upon a time, only companies had data. There used to be computer centers where that data was stored and processed. First with the arrival of personal computers and later with the widespread use of wireless communications, we all became producers of data. Every time we buy a product, every time we rent a movie, visit a web page, write a blog, or post on the social media, even when we just walk or drive around, we are generating data.

Each of us is not only a generator but also a consumer of data. We want to have products and services specialized for us. We want our needs to be understood and interests to be predicted.

Think, for example, of a supermarket chain that is selling thousands of goods to millions of customers either at hundreds of brick-and-mortar stores all over a country or through a virtual store over the web. The details of each transaction are stored: date, customer id, goods bought and their amount, total money spent, and so forth. This typically amounts to a lot of data every day. What the supermarket chain wants is to be able to predict which customer is likely to buy which product, to maximize sales and profit. Similarly each customer wants to find the set of products best matching his/her needs.

This task is not evident. We do not know exactly which people are likely to buy this ice cream flavor or the next book of this author, see this new movie, visit this city, or click this link. Customer behavior changes in time and by geographic location. But we know that it is not completely random. People do not go to supermarkets and buy things at random. When they buy beer, they buy chips; they buy ice cream in summer and

spices for Glühwein in winter. There are certain patterns in the data.

To solve a problem on a computer, we need an algorithm. An algorithm is a sequence of instructions that should be carried out to transform the input to output. For example, one can devise an algorithm for sorting. The input is a set of numbers and the output is their ordered list. For the same task, there may be various algorithms and we may be interested in finding the most efficient one, requiring the least number of instructions or memory or both.

For some tasks, however, we do not have an algorithm. Predicting customer behavior is one; another is to tell spam emails from legitimate ones. We know what the input is: an email document that in the simplest case is a file of characters. We know what the output should be: a yes/no output indicating whether the message is spam or not. But we do not know how to transform the input to the output. What is considered spam changes in time and from individual to individual.

What we lack in knowledge, we make up for in data. We can easily compile thousands of example messages, some of which we know to be spam and some of which are not, and what we want is to "learn" what constitutes spam from them. In other words, we would like the computer (machine) to extract automatically the algorithm for this task. There is no need to learn to sort numbers since we already have algorithms for that, but there are many applications for which we do not have an algorithm but have lots of data.

We may not be able to identify the process completely, but we believe we can construct *a good and useful approximation*. That approximation may not explain everything, but may still be able to account for some part of the data. We believe that though identifying the complete process may not be possible, we can still detect certain patterns or regularities. This is the niche of machine learning. Such patterns may help us understand the process, or we can use those patterns to make predictions: Assuming that the future, at least the near future, will not be much different from the past when the sample data was collected, the future predictions can also be expected to be right.

Application of machine learning methods to large databases is called *data mining*. The analogy is that a large volume of earth and raw material is extracted from a mine, which when processed leads to a small amount of very precious material; similarly, in data mining, a large volume of data is processed to construct a simple model with valuable use, for example, having high predictive accuracy. Its application areas are

abundant: In addition to retail, in finance banks analyze their past data to build models to use in credit applications, fraud detection, and the stock market. In manufacturing, learning models are used for optimization, control, and troubleshooting. In medicine, learning programs are used for medical diagnosis. In telecommunications, call patterns are analyzed for network optimization and maximizing the quality of service. In science, large amounts of data in physics, astronomy, and biology can only be analyzed fast enough by computers. The World Wide Web is huge; it is constantly growing, and searching for relevant information cannot be done manually.

But machine learning is not just a database problem; it is also a part of artificial intelligence. To be intelligent, a system that is in a changing environment should have the ability to learn. If the system can learn and adapt to such changes, the system designer need not foresee and provide solutions for all possible situations.

Machine learning also helps us find solutions to many problems in vision, speech recognition, and robotics. Let us take the example of recognizing faces: This is a task we do effortlessly; every day we recognize family members and friends by looking at their faces or from their photographs, despite differences in pose, lighting, hair style, and so forth. But we do it unconsciously and are unable to explain how we do it. Because we are not able to explain our expertise, we cannot write the computer program. At the same time, we know that a face image is not just a random collection of pixels; a face has structure. It is symmetric. There are the eyes, the nose, the mouth, located in certain places on the face. Each person's face is a pattern composed of a particular combination of these. By analyzing sample face images of a person, a learning program captures the pattern specific to that person and then recognizes by checking for this pattern in a given image. This is one example of *pattern recognition*.

Machine learning is programming computers to optimize a performance criterion using example data or past experience. We have a model defined up to some parameters, and learning is the execution of a computer program to optimize the parameters of the model using the training data or past experience. The model may be *predictive* to make predictions in the future, or *descriptive* to gain knowledge from data, or both.

Machine learning uses the theory of statistics in building mathematical models, because the core task is making inference from a sample. The role of computer science is twofold: First, in training, we need efficient

algorithms to solve the optimization problem, as well as to store and process the massive amount of data we generally have. Second, once a model is learned, its representation and algorithmic solution for inference needs to be efficient as well. In certain applications, the efficiency of the learning or inference algorithm, namely, its space and time complexity, may be as important as its predictive accuracy.

Let us now discuss some example applications in more detail to gain more insight into the types and uses of machine learning.

1.2 Examples of Machine Learning Applications

1.2.1 Learning Associations

In the case of retail—for example, a supermarket chain—one application of machine learning is *basket analysis*, which is finding associations between products bought by customers: If people who buy X typically also buy Y, and if there is a customer who buys X and does not buy Y, he or she is a potential Y customer. Once we find such customers, we can target them for cross-selling.

ASSOCIATION RULE

In finding an *association rule*, we are interested in learning a conditional probability of the form $P(Y|X)$ where Y is the product we would like to condition on X, which is the product or the set of products which we know that the customer has already purchased.

Let us say, going over our data, we calculate that $P(\text{chips}|\text{beer}) = 0.7$. Then, we can define the rule:

70 percent of customers who buy beer also buy chips.

We may want to make a distinction among customers and toward this, estimate $P(Y|X, D)$ where D is the set of customer attributes, for example, gender, age, marital status, and so on, assuming that we have access to this information. If this is a bookseller instead of a supermarket, products can be books or authors. In the case of a web portal, items correspond to links to web pages, and we can estimate the links a user is likely to click and use this information to download such pages in advance for faster access.

1.2.2 Classification

A credit is an amount of money loaned by a financial institution, for example, a bank, to be paid back with interest, generally in installments. It is important for the bank to be able to predict in advance the risk associated with a loan, which is the probability that the customer will default and not pay the whole amount back. This is both to make sure that the bank will make a profit and also to not inconvenience a customer with a loan over his or her financial capacity.

In *credit scoring* (Hand 1998), the bank calculates the risk given the amount of credit and the information about the customer. The information about the customer includes data we have access to and is relevant in calculating his or her financial capacity—namely, income, savings, collaterals, profession, age, past financial history, and so forth. The bank has a record of past loans containing such customer data and whether the loan was paid back or not. From this data of particular applications, the aim is to infer a general rule coding the association between a customer's attributes and his risk. That is, the machine learning system fits a model to the past data to be able to calculate the risk for a new application and then decides to accept or refuse it accordingly.

CLASSIFICATION

This is an example of a *classification* problem where there are two classes: low-risk and high-risk customers. The information about a customer makes up the *input* to the classifier whose task is to assign the input to one of the two classes.

After training with the past data, a classification rule learned may be of the form

IF income$> \theta_1$ AND savings$> \theta_2$ THEN low-risk ELSE high-risk

for suitable values of θ_1 and θ_2 (see figure 1.1). This is an example of a *discriminant*; it is a function that separates the examples of different classes.

DISCRIMINANT

PREDICTION

Having a rule like this, the main application is *prediction*: Once we have a rule that fits the past data, if the future is similar to the past, then we can make correct predictions for novel instances. Given a new application with a certain income and savings, we can easily decide whether it is low-risk or high-risk.

In some cases, instead of making a 0/1 (low-risk/high-risk) type decision, we may want to calculate a probability, namely, $P(Y|X)$, where X are the customer attributes and Y is 0 or 1 respectively for low-risk

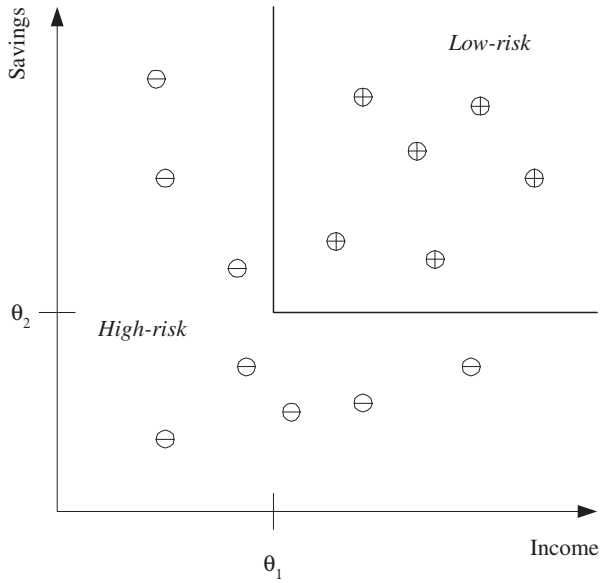

Figure 1.1 Example of a training dataset where each circle corresponds to one data instance with input values in the corresponding axes and its sign indicates the class. For simplicity, only two customer attributes, income and savings, are taken as input and the two classes are low-risk ('+') and high-risk ('−'). An example discriminant that separates the two types of examples is also shown.

and high-risk. From this perspective, we can see classification as learning an association from X to Y. Then for a given $X = x$, if we have $P(Y = 1|X = x) = 0.8$, we say that the customer has an 80 percent probability of being high-risk, or equivalently a 20 percent probability of being low-risk. We then decide whether to accept or refuse the loan depending on the possible gain and loss.

PATTERN RECOGNITION

There are many applications of machine learning in *pattern recognition*. One is *optical character recognition*, which is recognizing character codes from their images. This is an example where there are multiple classes, as many as there are characters we would like to recognize. Especially interesting is the case when the characters are handwritten—for example, to read zip codes on envelopes or amounts on checks. People have different handwriting styles; characters may be written small or large, slanted, with a pen or pencil, and there are many possible images corresponding

to the same character. Though writing is a human invention, we do not have any system that is as accurate as a human reader. We do not have a formal description of 'A' that covers all 'A's and none of the non-'A's. Not having it, we take samples from writers and learn a definition of A-ness from these examples. But though we do not know what it is that makes an image an 'A', we are certain that all those distinct 'A's have something in common, which is what we want to extract from the examples. We know that a character image is not just a collection of random dots; it is a collection of strokes and has a regularity that we can capture by a learning program.

If we are reading a text, one factor we can make use of is the redundancy in human languages. A word is a *sequence* of characters and successive characters are not independent but are constrained by the words of the language. This has the advantage that even if we cannot recognize a character, we can still read t?e word. Such contextual dependencies may also occur in higher levels, between words and sentences, through the syntax and semantics of the language. There are machine learning algorithms to learn sequences and model such dependencies.

In the case of *face recognition*, the input is an image, the classes are people to be recognized, and the learning program should learn to associate the face images to identities. This problem is more difficult than optical character recognition because there are more classes, input image is larger, and a face is three-dimensional and differences in pose and lighting cause significant changes in the image. There may also be occlusion of certain inputs; for example, glasses may hide the eyes and eyebrows, and a beard may hide the chin.

In *medical diagnosis*, the inputs are the relevant information we have about the patient and the classes are the illnesses. The inputs contain the patient's age, gender, past medical history, and current symptoms. Some tests may not have been applied to the patient, and thus these inputs would be missing. Tests take time, may be costly, and may inconvenience the patient so we do not want to apply them unless we believe that they will give us valuable information. In the case of a medical diagnosis, a wrong decision may lead to a wrong or no treatment, and in cases of doubt it is preferable that the classifier reject and defer decision to a human expert.

In *speech recognition*, the input is acoustic and the classes are words that can be uttered. This time the association to be learned is from an acoustic signal to a word of some language. Different people, because

of differences in age, gender, or accent, pronounce the same word differently, which makes this task rather difficult. Another difference of speech is that the input is *temporal*; words are uttered in time as a sequence of speech phonemes and some words are longer than others.

Acoustic information only helps up to a certain point, and as in optical character recognition, the integration of a "language model" is critical in speech recognition, and the best way to come up with a language model is again by learning it from some large corpus of example data. The applications of machine learning to *natural language processing* is constantly increasing. Spam filtering is one where spam generators on one side and filters on the other side keep finding more and more ingenious ways to outdo each other. Summarizing large documents is another interesting example, yet another is analyzing blogs or posts on social networking sites to extract "trending" topics or to determine what to advertise. Perhaps the most impressive would be *machine translation*. After decades of research on hand-coded translation rules, it has become apparent that the most promising way is to provide a very large number of example pairs of texts in both languages and have a program figure out automatically the rules to map one to the other.

Biometrics is recognition or authentication of people using their physiological and/or behavioral characteristics that requires an integration of inputs from different modalities. Examples of physiological characteristics are images of the face, fingerprint, iris, and palm; examples of behavioral characteristics are dynamics of signature, voice, gait, and key stroke. As opposed to the usual identification procedures—photo, printed signature, or password—when there are many different (uncorrelated) inputs, forgeries (spoofing) would be more difficult and the system would be more accurate, hopefully without too much inconvenience to the users. Machine learning is used both in the separate recognizers for these different modalities and in the combination of their decisions to get an overall accept/reject decision, taking into account how reliable these different sources are.

KNOWLEDGE EXTRACTION

Learning a rule from data also allows *knowledge extraction*. The rule is a simple model that explains the data, and looking at this model we have an explanation about the process underlying the data. For example, once we learn the discriminant separating low-risk and high-risk customers, we have the knowledge of the properties of low-risk customers. We can then use this information to target potential low-risk customers more efficiently, for example, through advertising. Learning also performs *com-*

COMPRESSION