

Machine Learning IN ACTION

Peter Harrington



contents

preface xvii
acknowledgments xix
about this book xxi
about the author xxv
about the cover illustration xxvi

PART 1 CLASSIFICATION1

1 *Machine learning basics 3*

- 1.1 What is machine learning? 5
 - Sensors and the data deluge 6* ▪ *Machine learning will be more important in the future 7*
- 1.2 Key terminology 7
- 1.3 Key tasks of machine learning 10
- 1.4 How to choose the right algorithm 11
- 1.5 Steps in developing a machine learning application 11
- 1.6 Why Python? 13
 - Executable pseudo-code 13* ▪ *Python is popular 13* ▪ *What Python has that other languages don't have 14* ▪ *Drawbacks 14*
- 1.7 Getting started with the NumPy library 15
- 1.8 Summary 17

- ## 2 *Classifying with k-Nearest Neighbors* 18
- 2.1 Classifying with distance measurements 19
 - Prepare: importing data with Python* 21 ▪ *Putting the kNN classification algorithm into action* 23 ▪ *How to test a classifier* 24
 - 2.2 Example: improving matches from a dating site with kNN 24
 - Prepare: parsing data from a text file* 25 ▪ *Analyze: creating scatter plots with Matplotlib* 27 ▪ *Prepare: normalizing numeric values* 29 ▪ *Test: testing the classifier as a whole program* 31 ▪ *Use: putting together a useful system* 32
 - 2.3 Example: a handwriting recognition system 33
 - Prepare: converting images into test vectors* 33 ▪ *Test: kNN on handwritten digits* 35
 - 2.4 Summary 36
- ## 3 *Splitting datasets one feature at a time: decision trees* 37
- 3.1 Tree construction 39
 - Information gain* 40 ▪ *Splitting the dataset* 43 ▪ *Recursively building the tree* 46
 - 3.2 Plotting trees in Python with Matplotlib annotations 48
 - Matplotlib annotations* 49 ▪ *Constructing a tree of annotations* 51
 - 3.3 Testing and storing the classifier 56
 - Test: using the tree for classification* 56 ▪ *Use: persisting the decision tree* 57
 - 3.4 Example: using decision trees to predict contact lens type 57
 - 3.5 Summary 59
- ## 4 *Classifying with probability theory: naïve Bayes* 61
- 4.1 Classifying with Bayesian decision theory 62
 - 4.2 Conditional probability 63
 - 4.3 Classifying with conditional probabilities 65
 - 4.4 Document classification with naïve Bayes 65
 - 4.5 Classifying text with Python 67
 - Prepare: making word vectors from text* 67 ▪ *Train: calculating probabilities from word vectors* 69 ▪ *Test: modifying the classifier for real-world conditions* 71 ▪ *Prepare: the bag-of-words document model* 73
 - 4.6 Example: classifying spam email with naïve Bayes 74
 - Prepare: tokenizing text* 74 ▪ *Test: cross validation with naïve Bayes* 75

- 4.7 Example: using naïve Bayes to reveal local attitudes from personal ads 77
 - Collect: importing RSS feeds* 78 ▪ *Analyze: displaying locally used words* 80
- 4.8 Summary 82

5 *Logistic regression* 83

- 5.1 Classification with logistic regression and the sigmoid function: a tractable step function 84
- 5.2 Using optimization to find the best regression coefficients 86
 - Gradient ascent* 86 ▪ *Train: using gradient ascent to find the best parameters* 88 ▪ *Analyze: plotting the decision boundary* 90
 - Train: stochastic gradient ascent* 91
- 5.3 Example: estimating horse fatalities from colic 96
 - Prepare: dealing with missing values in the data* 97 ▪ *Test: classifying with logistic regression* 98
- 5.4 Summary 100

6 *Support vector machines* 101

- 6.1 Separating data with the maximum margin 102
- 6.2 Finding the maximum margin 104
 - Framing the optimization problem in terms of our classifier* 104
 - Approaching SVMs with our general framework* 106
- 6.3 Efficient optimization with the SMO algorithm 106
 - Platt's SMO algorithm* 106 ▪ *Solving small datasets with the simplified SMO* 107
- 6.4 Speeding up optimization with the full Platt SMO 112
- 6.5 Using kernels for more complex data 118
 - Mapping data to higher dimensions with kernels* 118 ▪ *The radial bias function as a kernel* 119 ▪ *Using a kernel for testing* 122
- 6.6 Example: revisiting handwriting classification 125
- 6.7 Summary 127

7 *Improving classification with the AdaBoost meta-algorithm* 129

- 7.1 Classifiers using multiple samples of the dataset 130
 - Building classifiers from randomly resampled data: bagging* 130
 - Boosting* 131
- 7.2 Train: improving the classifier by focusing on errors 131

- 7.3 Creating a weak learner with a decision stump 133
- 7.4 Implementing the full AdaBoost algorithm 136
- 7.5 Test: classifying with AdaBoost 139
- 7.6 Example: AdaBoost on a difficult dataset 140
- 7.7 Classification imbalance 142
 - Alternative performance metrics: precision, recall, and ROC* 143
 - Manipulating the classifier's decision with a cost function* 147
 - Data sampling for dealing with classification imbalance* 148
- 7.8 Summary 148

PART 2 FORECASTING NUMERIC VALUES WITH REGRESSION .151

8 Predicting numeric values: regression 153

- 8.1 Finding best-fit lines with linear regression 154
- 8.2 Locally weighted linear regression 160
- 8.3 Example: predicting the age of an abalone 163
- 8.4 Shrinking coefficients to understand our data 164
 - Ridge regression* 164 ▪ *The lasso* 167 ▪ *Forward stagewise regression* 167
- 8.5 The bias/variance tradeoff 170
- 8.6 Example: forecasting the price of LEGO sets 172
 - Collect: using the Google shopping API* 173 ▪ *Train: building a model* 174
- 8.7 Summary 177

9 Tree-based regression 179

- 9.1 Locally modeling complex data 180
- 9.2 Building trees with continuous and discrete features 181
- 9.3 Using CART for regression 184
 - Building the tree* 184 ▪ *Executing the code* 186
- 9.4 Tree pruning 188
 - Prepruning* 188 ▪ *Postpruning* 190
- 9.5 Model trees 192
- 9.6 Example: comparing tree methods to standard regression 195
- 9.7 Using Tkinter to create a GUI in Python 198
 - Building a GUI in Tkinter* 199 ▪ *Interfacing Matplotlib and Tkinter* 201
- 9.8 Summary 203

PART 3 UNSUPERVISED LEARNING205

10 *Grouping unlabeled items using k-means clustering* 207

- 10.1 The k-means clustering algorithm 208
- 10.2 Improving cluster performance with postprocessing 213
- 10.3 Bisecting k-means 214
- 10.4 Example: clustering points on a map 217
 - The Yahoo! PlaceFinder API* 218 ▪ *Clustering geographic coordinates* 220
- 10.5 Summary 223

11 *Association analysis with the Apriori algorithm* 224

- 11.1 Association analysis 225
- 11.2 The Apriori principle 226
- 11.3 Finding frequent itemsets with the Apriori algorithm 228
 - Generating candidate itemsets* 229 ▪ *Putting together the full Apriori algorithm* 231
- 11.4 Mining association rules from frequent item sets 233
- 11.5 Example: uncovering patterns in congressional voting 237
 - Collect: build a transaction data set of congressional voting records* 238 ▪ *Test: association rules from congressional voting records* 243
- 11.6 Example: finding similar features in poisonous mushrooms 245
- 11.7 Summary 246

12 *Efficiently finding frequent itemsets with FP-growth* 248

- 12.1 FP-trees: an efficient way to encode a dataset 249
- 12.2 Build an FP-tree 251
 - Creating the FP-tree data structure* 251 ▪ *Constructing the FP-tree* 252
- 12.3 Mining frequent items from an FP-tree 256
 - Extracting conditional pattern bases* 257 ▪ *Creating conditional FP-trees* 258
- 12.4 Example: finding co-occurring words in a Twitter feed 260
- 12.5 Example: mining a clickstream from a news site 264
- 12.6 Summary 265

PART 4 ADDITIONAL TOOLS267

13 *Using principal component analysis to simplify data* 269

- 13.1 Dimensionality reduction techniques 270
- 13.2 Principal component analysis 271
 - Moving the coordinate axes* 271 ▪ *Performing PCA in NumPy* 273
- 13.3 Example: using PCA to reduce the dimensionality of semiconductor manufacturing data 275
- 13.4 Summary 278

14 *Simplifying data with the singular value decomposition* 280

- 14.1 Applications of the SVD 281
 - Latent semantic indexing* 281 ▪ *Recommendation systems* 282
- 14.2 Matrix factorization 283
- 14.3 SVD in Python 284
- 14.4 Collaborative filtering–based recommendation engines 286
 - Measuring similarity* 287 ▪ *Item-based or user-based similarity?* 289
 - Evaluating recommendation engines* 289
- 14.5 Example: a restaurant dish recommendation engine 290
 - Recommending untasted dishes* 290 ▪ *Improving recommendations with the SVD* 292 ▪ *Challenges with building recommendation engines* 295
- 14.6 Example: image compression with the SVD 295
- 14.7 Summary 298

15 *Big data and MapReduce* 299

- 15.1 MapReduce: a framework for distributed computing 300
- 15.2 Hadoop Streaming 302
 - Distributed mean and variance mapper* 303 ▪ *Distributed mean and variance reducer* 304
- 15.3 Running Hadoop jobs on Amazon Web Services 305
 - Services available on AWS* 305 ▪ *Getting started with Amazon Web Services* 306 ▪ *Running a Hadoop job on EMR* 307
- 15.4 Machine learning in MapReduce 312
- 15.5 Using mrjob to automate MapReduce in Python 313
 - Using mrjob for seamless integration with EMR* 313 ▪ *The anatomy of a MapReduce script in mrjob* 314

| | | |
|-------------------|---|-----|
| 15.6 | Example: the Pegasos algorithm for distributed SVMs | 316 |
| | <i>The Pegasos algorithm</i> | 317 |
| | ▪ <i>Training: MapReduce support</i> | |
| | <i>vector machines with mrjob</i> | 318 |
| 15.7 | Do you really need MapReduce? | 322 |
| 15.8 | Summary | 323 |
| <i>appendix A</i> | <i>Getting started with Python</i> | 325 |
| <i>appendix B</i> | <i>Linear algebra</i> | 335 |
| <i>appendix C</i> | <i>Probability refresher</i> | 341 |
| <i>appendix D</i> | <i>Resources</i> | 345 |
| | <i>index</i> | 347 |