Gareth James • Daniela Witten •

Trevor Hastie • Robert Tibshirani

An Introduction to Statistical Learning

with Applications in R

Second Edition

123

*To our parents:*

*Alison and Michael James Chiara Nappi and Edward Witten Valerie and Patrick Hastie Vera and Sami Tibshirani*

*and to our families:*

*Michael, Daniel, and Catherine Tessa, Theo, Otto, and Ari Samantha, Timothy, and Lynda Charlie, Ryan, Julie, and Cheryl*

Preface

Statistical learning refers to a set of tools for *making sense of complex datasets*. In recent years, we have seen a staggering increase in the scale and scope of data collection across virtually all areas of science and industry. As a result, statistical learning has become a critical toolkit for anyone who wishes to understand data — and as more and more of today’s jobs involve data, this means that statistical learning is fast becoming a critical toolkit for *everyone*.

One of the first books on statistical learning — *The Elements of Statisti cal Learning* (ESL, by Hastie, Tibshirani, and Friedman) — was published in 2001, with a second edition in 2009. ESL has become a popular text not only in statistics but also in related fields. One of the reasons for ESL’s popularity is its relatively accessible style. But ESL is best-suited for indi viduals with advanced training in the mathematical sciences.

*An Introduction to Statistical Learning* (ISL) arose from the clear need for a broader and less technical treatment of the key topics in statistical learning. The intention behind ISL is to concentrate more on the applica tions of the methods and less on the mathematical details. Beginning with Chapter 2, each chapter in ISL contains a lab illustrating how to implement the statistical learning methods seen in that chapter using the popular sta tistical software package R. These labs provide the reader with valuable hands-on experience.

ISL is appropriate for advanced undergraduates or master’s students in Statistics or related quantitative fields, or for individuals in other disciplines who wish to use statistical learning tools to analyze their data. It can be used as a textbook for a course spanning two semesters.

vii

viii Preface

The first edition of ISL covered a number of important topics, including sparse methods for classification and regression, decision trees, boosting, support vector machines, and clustering. Since it was published in 2013, it has become a mainstay of undergraduate and graduate classrooms across the United States and worldwide, as well as a key reference book for data scientists.

In this second edition of ISL, we have greatly expanded the set of topics covered. In particular, the second edition includes new chapters on deep learning (Chapter 10), survival analysis (Chapter 11), and multiple testing (Chapter 13). We have also substantially expanded some chapters that were part of the first edition: among other updates, we now include treatments of naive Bayes and generalized linear models in Chapter 4, Bayesian addi tive regression trees in Chapter 8, and matrix completion in Chapter 12. Furthermore, we have updated the R code throughout the labs to ensure that the results that they produce agree with recent R releases.

We are grateful to these readers for providing valuable comments on the first edition of this book: Pallavi Basu, Alexandra Chouldechova, Patrick Danaher, Will Fithian, Luella Fu, Sam Gross, Max Grazier G’Sell, Court ney Paulson, Xinghao Qiao, Elisa Sheng, Noah Simon, Kean Ming Tan, Xin Lu Tan. We thank these readers for helpful input on the second edi tion of this book: Alan Agresti, Iain Carmichael, Yiqun Chen, Erin Craig, Daisy Ding, Lucy Gao, Ismael Lemhadri, Bryan Martin, Anna Neufeld, Ge off Tims, Carsten Voelkmann, Steve Yadlowsky, and James Zou. We also thank Anna Neufeld for her assistance in reformatting the R code through out this book. We are immensely grateful to Balasubramanian “Naras” Narasimhan for his assistance on both editions of this textbook.

It has been an honor and a privilege for us to see the considerable impact that the first edition of ISL has had on the way in which statistical learning is practiced, both in and out of the academic setting. We hope that this new edition will continue to give today’s and tomorrow’s applied statisticians and data scientists the tools they need for success in a data-driven world.

*It’s tough to make predictions, especially about the future.* -Yogi Berra

Contents

Preface vii 1 Introduction 1

2 Statistical Learning 15 2.1 What Is Statistical Learning? . . . . . . . . . . . . . . . . . 15 2.1.1 Why Estimate *f*? . . . . . . . . . . . . . . . . . . . 17 2.1.2 How Do We Estimate *f*? . . . . . . . . . . . . . . . 21 2.1.3 The Trade-Off Between Prediction Accuracy

and Model Interpretability . . . . . . . . . . . . . . 24 2.1.4 Supervised Versus Unsupervised Learning . . . . . 26 2.1.5 Regression Versus Classification Problems . . . . . 28

2.2 Assessing Model Accuracy . . . . . . . . . . . . . . . . . . 29 2.2.1 Measuring the Quality of Fit . . . . . . . . . . . . 29 2.2.2 The Bias-Variance Trade-Off . . . . . . . . . . . . . 33 2.2.3 The Classification Setting . . . . . . . . . . . . . . 37

2.3 Lab: Introduction to R . . . . . . . . . . . . . . . . . . . . 42 2.3.1 Basic Commands . . . . . . . . . . . . . . . . . . . 43 2.3.2 Graphics . . . . . . . . . . . . . . . . . . . . . . . . 45 2.3.3 Indexing Data . . . . . . . . . . . . . . . . . . . . . 47 2.3.4 Loading Data . . . . . . . . . . . . . . . . . . . . . 48 2.3.5 Additional Graphical and Numerical Summaries . . 50 2.4 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 52

3 Linear Regression 59 3.1 Simple Linear Regression . . . . . . . . . . . . . . . . . . . 60 3.1.1 Estimating the Coefficients . . . . . . . . . . . . . 61 3.1.2 Assessing the Accuracy of the Coefficient

Estimates . . . . . . . . . . . . . . . . . . . . . . . 63 3.1.3 Assessing the Accuracy of the Model . . . . . . . . 68 3.2 Multiple Linear Regression . . . . . . . . . . . . . . . . . . 71 3.2.1 Estimating the Regression Coefficients . . . . . . . 72 3.2.2 Some Important Questions . . . . . . . . . . . . . . 75 3.3 Other Considerations in the Regression Model . . . . . . . 83

ix

x CONTENTS

3.3.1 Qualitative Predictors . . . . . . . . . . . . . . . . 83 3.3.2 Extensions of the Linear Model . . . . . . . . . . . 87 3.3.3 Potential Problems . . . . . . . . . . . . . . . . . . 92

3.4 The Marketing Plan . . . . . . . . . . . . . . . . . . . . . . 103 3.5 Comparison of Linear Regression with *K*-Nearest

Neighbors . . . . . . . . . . . . . . . . . . . . . . . . . . . . 105 3.6 Lab: Linear Regression . . . . . . . . . . . . . . . . . . . . 110 3.6.1 Libraries . . . . . . . . . . . . . . . . . . . . . . . . 110 3.6.2 Simple Linear Regression . . . . . . . . . . . . . . . 111 3.6.3 Multiple Linear Regression . . . . . . . . . . . . . . 114 3.6.4 Interaction Terms . . . . . . . . . . . . . . . . . . . 116 3.6.5 Non-linear Transformations of the Predictors . . . 116 3.6.6 Qualitative Predictors . . . . . . . . . . . . . . . . 119 3.6.7 Writing Functions . . . . . . . . . . . . . . . . . . . 120 3.7 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 121

4 Classification 129 4.1 An Overview of Classification . . . . . . . . . . . . . . . . . 130 4.2 Why Not Linear Regression? . . . . . . . . . . . . . . . . . 131 4.3 Logistic Regression . . . . . . . . . . . . . . . . . . . . . . 133

4.3.1 The Logistic Model . . . . . . . . . . . . . . . . . . 133 4.3.2 Estimating the Regression Coefficients . . . . . . . 135 4.3.3 Making Predictions . . . . . . . . . . . . . . . . . . 136 4.3.4 Multiple Logistic Regression . . . . . . . . . . . . . 137 4.3.5 Multinomial Logistic Regression . . . . . . . . . . . 140

4.4 Generative Models for Classification . . . . . . . . . . . . . 141 4.4.1 Linear Discriminant Analysis for *p* = 1 . . . . . . . 142 4.4.2 Linear Discriminant Analysis for *p >*1 . . . . . . . 145 4.4.3 Quadratic Discriminant Analysis . . . . . . . . . . 152 4.4.4 Naive Bayes . . . . . . . . . . . . . . . . . . . . . . 153

4.5 A Comparison of Classification Methods . . . . . . . . . . 158 4.5.1 An Analytical Comparison . . . . . . . . . . . . . . 158 4.5.2 An Empirical Comparison . . . . . . . . . . . . . . 161

4.6 Generalized Linear Models . . . . . . . . . . . . . . . . . . 164 4.6.1 Linear Regression on the Bikeshare Data . . . . . . 164 4.6.2 Poisson Regression on the Bikeshare Data . . . . . 167 4.6.3 Generalized Linear Models in Greater Generality . 170

4.7 Lab: Classification Methods . . . . . . . . . . . . . . . . . . 171 4.7.1 The Stock Market Data . . . . . . . . . . . . . . . 171 4.7.2 Logistic Regression . . . . . . . . . . . . . . . . . . 172 4.7.3 Linear Discriminant Analysis . . . . . . . . . . . . 177 4.7.4 Quadratic Discriminant Analysis . . . . . . . . . . 179 4.7.5 Naive Bayes . . . . . . . . . . . . . . . . . . . . . . 180 4.7.6 *K*-Nearest Neighbors . . . . . . . . . . . . . . . . . 181 4.7.7 Poisson Regression . . . . . . . . . . . . . . . . . . 185

CONTENTS xi

4.8 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 189

5 Resampling Methods 197 5.1 Cross-Validation . . . . . . . . . . . . . . . . . . . . . . . . 198 5.1.1 The Validation Set Approach . . . . . . . . . . . . 198 5.1.2 Leave-One-Out Cross-Validation . . . . . . . . . . 200 5.1.3 *k*-Fold Cross-Validation . . . . . . . . . . . . . . . 203 5.1.4 Bias-Variance Trade-Off for *k*-Fold

Cross-Validation . . . . . . . . . . . . . . . . . . . 205 5.1.5 Cross-Validation on Classification Problems . . . . 206 5.2 The Bootstrap . . . . . . . . . . . . . . . . . . . . . . . . . 209 5.3 Lab: Cross-Validation and the Bootstrap . . . . . . . . . . 212 5.3.1 The Validation Set Approach . . . . . . . . . . . . 213 5.3.2 Leave-One-Out Cross-Validation . . . . . . . . . . 214 5.3.3 *k*-Fold Cross-Validation . . . . . . . . . . . . . . . 215 5.3.4 The Bootstrap . . . . . . . . . . . . . . . . . . . . 216 5.4 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 219

6 Linear Model Selection and Regularization 225 6.1 Subset Selection . . . . . . . . . . . . . . . . . . . . . . . . 227 6.1.1 Best Subset Selection . . . . . . . . . . . . . . . . . 227 6.1.2 Stepwise Selection . . . . . . . . . . . . . . . . . . 229 6.1.3 Choosing the Optimal Model . . . . . . . . . . . . 232 6.2 Shrinkage Methods . . . . . . . . . . . . . . . . . . . . . . 237 6.2.1 Ridge Regression . . . . . . . . . . . . . . . . . . . 237 6.2.2 The Lasso . . . . . . . . . . . . . . . . . . . . . . . 241 6.2.3 Selecting the Tuning Parameter . . . . . . . . . . . 250 6.3 Dimension Reduction Methods . . . . . . . . . . . . . . . . 251 6.3.1 Principal Components Regression . . . . . . . . . . 252 6.3.2 Partial Least Squares . . . . . . . . . . . . . . . . . 259 6.4 Considerations in High Dimensions . . . . . . . . . . . . . 261 6.4.1 High-Dimensional Data . . . . . . . . . . . . . . . . 261 6.4.2 What Goes Wrong in High Dimensions? . . . . . . 262 6.4.3 Regression in High Dimensions . . . . . . . . . . . 264 6.4.4 Interpreting Results in High Dimensions . . . . . . 266 6.5 Lab: Linear Models and Regularization Methods . . . . . . 267 6.5.1 Subset Selection Methods . . . . . . . . . . . . . . 267 6.5.2 Ridge Regression and the Lasso . . . . . . . . . . . 274 6.5.3 PCR and PLS Regression . . . . . . . . . . . . . . 279 6.6 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 282

7 Moving Beyond Linearity 289 7.1 Polynomial Regression . . . . . . . . . . . . . . . . . . . . . 290 7.2 Step Functions . . . . . . . . . . . . . . . . . . . . . . . . . 292 7.3 Basis Functions . . . . . . . . . . . . . . . . . . . . . . . . 294

xii CONTENTS

7.4 Regression Splines . . . . . . . . . . . . . . . . . . . . . . . 295 7.4.1 Piecewise Polynomials . . . . . . . . . . . . . . . . 295 7.4.2 Constraints and Splines . . . . . . . . . . . . . . . 295 7.4.3 The Spline Basis Representation . . . . . . . . . . 297 7.4.4 Choosing the Number and Locations

of the Knots . . . . . . . . . . . . . . . . . . . . . . 298 7.4.5 Comparison to Polynomial Regression . . . . . . . 300 7.5 Smoothing Splines . . . . . . . . . . . . . . . . . . . . . . . 301 7.5.1 An Overview of Smoothing Splines . . . . . . . . . 301 7.5.2 Choosing the Smoothing Parameter *λ* . . . . . . . 302 7.6 Local Regression . . . . . . . . . . . . . . . . . . . . . . . . 304 7.7 Generalized Additive Models . . . . . . . . . . . . . . . . . 306 7.7.1 GAMs for Regression Problems . . . . . . . . . . . 307 7.7.2 GAMs for Classification Problems . . . . . . . . . . 310 7.8 Lab: Non-linear Modeling . . . . . . . . . . . . . . . . . . . 311 7.8.1 Polynomial Regression and Step Functions . . . . . 312 7.8.2 Splines . . . . . . . . . . . . . . . . . . . . . . . . . 317 7.8.3 GAMs . . . . . . . . . . . . . . . . . . . . . . . . . 318 7.9 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 321

8 Tree-Based Methods 327 8.1 The Basics of Decision Trees . . . . . . . . . . . . . . . . . 327 8.1.1 Regression Trees . . . . . . . . . . . . . . . . . . . 328 8.1.2 Classification Trees . . . . . . . . . . . . . . . . . . 335 8.1.3 Trees Versus Linear Models . . . . . . . . . . . . . 338 8.1.4 Advantages and Disadvantages of Trees . . . . . . 339

8.2 Bagging, Random Forests, Boosting, and Bayesian Additive Regression Trees . . . . . . . . . . . . . . . . . . . . . . . . 340 8.2.1 Bagging . . . . . . . . . . . . . . . . . . . . . . . . 340 8.2.2 Random Forests . . . . . . . . . . . . . . . . . . . . 343 8.2.3 Boosting . . . . . . . . . . . . . . . . . . . . . . . . 345 8.2.4 Bayesian Additive Regression Trees . . . . . . . . . 348 8.2.5 Summary of Tree Ensemble Methods . . . . . . . . 351

8.3 Lab: Decision Trees . . . . . . . . . . . . . . . . . . . . . . 353 8.3.1 Fitting Classification Trees . . . . . . . . . . . . . . 353 8.3.2 Fitting Regression Trees . . . . . . . . . . . . . . . 356 8.3.3 Bagging and Random Forests . . . . . . . . . . . . 357 8.3.4 Boosting . . . . . . . . . . . . . . . . . . . . . . . . 359 8.3.5 Bayesian Additive Regression Trees . . . . . . . . . 360 8.4 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 361

9 Support Vector Machines 367 9.1 Maximal Margin Classifier . . . . . . . . . . . . . . . . . . 368 9.1.1 What Is a Hyperplane? . . . . . . . . . . . . . . . . 368 9.1.2 Classification Using a Separating Hyperplane . . . 369

CONTENTS xiii

9.1.3 The Maximal Margin Classifier . . . . . . . . . . . 371 9.1.4 Construction of the Maximal Margin Classifier . . 372 9.1.5 The Non-separable Case . . . . . . . . . . . . . . . 373

9.2 Support Vector Classifiers . . . . . . . . . . . . . . . . . . . 373 9.2.1 Overview of the Support Vector Classifier . . . . . 373 9.2.2 Details of the Support Vector Classifier . . . . . . . 375

9.3 Support Vector Machines . . . . . . . . . . . . . . . . . . . 379 9.3.1 Classification with Non-Linear Decision

Boundaries . . . . . . . . . . . . . . . . . . . . . . 379 9.3.2 The Support Vector Machine . . . . . . . . . . . . 380 9.3.3 An Application to the Heart Disease Data . . . . . 383

9.4 SVMs with More than Two Classes . . . . . . . . . . . . . 385 9.4.1 One-Versus-One Classification . . . . . . . . . . . . 385 9.4.2 One-Versus-All Classification . . . . . . . . . . . . 385

9.5 Relationship to Logistic Regression . . . . . . . . . . . . . 386 9.6 Lab: Support Vector Machines . . . . . . . . . . . . . . . . 388 9.6.1 Support Vector Classifier . . . . . . . . . . . . . . . 389 9.6.2 Support Vector Machine . . . . . . . . . . . . . . . 392 9.6.3 ROC Curves . . . . . . . . . . . . . . . . . . . . . . 394 9.6.4 SVM with Multiple Classes . . . . . . . . . . . . . 396 9.6.5 Application to Gene Expression Data . . . . . . . . 396 9.7 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 398

10 Deep Learning 403 10.1 Single Layer Neural Networks . . . . . . . . . . . . . . . . 404 10.2 Multilayer Neural Networks . . . . . . . . . . . . . . . . . . 407 10.3 Convolutional Neural Networks . . . . . . . . . . . . . . . . 411

10.3.1 Convolution Layers . . . . . . . . . . . . . . . . . . 412 10.3.2 Pooling Layers . . . . . . . . . . . . . . . . . . . . 415 10.3.3 Architecture of a Convolutional Neural Network . . 415 10.3.4 Data Augmentation . . . . . . . . . . . . . . . . . . 417 10.3.5 Results Using a Pretrained Classifier . . . . . . . . 417

10.4 Document Classification . . . . . . . . . . . . . . . . . . . . 419 10.5 Recurrent Neural Networks . . . . . . . . . . . . . . . . . . 421 10.5.1 Sequential Models for Document Classification . . 424 10.5.2 Time Series Forecasting . . . . . . . . . . . . . . . 427 10.5.3 Summary of RNNs . . . . . . . . . . . . . . . . . . 431 10.6 When to Use Deep Learning . . . . . . . . . . . . . . . . . 432 10.7 Fitting a Neural Network . . . . . . . . . . . . . . . . . . . 434 10.7.1 Backpropagation . . . . . . . . . . . . . . . . . . . 435 10.7.2 Regularization and Stochastic Gradient Descent . . 436 10.7.3 Dropout Learning . . . . . . . . . . . . . . . . . . . 438 10.7.4 Network Tuning . . . . . . . . . . . . . . . . . . . . 438 10.8 Interpolation and Double Descent . . . . . . . . . . . . . . 439 10.9 Lab: Deep Learning . . . . . . . . . . . . . . . . . . . . . . 443

xiv CONTENTS

10.9.1 A Single Layer Network on the Hitters Data . . . . 443 10.9.2 A Multilayer Network on the MNIST Digit Data . 445 10.9.3 Convolutional Neural Networks . . . . . . . . . . . 448 10.9.4 Using Pretrained CNN Models . . . . . . . . . . . 451 10.9.5 IMDb Document Classification . . . . . . . . . . . 452 10.9.6 Recurrent Neural Networks . . . . . . . . . . . . . 454 10.10 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 458

11 Survival Analysis and Censored Data 461 11.1 Survival and Censoring Times . . . . . . . . . . . . . . . . 462 11.2 A Closer Look at Censoring . . . . . . . . . . . . . . . . . . 463 11.3 The Kaplan-Meier Survival Curve . . . . . . . . . . . . . . 464 11.4 The Log-Rank Test . . . . . . . . . . . . . . . . . . . . . . 466 11.5 Regression Models With a Survival Response . . . . . . . . 469

11.5.1 The Hazard Function . . . . . . . . . . . . . . . . . 469 11.5.2 Proportional Hazards . . . . . . . . . . . . . . . . . 471 11.5.3 Example: Brain Cancer Data . . . . . . . . . . . . 475 11.5.4 Example: Publication Data . . . . . . . . . . . . . 475

11.6 Shrinkage for the Cox Model . . . . . . . . . . . . . . . . . 478 11.7 Additional Topics . . . . . . . . . . . . . . . . . . . . . . . 480 11.7.1 Area Under the Curve for Survival Analysis . . . . 480 11.7.2 Choice of Time Scale . . . . . . . . . . . . . . . . . 481 11.7.3 Time-Dependent Covariates . . . . . . . . . . . . . 481 11.7.4 Checking the Proportional Hazards Assumption . . 482 11.7.5 Survival Trees . . . . . . . . . . . . . . . . . . . . . 482 11.8 Lab: Survival Analysis . . . . . . . . . . . . . . . . . . . . . 483 11.8.1 Brain Cancer Data . . . . . . . . . . . . . . . . . . 483 11.8.2 Publication Data . . . . . . . . . . . . . . . . . . . 486 11.8.3 Call Center Data . . . . . . . . . . . . . . . . . . . 487 11.9 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 490

12 Unsupervised Learning 497 12.1 The Challenge of Unsupervised Learning . . . . . . . . . . 497 12.2 Principal Components Analysis . . . . . . . . . . . . . . . . 498

12.2.1 What Are Principal Components? . . . . . . . . . . 499 12.2.2 Another Interpretation of Principal Components . 503 12.2.3 The Proportion of Variance Explained . . . . . . . 505 12.2.4 More on PCA . . . . . . . . . . . . . . . . . . . . . 507 12.2.5 Other Uses for Principal Components . . . . . . . . 510

12.3 Missing Values and Matrix Completion . . . . . . . . . . . 510 12.4 Clustering Methods . . . . . . . . . . . . . . . . . . . . . . 516 12.4.1 *K*-Means Clustering . . . . . . . . . . . . . . . . . 517 12.4.2 Hierarchical Clustering . . . . . . . . . . . . . . . . 521 12.4.3 Practical Issues in Clustering . . . . . . . . . . . . 530 12.5 Lab: Unsupervised Learning . . . . . . . . . . . . . . . . . 532

CONTENTS xv

12.5.1 Principal Components Analysis . . . . . . . . . . . 532 12.5.2 Matrix Completion . . . . . . . . . . . . . . . . . . 535 12.5.3 Clustering . . . . . . . . . . . . . . . . . . . . . . . 538 12.5.4 NCI60 Data Example . . . . . . . . . . . . . . . . . 542 12.6 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 548

13 Multiple Testing 553 13.1 A Quick Review of Hypothesis Testing . . . . . . . . . . . 554 13.1.1 Testing a Hypothesis . . . . . . . . . . . . . . . . . 555 13.1.2 Type I and Type II Errors . . . . . . . . . . . . . . 559 13.2 The Challenge of Multiple Testing . . . . . . . . . . . . . . 560 13.3 The Family-Wise Error Rate . . . . . . . . . . . . . . . . . 561 13.3.1 What is the Family-Wise Error Rate? . . . . . . . 562 13.3.2 Approaches to Control the Family-Wise Error Rate 564 13.3.3 Trade-Off Between the FWER and Power . . . . . 570 13.4 The False Discovery Rate . . . . . . . . . . . . . . . . . . . 571 13.4.1 Intuition for the False Discovery Rate . . . . . . . 571 13.4.2 The Benjamini-Hochberg Procedure . . . . . . . . 573

13.5 A Re-Sampling Approach to *p*-Values and False Discovery Rates . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 575 13.5.1 A Re-Sampling Approach to the *p*-Value . . . . . . 576 13.5.2 A Re-Sampling Approach to the False Discovery Rate578 13.5.3 When Are Re-Sampling Approaches Useful? . . . . 581

13.6 Lab: Multiple Testing . . . . . . . . . . . . . . . . . . . . . 582 13.6.1 Review of Hypothesis Tests . . . . . . . . . . . . . 582 13.6.2 The Family-Wise Error Rate . . . . . . . . . . . . . 583 13.6.3 The False Discovery Rate . . . . . . . . . . . . . . 586 13.6.4 A Re-Sampling Approach . . . . . . . . . . . . . . 588 13.7 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . 591

Index 597

1

Introduction

An Overview of Statistical Learning



*Statistical learning* refers to a vast set of tools for *understanding data*. These tools can be classified as *supervised* or *unsupervised*. Broadly speaking, supervised statistical learning involves building a statistical model for pre dicting, or estimating, an *output* based on one or more *inputs*. Problems of this nature occur in fields as diverse as business, medicine, astrophysics, and public policy. With unsupervised statistical learning, there are inputs but no supervising output; nevertheless we can learn relationships and struc ture from such data. To provide an illustration of some applications of statistical learning, we briefly discuss three real-world data sets that are considered in this book.

*Wage Data*

In this application (which we refer to as the Wage data set throughout this book), we examine a number of factors that relate to wages for a group of men from the Atlantic region of the United States. In particular, we wish to understand the association between an employee’s age and education, as well as the calendar year, on his wage. Consider, for example, the left-hand panel of Figure 1.1, which displays wage versus age for each of the individu als in the data set. There is evidence that wage increases with age but then decreases again after approximately age 60. The blue line, which provides an estimate of the average wage for a given age, makes this trend clearer.

1

© Springer Science+Business Media, LLC, part of Springer Nature 2021 G. James et al., An Introduction to Statistical Learning, Springer Texts in Statistics, https://doi.org/10.1007/978-1-0716-1418-1\_1

2 1. Introduction

0

0

3

0

0

e

2

g

a

W

0

0

1

0

5

20 40 60 80

Age

e

ga

W

0

0

3

0

0

2

0

0

1

0

5

2003 2006 2009 Year

e

ga

W

0

0

3

0

0

2

0

0

1

0

5

12345

Education Level

FIGURE 1.1. Wage *data, which contains income survey information for men from the central Atlantic region of the United States.* Left: wage *as a function of* age*. On average,* wage *increases with* age *until about* 60 *years of age, at which point it begins to decline.* Center: wage *as a function of* year*. There is a slow but steady increase of approximately* $10*,*000 *in the average* wage *between* 2003 *and* 2009*.* Right: *Boxplots displaying* wage *as a function of* education*, with* 1 *indicating the lowest level (no high school diploma) and* 5 *the highest level (an advanced graduate degree). On average,* wage *increases with the level of education.*

Given an employee’s age, we can use this curve to *predict* his wage. However, it is also clear from Figure 1.1 that there is a significant amount of vari ability associated with this average value, and so age alone is unlikely to provide an accurate prediction of a particular man’s wage.

We also have information regarding each employee’s education level and the year in which the wage was earned. The center and right-hand panels of Figure 1.1, which display wage as a function of both year and education, in dicate that both of these factors are associated with wage. Wages increase by approximately $10*,*000, in a roughly linear (or straight-line) fashion, between 2003 and 2009, though this rise is very slight relative to the vari ability in the data. Wages are also typically greater for individuals with higher education levels: men with the lowest education level (1) tend to have substantially lower wages than those with the highest education level (5). Clearly, the most accurate prediction of a given man’s wage will be obtained by combining his age, his education, and the year. In Chapter 3, we discuss linear regression, which can be used to predict wage from this data set. Ideally, we should predict wage in a way that accounts for the non-linear relationship between wage and age. In Chapter 7, we discuss a class of approaches for addressing this problem.

P &

S

n

i

e

gn

a

h

c

e

g

a

t

n

e

c

r

e

P

6 4 2

0

2−

4 −

**Yesterday**

Down Up Today’s Direction

P &

S

n

i

e

gn

a

h

c

e

g

a

t

n

e

c

r

e

P

6 4 2

0

2−

4 −

**Two Days Previous**

Down Up

Today’s Direction

P &

S

n

i

e

gn

a

h

c

e

g

a

t

n

e

c

r

e

P

1. Introduction 3

**Three Days Previous**

6

4

2

0

2

−

4

−

Down Up

Today’s Direction

FIGURE 1.2. Left: *Boxplots of the previous day’s percentage change in the S&P index for the days for which the market increased or decreased, obtained from the* Smarket *data.* Center and Right: *Same as left panel, but the percentage changes for 2 and 3 days previous are shown.*

*Stock Market Data*

The Wage data involves predicting a *continuous* or *quantitative* output value. This is often referred to as a *regression* problem. However, in certain cases we may instead wish to predict a non-numerical value—that is, a *categorical* or *qualitative* output. For example, in Chapter 4 we examine a stock market data set that contains the daily movements in the Standard & Poor’s 500 (S&P) stock index over a 5-year period between 2001 and 2005. We refer to this as the Smarket data. The goal is to predict whether the index will *increase* or *decrease* on a given day, using the past 5 days’ percentage changes in the index. Here the statistical learning problem does not involve predicting a numerical value. Instead it involves predicting whether a given day’s stock market performance will fall into the Up bucket or the Down bucket. This is known as a *classification* problem. A model that could accurately predict the direction in which the market will move would be very useful!

The left-hand panel of Figure 1.2 displays two boxplots of the previous day’s percentage changes in the stock index: one for the 648 days for which the market increased on the subsequent day, and one for the 602 days for which the market decreased. The two plots look almost identical, suggest ing that there is no simple strategy for using yesterday’s movement in the S&P to predict today’s returns. The remaining panels, which display box plots for the percentage changes 2 and 3 days previous to today, similarly indicate little association between past and present returns. Of course, this lack of pattern is to be expected: in the presence of strong correlations be tween successive days’ returns, one could adopt a simple trading strategy

4 1. Introduction

2

5

.

0

y

t

i

l

i

b

0

a

5

.

b

0

o

r

P

d

8

e

4

t

.

0

c

i

d

e

r

6

P

4

.

0

Down Up

Today’s Direction

FIGURE 1.3. *We fit a quadratic discriminant analysis model to the subset of the* Smarket *data corresponding to the 2001–2004 time period, and predicted the probability of a stock market decrease using the 2005 data. On average, the predicted probability of decrease is higher for the days in which the market does decrease. Based on these results, we are able to correctly predict the direction of movement in the market 60% of the time.*

to generate profits from the market. Nevertheless, in Chapter 4, we explore these data using several different statistical learning methods. Interestingly, there are hints of some weak trends in the data that suggest that, at least for this 5-year period, it is possible to correctly predict the direction of movement in the market approximately 60% of the time (Figure 1.3).

*Gene Expression Data*

The previous two applications illustrate data sets with both input and output variables. However, another important class of problems involves situations in which we only observe input variables, with no corresponding output. For example, in a marketing setting, we might have demographic information for a number of current or potential customers. We may wish to understand which types of customers are similar to each other by grouping individuals according to their observed characteristics. This is known as a *clustering* problem. Unlike in the previous examples, here we are not trying to predict an output variable.

We devote Chapter 12 to a discussion of statistical learning methods for problems in which no natural output variable is available. We consider the NCI60 data set, which consists of 6*,*830 gene expression measurements for each of 64 cancer cell lines. Instead of predicting a particular output variable, we are interested in determining whether there are groups, or clusters, among the cell lines based on their gene expression measurements. This is a difficult question to address, in part because there are thousands of gene expression measurements per cell line, making it hard to visualize the data.

2

*Z*

0

0

2

2

0

0

2

0

0

*Z*

2

2

−

−

0

0

4

4

−

−

0

0

6

6

−

−

−40 −20 0 20 40 60

1. Introduction 5

−40 −20 0 20 40 60

*Z*1 *Z*1

FIGURE 1.4. Left: *Representation of the* NCI60 *gene expression data set in a two-dimensional space, Z*1 *and Z*2*. Each point corresponds to one of the* 64 *cell lines. There appear to be four groups of cell lines, which we have represented using different colors.* Right: *Same as left panel except that we have represented each of the* 14 *different types of cancer using a different colored symbol. Cell lines corresponding to the same cancer type tend to be nearby in the two-dimensional space.*

The left-hand panel of Figure 1.4 addresses this problem by represent ing each of the 64 cell lines using just two numbers, *Z*1 and *Z*2. These are the first two *principal components* of the data, which summarize the 6*,*830 expression measurements for each cell line down to two numbers or *dimensions*. While it is likely that this dimension reduction has resulted in some loss of information, it is now possible to visually examine the data for evidence of clustering. Deciding on the number of clusters is often a difficult problem. But the left-hand panel of Figure 1.4 suggests at least four groups of cell lines, which we have represented using separate colors.

In this particular data set, it turns out that the cell lines correspond to 14 different types of cancer. (However, this information was not used to create the left-hand panel of Figure 1.4.) The right-hand panel of Fig ure 1.4 is identical to the left-hand panel, except that the 14 cancer types are shown using distinct colored symbols. There is clear evidence that cell lines with the same cancer type tend to be located near each other in this two-dimensional representation. In addition, even though the cancer infor mation was not used to produce the left-hand panel, the clustering obtained does bear some resemblance to some of the actual cancer types observed in the right-hand panel. This provides some independent verification of the accuracy of our clustering analysis.

6 1. Introduction

A Brief History of Statistical Learning

Though the term *statistical learning* is fairly new, many of the concepts that underlie the field were developed long ago. At the beginning of the nine teenth century, the method of *least squares* was developed, implementing the earliest form of what is now known as *linear regression*. The approach was first successfully applied to problems in astronomy. Linear regression is used for predicting quantitative values, such as an individual’s salary. In order to predict qualitative values, such as whether a patient survives or dies, or whether the stock market increases or decreases, *linear discrim inant analysis* was proposed in 1936. In the 1940s, various authors put forth an alternative approach, *logistic regression*. In the early 1970s, the term *generalized linear model* was developed to describe an entire class of statistical learning methods that include both linear and logistic regression as special cases.

By the end of the 1970s, many more techniques for learning from data were available. However, they were almost exclusively *linear* methods be cause fitting *non-linear* relationships was computationally difficult at the time. By the 1980s, computing technology had finally improved sufficiently that non-linear methods were no longer computationally prohibitive. In the mid 1980s, *classification and regression trees* were developed, followed shortly by *generalized additive models*. *Neural networks* gained popularity in the 1980s, and *support vector machines* arose in the 1990s.

Since that time, statistical learning has emerged as a new subfield in statistics, focused on supervised and unsupervised modeling and prediction. In recent years, progress in statistical learning has been marked by the increasing availability of powerful and relatively user-friendly software, such as the popular and freely available R system. This has the potential to continue the transformation of the field from a set of techniques used and developed by statisticians and computer scientists to an essential toolkit for a much broader community.

This Book

*The Elements of Statistical Learning* (ESL) by Hastie, Tibshirani, and Friedman was first published in 2001. Since that time, it has become an important reference on the fundamentals of statistical machine learning. Its success derives from its comprehensive and detailed treatment of many important topics in statistical learning, as well as the fact that (relative to many upper-level statistics textbooks) it is accessible to a wide audience. However, the greatest factor behind the success of ESL has been its topical nature. At the time of its publication, interest in the field of statistical

1. Introduction 7

learning was starting to explode. ESL provided one of the first accessible and comprehensive introductions to the topic.

Since ESL was first published, the field of statistical learning has con tinued to flourish. The field’s expansion has taken two forms. The most obvious growth has involved the development of new and improved statis tical learning approaches aimed at answering a range of scientific questions across a number of fields. However, the field of statistical learning has also expanded its audience. In the 1990s, increases in computational power generated a surge of interest in the field from non-statisticians who were eager to use cutting-edge statistical tools to analyze their data. Unfortu nately, the highly technical nature of these approaches meant that the user community remained primarily restricted to experts in statistics, computer science, and related fields with the training (and time) to understand and implement them.

In recent years, new and improved software packages have significantly eased the implementation burden for many statistical learning methods. At the same time, there has been growing recognition across a number of fields, from business to health care to genetics to the social sciences and beyond, that statistical learning is a powerful tool with important practical applications. As a result, the field has moved from one of primarily academic interest to a mainstream discipline, with an enormous potential audience. This trend will surely continue with the increasing availability of enormous quantities of data and the software to analyze it.

The purpose of *An Introduction to Statistical Learning* (ISL) is to facili tate the transition of statistical learning from an academic to a mainstream field. ISL is not intended to replace ESL, which is a far more comprehen sive text both in terms of the number of approaches considered and the depth to which they are explored. We consider ESL to be an important companion for professionals (with graduate degrees in statistics, machine learning, or related fields) who need to understand the technical details behind statistical learning approaches. However, the community of users of statistical learning techniques has expanded to include individuals with a wider range of interests and backgrounds. Therefore, there is a place for a less technical and more accessible version of ESL.

In teaching these topics over the years, we have discovered that they are of interest to master’s and PhD students in fields as disparate as business administration, biology, and computer science, as well as to quantitatively oriented upper-division undergraduates. It is important for this diverse group to be able to understand the models, intuitions, and strengths and weaknesses of the various approaches. But for this audience, many of the technical details behind statistical learning methods, such as optimiza tion algorithms and theoretical properties, are not of primary interest. We believe that these students do not need a deep understanding of these aspects in order to become informed users of the various methodologies, and

8 1. Introduction

in order to contribute to their chosen fields through the use of statistical learning tools.

ISL is based on the following four premises.

1. *Many statistical learning methods are relevant and useful in a wide range of academic and non-academic disciplines, beyond just the sta tistical sciences.* We believe that many contemporary statistical learn ing procedures should, and will, become as widely available and used as is currently the case for classical methods such as linear regres sion. As a result, rather than attempting to consider every possible approach (an impossible task), we have concentrated on presenting the methods that we believe are most widely applicable.

2. *Statistical learning should not be viewed as a series of black boxes.* No single approach will perform well in all possible applications. With out understanding all of the cogs inside the box, or the interaction between those cogs, it is impossible to select the best box. Hence, we have attempted to carefully describe the model, intuition, assump tions, and trade-offs behind each of the methods that we consider.

3. *While it is important to know what job is performed by each cog, it is not necessary to have the skills to construct the machine inside the box!* Thus, we have minimized discussion of technical details related to fitting procedures and theoretical properties. We assume that the reader is comfortable with basic mathematical concepts, but we do not assume a graduate degree in the mathematical sciences. For in stance, we have almost completely avoided the use of matrix algebra, and it is possible to understand the entire book without a detailed knowledge of matrices and vectors.

4. *We presume that the reader is interested in applying statistical learn ing methods to real-world problems.* In order to facilitate this, as well as to motivate the techniques discussed, we have devoted a section within each chapter to R computer labs. In each lab, we walk the reader through a realistic application of the methods considered in that chapter. When we have taught this material in our courses, we have allocated roughly one-third of classroom time to working through the labs, and we have found them to be extremely useful. Many of the less computationally-oriented students who were ini tially intimidated by R’s command level interface got the hang of things over the course of the quarter or semester. We have used R because it is freely available and is powerful enough to implement all of the methods discussed in the book. It also has optional packages that can be downloaded to implement literally thousands of addi tional methods. Most importantly, R is the language of choice for academic statisticians, and new approaches often become available in

1. Introduction 9

R years before they are implemented in commercial packages. How ever, the labs in ISL are self-contained, and can be skipped if the reader wishes to use a different software package or does not wish to apply the methods discussed to real-world problems.

Who Should Read This Book?

This book is intended for anyone who is interested in using modern statis tical methods for modeling and prediction from data. This group includes scientists, engineers, data analysts, data scientists, and quants, but also less technical individuals with degrees in non-quantitative fields such as the social sciences or business. We expect that the reader will have had at least one elementary course in statistics. Background in linear regression is also useful, though not required, since we review the key concepts behind linear regression in Chapter 3. The mathematical level of this book is mod est, and a detailed knowledge of matrix operations is not required. This book provides an introduction to the statistical programming language R. Previous exposure to a programming language, such as MATLAB or Python, is useful but not required.

The first edition of this textbook has been used as to teach master’s and PhD students in business, economics, computer science, biology, earth sci ences, psychology, and many other areas of the physical and social sciences. It has also been used to teach advanced undergraduates who have already taken a course on linear regression. In the context of a more mathemat ically rigorous course in which ESL serves as the primary textbook, ISL could be used as a supplementary text for teaching computational aspects of the various approaches.

Notation and Simple Matrix Algebra

Choosing notation for a textbook is always a difficult task. For the most part we adopt the same notational conventions as ESL.

We will use *n* to represent the number of distinct data points, or observa tions, in our sample. We will let *p* denote the number of variables that are available for use in making predictions. For example, the Wage data set con sists of 11 variables for 3*,*000 people, so we have *n* = 3*,*000 observations and *p* = 11 variables (such as year, age, race, and more). Note that throughout this book, we indicate variable names using colored font: Variable Name.

In some examples, *p* might be quite large, such as on the order of thou sands or even millions; this situation arises quite often, for example, in the analysis of modern biological data or web-based advertising data.

10 1. Introduction

In general, we will let *xij* represent the value of the *j*th variable for the *i*th observation, where *i* = 1*,* 2*,...,n* and *j* = 1*,* 2*,...,p*. Throughout this book, *i* will be used to index the samples or observations (from 1 to *n*) and *j* will be used to index the variables (from 1 to *p*). We let X denote an *n × p* matrix whose (*i, j*)th element is *xij* . That is,

X =

⎛

⎜⎜⎜⎝*x*11 *x*12 *... x*1*p*

*x*21 *x*22 *... x*2*p* ... ... ... ... *xn*1 *xn*2 *... xnp*

⎞

⎟⎟⎟⎠ *.*

For readers who are unfamiliar with matrices, it is useful to visualize X as a spreadsheet of numbers with *n* rows and *p* columns.

At times we will be interested in the rows of X, which we write as *x*1*, x*2*,...,xn*. Here *xi* is a vector of length *p*, containing the *p* variable measurements for the *i*th observation. That is,

*xi* =

⎛

⎜⎜⎜⎝*xi*1

*xi*2...

*xip*

⎞

⎟⎟⎟⎠ *.* (1.1)

(Vectors are by default represented as columns.) For example, for the Wage data, *xi* is a vector of length 11, consisting of year, age, race, and other values for the *i*th individual. At other times we will instead be interested in the columns of X, which we write as x1*,* x2*,...,* x*p*. Each is a vector of

length *n*. That is,

x*j* =

⎛

⎜⎜⎜⎝*x*1*j*

*x*2*j*...

*xnj*

⎞

⎟⎟⎟⎠ *.*

For example, for the Wage data, x1 contains the *n* = 3*,*000 values for year. Using this notation, the matrix X can be written as

X = 'x1 x2 *···* x*p*(*,*

or

X =

⎛

⎜⎜⎜⎝*xT*1

*xT*2...

*xTn*

⎞

⎟⎟⎟⎠ *.*

1. Introduction 11

The *T* notation denotes the *transpose* of a matrix or vector. So, for example,

X*T* =

⎛

⎜⎜⎜⎝*x*11 *x*21 *... xn*1

*x*12 *x*22 *... xn*2 ... ... ... *x*1*p x*2*p ... xnp*

⎞

⎟⎟⎟⎠ *,*

while

*xTi* = '*xi*1 *xi*2 *··· xip*(*.*

We use *yi* to denote the *i*th observation of the variable on which we wish to make predictions, such as wage. Hence, we write the set of all *n* observations in vector form as

y =

⎛

⎜⎜⎜⎝*y*1

*y*2...

*yn*

⎞

⎟⎟⎟⎠ *.*

Then our observed data consists of *{*(*x*1*, y*1)*,*(*x*2*, y*2)*,...,*(*xn, yn*)*}*, where each *xi* is a vector of length *p*. (If *p* = 1, then *xi* is simply a scalar.) In this text, a vector of length *n* will always be denoted in *lower case*

*bold*; e.g.

a =

⎛

⎜⎜⎜⎝*a*1

*a*2...

*an*

⎞

⎟⎟⎟⎠ *.*

However, vectors that are not of length *n* (such as feature vectors of length *p*, as in (1.1)) will be denoted in *lower case normal font*, e.g. *a*. Scalars will also be denoted in *lower case normal font*, e.g. *a*. In the rare cases in which these two uses for lower case normal font lead to ambiguity, we will clarify which use is intended. Matrices will be denoted using *bold capitals*, such as A. Random variables will be denoted using *capital normal font*, e.g. *A*, regardless of their dimensions.

Occasionally we will want to indicate the dimension of a particular ob ject. To indicate that an object is a scalar, we will use the notation *a ∈* R. To indicate that it is a vector of length *k*, we will use *a ∈* R*k* (or a *∈* R*n* if it is of length *n*). We will indicate that an object is an *r × s* matrix using A *∈* R*r×s*.

We have avoided using matrix algebra whenever possible. However, in a few instances it becomes too cumbersome to avoid it entirely. In these rare instances it is important to understand the concept of multiplying two matrices. Suppose that A *∈* R*r×d* and B *∈* R*d×s*. Then the product of A and B is denoted AB. The (*i, j*)th element of AB is computed by

12 1. Introduction

multiplying each element of the *i*th row of A by the corresponding element of the *j*th column of B. That is, (AB)*ij* = )*dk*=1 *aikbkj* . As an example,

consider

A =

Then

\*1 2 3 4

+

and B =

\*5 6 7 8

+

*.*

AB =

\*1 2 3 4

+ \*5 6 7 8

+

=

\*1 *×* 5+2 *×* 7 1 *×* 6+2 *×* 8 3 *×* 5+4 *×* 7 3 *×* 6+4 *×* 8

+

=

\*19 22 43 50

+

*.*

Note that this operation produces an *r × s* matrix. It is only possible to compute AB if the number of columns of A is the same as the number of rows of B.

Organization of This Book

Chapter 2 introduces the basic terminology and concepts behind statisti cal learning. This chapter also presents the *K-nearest neighbor* classifier, a very simple method that works surprisingly well on many problems. Chap ters 3 and 4 cover classical linear methods for regression and classification. In particular, Chapter 3 reviews *linear regression*, the fundamental start ing point for all regression methods. In Chapter 4 we discuss two of the most important classical classification methods, *logistic regression* and *lin ear discriminant analysis*.

A central problem in all statistical learning situations involves choosing the best method for a given application. Hence, in Chapter 5 we intro duce *cross-validation* and the *bootstrap*, which can be used to estimate the accuracy of a number of different methods in order to choose the best one.

Much of the recent research in statistical learning has concentrated on non-linear methods. However, linear methods often have advantages over their non-linear competitors in terms of interpretability and sometimes also accuracy. Hence, in Chapter 6 we consider a host of linear methods, both classical and more modern, which offer potential improvements over stan dard linear regression. These include *stepwise selection*, *ridge regression*, *principal components regression*, and the *lasso*.

The remaining chapters move into the world of non-linear statistical learning. We first introduce in Chapter 7 a number of non-linear meth ods that work well for problems with a single input variable. We then show how these methods can be used to fit non-linear *additive* models for which there is more than one input. In Chapter 8, we investigate *tree*-based methods, including *bagging*, *boosting*, and *random forests*. *Support vector machines*, a set of approaches for performing both linear and non-linear classification, are discussed in Chapter 9. We cover *deep learning*, an ap proach for non-linear regression and classification that has received a lot

1. Introduction 13

of attention in recent years, in Chapter 10. Chapter 11 explores *survival analysis*, a regression approach that is specialized to the setting in which the output variable is *censored*, i.e. not fully observed.

In Chapter 12, we consider the *unsupervised* setting in which we have input variables but no output variable. In particular, we present *princi pal components analysis*, *K-means clustering*, and *hierarchical clustering*. Finally, in Chapter 13 we cover the very important topic of multiple hy pothesis testing.

At the end of each chapter, we present one or more R lab sections in which we systematically work through applications of the various meth ods discussed in that chapter. These labs demonstrate the strengths and weaknesses of the various approaches, and also provide a useful reference for the syntax required to implement the various methods. The reader may choose to work through the labs at his or her own pace, or the labs may be the focus of group sessions as part of a classroom environment. Within each R lab, we present the results that we obtained when we performed the lab at the time of writing this book. However, new versions of R are continuously released, and over time, the packages called in the labs will be updated. Therefore, in the future, it is possible that the results shown in the lab sections may no longer correspond precisely to the results obtained by the reader who performs the labs. As necessary, we will post updates to the labs on the book website. 

We use the symbol to denote sections or exercises that contain more challenging concepts. These can be easily skipped by readers who do not wish to delve as deeply into the material, or who lack the mathematical background.

Data Sets Used in Labs and Exercises

In this textbook, we illustrate statistical learning methods using applica tions from marketing, finance, biology, and other areas. The ISLR2 package available on the book website and CRAN contains a number of data sets that are required in order to perform the labs and exercises associated with this book. One other data set is part of the base R distribution. Table 1.1 contains a summary of the data sets required to perform the labs and ex ercises. A couple of these data sets are also available as text files on the book website, for use in Chapter 2.

14 1. Introduction

Name Description

Auto Gas mileage, horsepower, and other information for cars. Bikeshare Hourly usage of a bike sharing program in Washington, DC. Boston Housing values and other information about Boston census tracts. BrainCancer Survival times for patients diagnosed with brain cancer. Caravan Information about individuals offered caravan insurance. Carseats Information about car seat sales in 400 stores.

College Demographic characteristics, tuition, and more for USA colleges. Credit Information about credit card debt for 10,000 customers. Default Customer default records for a credit card company. Fund Returns of 2,000 hedge fund managers over 50 months. Hitters Records and salaries for baseball players.

Khan Gene expression measurements for four cancer types. NCI60 Gene expression measurements for 64 cancer cell lines. NYSE Returns, volatility, and volume for the New York Stock Exchange. OJ Sales information for Citrus Hill and Minute Maid orange juice. Portfolio Past values of financial assets, for use in portfolio allocation. Publication Time to publication for 244 clinical trials.

Smarket Daily percentage returns for S&P 500 over a 5-year period. USArrests Crime statistics per 100,000 residents in 50 states of USA. Wage Income survey data for men in central Atlantic region of USA. Weekly 1,089 weekly stock market returns for 21 years.

TABLE 1.1. *A list of data sets needed to perform the labs and exercises in this textbook. All data sets are available in the* ISLR2 *library, with the exception of* USArrests*, which is part of the base* R *distribution*.

Book Website

The website for this book is located at

www.statlearning.com

It contains a number of resources, including the R package associated with this book, and some additional data sets.

Acknowledgements

A few of the plots in this book were taken from ESL: Figures 6.7, 8.3, and 12.14. All other plots are new to this book.

2

Statistical Learning 2.1 What Is Statistical Learning?



In order to motivate our study of statistical learning, we begin with a simple example. Suppose that we are statistical consultants hired by a client to investigate the association between advertising and sales of a particular product. The Advertising data set consists of the sales of that product in 200 different markets, along with advertising budgets for the product in each of those markets for three different media: TV, radio, and newspaper. The data are displayed in Figure 2.1. It is not possible for our client to directly increase sales of the product. On the other hand, they can control the advertising expenditure in each of the three media. Therefore, if we determine that there is an association between advertising and sales, then we can instruct our client to adjust advertising budgets, thereby indirectly increasing sales. In other words, our goal is to develop an accurate model that can be used to predict sales on the basis of the three media budgets.

In this setting, the advertising budgets are *input variables* while sales input variable is an *output variable*. The input variables are typically denoted using the

symbol *X*, with a subscript to distinguish them. So *X*1 might be the TV budget, *X*2 the radio budget, and *X*3 the newspaper budget. The inputs

output variable

go by different names, such as *predictors*, *independent variables*, *features*, predictor

or sometimes just *variables*. The output variable—in this case, sales—is often called the *response* or *dependent variable*, and is typically denoted using the symbol *Y* . Throughout this book, we will use all of these terms interchangeably.

independent variable

feature

variable

response

dependent

15

© Springer Science+Business Media, LLC, part of Springer Nature 2021 G. James et al., An Introduction to Statistical Learning, Springer Texts in Statistics, https://doi.org/10.1007/978-1-0716-1418-1\_2

variable

16 2. Statistical Learning

s

e

la

S

5

5

2

2

0

0

2

2

s

5

5

1

1

e

l

a

S

0

0

1

1

5

5

0 50 100 200 300

TV

5

2

0

2

s

5

1

e

l

a

S

0

1

5

0 10 20 30 40 50

Radio

0 20 40 60 80 100 Newspaper

FIGURE 2.1. *The* Advertising *data set. The plot displays* sales*, in thousands of units, as a function of* TV*,* radio*, and* newspaper *budgets, in thousands of dollars, for* 200 *different markets. In each plot we show the simple least squares fit of* sales *to that variable, as described in Chapter 3. In other words, each blue line represents a simple model that can be used to predict* sales *using* TV*,* radio*, and* newspaper*, respectively.*

More generally, suppose that we observe a quantitative response *Y* and *p* different predictors, *X*1*, X*2*,...,Xp*. We assume that there is some relationship between *Y* and *X* = (*X*1*, X*2*,...,Xp*), which can be written in the very general form

*Y* = *f*(*X*) + *ϵ.* (2.1)

Here *f* is some fixed but unknown function of *X*1*,...,Xp*, and *ϵ* is a random *error term*, which is independent of *X* and has mean zero. In this formula- error term tion, *f* represents the *systematic* information that *X* provides about *Y* . systematic As another example, consider the left-hand panel of Figure 2.2, a plot of income versus years of education for 30 individuals in the Income data set. The plot suggests that one might be able to predict income using years of education. However, the function *f* that connects the input variable to the output variable is in general unknown. In this situation one must estimate *f* based on the observed points. Since Income is a simulated data set, *f* is known and is shown by the blue curve in the right-hand panel of Figure 2.2. The vertical lines represent the error terms *ϵ*. We note that some of the 30 observations lie above the blue curve and some lie below it; overall, the errors have approximately mean zero.

In general, the function *f* may involve more than one input variable. In Figure 2.3 we plot income as a function of years of education and seniority. Here *f* is a two-dimensional surface that must be estimated based on the observed data.

e

m

o

c

n

I

0

8

0

7

0

6

0

5

0

4

0

3

0

2

10 12 14 16 18 20 22 Years of Education

2.1 What Is Statistical Learning? 17

0

8

0

7

0

6

e

0

m

5

o

c

n

I

0

4

0

3

0

2

10 12 14 16 18 20 22

Years of Education

FIGURE 2.2. *The* Income *data set.* Left: *The red dots are the observed values of* income *(in tens of thousands of dollars) and* years of education *for* 30 *indi viduals.* Right: *The blue curve represents the true underlying relationship between* income *and* years of education*, which is generally unknown (but is known in this case because the data were simulated). The black lines represent the error associated with each observation. Note that some errors are positive (if an ob servation lies above the blue curve) and some are negative (if an observation lies below the curve). Overall, these errors have approximately mean zero.*

In essence, statistical learning refers to a set of approaches for estimating *f*. In this chapter we outline some of the key theoretical concepts that arise in estimating *f*, as well as tools for evaluating the estimates obtained.

*2.1.1 Why Estimate f?*

There are two main reasons that we may wish to estimate *f*: *prediction* and *inference*. We discuss each in turn.

Prediction

In many situations, a set of inputs *X* are readily available, but the output *Y* cannot be easily obtained. In this setting, since the error term averages to zero, we can predict *Y* using

*Y*ˆ = ˆ*f*(*X*)*,* (2.2)

where ˆ*f* represents our estimate for *f*, and *Y*ˆ represents the resulting pre diction for *Y* . In this setting, ˆ*f* is often treated as a *black box*, in the sense that one is not typically concerned with the exact form of ˆ*f*, provided that it yields accurate predictions for *Y* .

As an example, suppose that *X*1*,...,Xp* are characteristics of a patient’s blood sample that can be easily measured in a lab, and *Y* is a variable encoding the patient’s risk for a severe adverse reaction to a particular

18 2. Statistical Learning

I

n

c

o

m

e

Years of EducationSeniority

FIGURE 2.3. *The plot displays* income *as a function of* years of education *and* seniority *in the* Income *data set. The blue surface represents the true un derlying relationship between* income *and* years of education *and* seniority*, which is known since the data are simulated. The red dots indicate the observed values of these quantities for* 30 *individuals.*

drug. It is natural to seek to predict *Y* using *X*, since we can then avoid giving the drug in question to patients who are at high risk of an adverse reaction—that is, patients for whom the estimate of *Y* is high.

The accuracy of *Y*ˆ as a prediction for *Y* depends on two quantities, which we will call the *reducible error* and the *irreducible error*. In general, reducible

ˆ*f* will not be a perfect estimate for *f*, and this inaccuracy will introduce some error. This error is *reducible* because we can potentially improve the accuracy of ˆ*f* by using the most appropriate statistical learning technique to estimate *f*. However, even if it were possible to form a perfect estimate for *f*, so that our estimated response took the form *Y*ˆ = *f*(*X*), our prediction would still have some error in it! This is because *Y* is also a function of *ϵ*, which, by definition, cannot be predicted using *X*. Therefore, variability associated with *ϵ* also affects the accuracy of our predictions. This is known as the *irreducible* error, because no matter how well we estimate *f*, we cannot reduce the error introduced by *ϵ*.

Why is the irreducible error larger than zero? The quantity *ϵ* may con tain unmeasured variables that are useful in predicting *Y* : since we don’t measure them, *f* cannot use them for its prediction. The quantity *ϵ* may also contain unmeasurable variation. For example, the risk of an adverse reaction might vary for a given patient on a given day, depending on manufacturing variation in the drug itself or the patient’s general feeling of well-being on that day.

error

irreducible error

2.1 What Is Statistical Learning? 19

Consider a given estimate ˆ*f* and a set of predictors *X*, which yields the prediction *Y*ˆ = ˆ*f*(*X*). Assume for a moment that both ˆ*f* and *X* are fixed, so that the only variability comes from *ϵ*. Then, it is easy to show that

E(*Y − Y*ˆ )2 = E[*f*(*X*) + *ϵ −* ˆ*f*(*X*)]2

= [*f*(*X*) *−* ˆ*f*(*X*)]2 , -. / Reducible

+ Var(*ϵ*) , -. / Irreducible

*,* (2.3)

where E(*Y − Y*ˆ )2 represents the average, or *expected value*, of the squared expected value difference between the predicted and actual value of *Y* , and Var(*ϵ*) repre sents the *variance* associated with the error term *ϵ*. variance The focus of this book is on techniques for estimating *f* with the aim of minimizing the reducible error. It is important to keep in mind that the irreducible error will always provide an upper bound on the accuracy of our prediction for *Y* . This bound is almost always unknown in practice.

Inference

We are often interested in understanding the association between *Y* and *X*1*,...,Xp*. In this situation we wish to estimate *f*, but our goal is not necessarily to make predictions for *Y* . Now ˆ*f* cannot be treated as a black box, because we need to know its exact form. In this setting, one may be interested in answering the following questions:

*• Which predictors are associated with the response?* It is often the case that only a small fraction of the available predictors are substantially associated with *Y* . Identifying the few *important* predictors among a large set of possible variables can be extremely useful, depending on the application.

*• What is the relationship between the response and each predictor?* Some predictors may have a positive relationship with *Y* , in the sense that larger values of the predictor are associated with larger values of *Y* . Other predictors may have the opposite relationship. Depending on the complexity of *f*, the relationship between the response and a given predictor may also depend on the values of the other predictors.

*• Can the relationship between Y and each predictor be adequately sum marized using a linear equation, or is the relationship more compli cated?* Historically, most methods for estimating *f* have taken a linear form. In some situations, such an assumption is reasonable or even de sirable. But often the true relationship is more complicated, in which case a linear model may not provide an accurate representation of the relationship between the input and output variables.

In this book, we will see a number of examples that fall into the prediction setting, the inference setting, or a combination of the two.

20 2. Statistical Learning

For instance, consider a company that is interested in conducting a direct-marketing campaign. The goal is to identify individuals who are likely to respond positively to a mailing, based on observations of demo graphic variables measured on each individual. In this case, the demo graphic variables serve as predictors, and response to the marketing cam paign (either positive or negative) serves as the outcome. The company is not interested in obtaining a deep understanding of the relationships be tween each individual predictor and the response; instead, the company simply wants to accurately predict the response using the predictors. This is an example of modeling for prediction.

In contrast, consider the Advertising data illustrated in Figure 2.1. One may be interested in answering questions such as:

– *Which media are associated with sales?*

– *Which media generate the biggest boost in sales?* or

– *How large of an increase in sales is associated with a given increase in TV advertising?*

This situation falls into the inference paradigm. Another example involves modeling the brand of a product that a customer might purchase based on variables such as price, store location, discount levels, competition price, and so forth. In this situation one might really be most interested in the association between each variable and the probability of purchase. For in stance, *to what extent is the product’s price associated with sales?* This is an example of modeling for inference.

Finally, some modeling could be conducted both for prediction and in ference. For example, in a real estate setting, one may seek to relate values of homes to inputs such as crime rate, zoning, distance from a river, air quality, schools, income level of community, size of houses, and so forth. In this case one might be interested in the association between each individ ual input variable and housing price—for instance, *how much extra will a house be worth if it has a view of the river?* This is an inference problem. Alternatively, one may simply be interested in predicting the value of a home given its characteristics: *is this house under- or over-valued?* This is a prediction problem.

Depending on whether our ultimate goal is prediction, inference, or a combination of the two, different methods for estimating *f* may be appro priate. For example, *linear models* allow for relatively simple and inter- linear model pretable inference, but may not yield as accurate predictions as some other approaches. In contrast, some of the highly non-linear approaches that we discuss in the later chapters of this book can potentially provide quite accu rate predictions for *Y* , but this comes at the expense of a less interpretable model for which inference is more challenging.

2.1 What Is Statistical Learning? 21

*2.1.2 How Do We Estimate f?*

Throughout this book, we explore many linear and non-linear approaches for estimating *f*. However, these methods generally share certain charac teristics. We provide an overview of these shared characteristics in this section. We will always assume that we have observed a set of *n* different data points. For example in Figure 2.2 we observed *n* = 30 data points. These observations are called the *training data* because we will use these training

data observations to train, or teach, our method how to estimate *f*. Let *xij* represent the value of the *j*th predictor, or input, for observation *i*, where *i* = 1*,* 2*,...,n* and *j* = 1*,* 2*,...,p*. Correspondingly, let *yi* represent the response variable for the *i*th observation. Then our training data consist of *{*(*x*1*, y*1)*,*(*x*2*, y*2)*,...,*(*xn, yn*)*}* where *xi* = (*xi*1*, xi*2*,...,xip*)*T* .

Our goal is to apply a statistical learning method to the training data in order to estimate the unknown function *f*. In other words, we want to find a function ˆ*f* such that *Y ≈* ˆ*f*(*X*) for any observation (*X, Y* ). Broadly speaking, most statistical learning methods for this task can be character ized as either *parametric* or *non-parametric*. We now briefly discuss these parametric

two types of approaches.

Parametric Methods

Parametric methods involve a two-step model-based approach.

1. First, we make an assumption about the functional form, or shape, of *f*. For example, one very simple assumption is that *f* is linear in *X*:

*f*(*X*) = *β*0 + *β*1*X*1 + *β*2*X*2 + *···* + *βpXp.* (2.4)

This is a *linear model*, which will be discussed extensively in Chap ter 3. Once we have assumed that *f* is linear, the problem of estimat ing *f* is greatly simplified. Instead of having to estimate an entirely arbitrary *p*-dimensional function *f*(*X*), one only needs to estimate the *p* + 1 coefficients *β*0*, β*1*,..., βp*.

2. After a model has been selected, we need a procedure that uses the

non

parametric

training data to *fit* or *train* the model. In the case of the linear model fit train (2.4), we need to estimate the parameters *β*0*, β*1*,..., βp*. That is, we want to find values of these parameters such that

*Y ≈ β*0 + *β*1*X*1 + *β*2*X*2 + *···* + *βpXp.*

The most common approach to fitting the model (2.4) is referred to as *(ordinary) least squares*, which we discuss in Chapter 3. However, least squares least squares is one of many possible ways to fit the linear model. In Chapter 6, we discuss other approaches for estimating the parameters in (2.4).

22 2. Statistical Learning

I

n

c

o

m

e

Years of EducationSeniority

FIGURE 2.4. *A linear model fit by least squares to the* Income *data from Fig ure 2.3. The observations are shown in red, and the yellow plane indicates the least squares fit to the data.*

The model-based approach just described is referred to as *parametric*; it reduces the problem of estimating *f* down to one of estimating a set of parameters. Assuming a parametric form for *f* simplifies the problem of estimating *f* because it is generally much easier to estimate a set of pa rameters, such as *β*0*, β*1*,..., βp* in the linear model (2.4), than it is to fit an entirely arbitrary function *f*. The potential disadvantage of a paramet ric approach is that the model we choose will usually not match the true unknown form of *f*. If the chosen model is too far from the true *f*, then our estimate will be poor. We can try to address this problem by choos ing *flexible* models that can fit many different possible functional forms flexible for *f*. But in general, fitting a more flexible model requires estimating a

greater number of parameters. These more complex models can lead to a phenomenon known as *overfitting* the data, which essentially means they overfitting follow the errors, or *noise*, too closely. These issues are discussed through- noise out this book. Figure 2.4 shows an example of the parametric approach applied to the Income data from Figure 2.3. We have fit a linear model of the form

income *≈ β*0 + *β*1 *×* education + *β*2 *×* seniority*.*

Since we have assumed a linear relationship between the response and the two predictors, the entire fitting problem reduces to estimating *β*0, *β*1, and *β*2, which we do using least squares linear regression. Comparing Figure 2.3 to Figure 2.4, we can see that the linear fit given in Figure 2.4 is not quite right: the true *f* has some curvature that is not captured in the linear fit. However, the linear fit still appears to do a reasonable job of capturing the

2.1 What Is Statistical Learning? 23

I

n

c

o

m

e

Years of EducationSeniority

FIGURE 2.5. *A smooth thin-plate spline fit to the* Income *data from Figure 2.3 is shown in yellow; the observations are displayed in red. Splines are discussed in Chapter 7.*

positive relationship between years of education and income, as well as the slightly less positive relationship between seniority and income. It may be that with such a small number of observations, this is the best we can do.

Non-Parametric Methods

Non-parametric methods do not make explicit assumptions about the func tional form of *f*. Instead they seek an estimate of *f* that gets as close to the data points as possible without being too rough or wiggly. Such approaches can have a major advantage over parametric approaches: by avoiding the assumption of a particular functional form for *f*, they have the potential to accurately fit a wider range of possible shapes for *f*. Any parametric approach brings with it the possibility that the functional form used to estimate *f* is very different from the true *f*, in which case the resulting model will not fit the data well. In contrast, non-parametric approaches completely avoid this danger, since essentially no assumption about the form of *f* is made. But non-parametric approaches do suffer from a major disadvantage: since they do not reduce the problem of estimating *f* to a small number of parameters, a very large number of observations (far more than is typically needed for a parametric approach) is required in order to obtain an accurate estimate for *f*.

An example of a non-parametric approach to fitting the Income data is shown in Figure 2.5. A *thin-plate spline* is used to estimate *f*. This ap- thin-plate spline proach does not impose any pre-specified model on *f*. It instead attempts to produce an estimate for *f* that is as close as possible to the observed data, subject to the fit—that is, the yellow surface in Figure 2.5—being

24 2. Statistical Learning

I

n

c

o

m

e

Years of EducationSeniority

FIGURE 2.6. *A rough thin-plate spline fit to the* Income *data from Figure 2.3. This fit makes zero errors on the training data.*

*smooth*. In this case, the non-parametric fit has produced a remarkably ac curate estimate of the true *f* shown in Figure 2.3. In order to fit a thin-plate spline, the data analyst must select a level of smoothness. Figure 2.6 shows the same thin-plate spline fit using a lower level of smoothness, allowing for a rougher fit. The resulting estimate fits the observed data perfectly! However, the spline fit shown in Figure 2.6 is far more variable than the true function *f*, from Figure 2.3. This is an example of overfitting the data, which we discussed previously. It is an undesirable situation because the fit obtained will not yield accurate estimates of the response on new observations that were not part of the original training data set. We dis cuss methods for choosing the *correct* amount of smoothness in Chapter 5. Splines are discussed in Chapter 7.

As we have seen, there are advantages and disadvantages to parametric and non-parametric methods for statistical learning. We explore both types of methods throughout this book.

*2.1.3 The Trade-Off Between Prediction Accuracy and Model Interpretability*

Of the many methods that we examine in this book, some are less flexible, or more restrictive, in the sense that they can produce just a relatively small range of shapes to estimate *f*. For example, linear regression is a relatively inflexible approach, because it can only generate linear functions such as the lines shown in Figure 2.1 or the plane shown in Figure 2.4. Other methods, such as the thin plate splines shown in Figures 2.5 and 2.6,

2.1 What Is Statistical Learning? 25

hSubset Selection

y

t

il

ib

a

t

e

r

pr

e

t

n

I

Lasso

g

i

H

Least Squares

Generalized Additive Models

Trees

Bagging, Boosting

Support Vector Machines

Deep Learning

w

o

L

Low High Flexibility

FIGURE 2.7. *A representation of the tradeoff between flexibility and inter pretability, using different statistical learning methods. In general, as the flexibil ity of a method increases, its interpretability decreases.*

are considerably more flexible because they can generate a much wider range of possible shapes to estimate *f*.

One might reasonably ask the following question: *why would we ever choose to use a more restrictive method instead of a very flexible approach?* There are several reasons that we might prefer a more restrictive model. If we are mainly interested in inference, then restrictive models are much more interpretable. For instance, when inference is the goal, the linear model may be a good choice since it will be quite easy to understand the relationship between *Y* and *X*1*, X*2*,...,Xp*. In contrast, very flexible approaches, such as the splines discussed in Chapter 7 and displayed in Figures 2.5 and 2.6, and the boosting methods discussed in Chapter 8, can lead to such complicated estimates of *f* that it is difficult to understand how any individual predictor is associated with the response.

Figure 2.7 provides an illustration of the trade-off between flexibility and interpretability for some of the methods that we cover in this book. Least squares linear regression, discussed in Chapter 3, is relatively inflexible but is quite interpretable. The *lasso*, discussed in Chapter 6, relies upon the lasso linear model (2.4) but uses an alternative fitting procedure for estimating

the coefficients *β*0*, β*1*,..., βp*. The new procedure is more restrictive in es timating the coefficients, and sets a number of them to exactly zero. Hence in this sense the lasso is a less flexible approach than linear regression. It is also more interpretable than linear regression, because in the final model the response variable will only be related to a small subset of the predictors—namely, those with nonzero coefficient estimates. *Generalized additive models* (GAMs), discussed in Chapter 7, instead extend the lin- generalized

ear model (2.4) to allow for certain non-linear relationships. Consequently,

additive model

26 2. Statistical Learning

GAMs are more flexible than linear regression. They are also somewhat less interpretable than linear regression, because the relationship between each predictor and the response is now modeled using a curve. Finally, fully non-linear methods such as *bagging*, *boosting*, *support vector machines* bagging

boosting with non-linear kernels, and *neural networks* (deep learning), discussed in

Chapters 8, 9, and 10, are highly flexible approaches that are harder to interpret.

We have established that when inference is the goal, there are clear ad vantages to using simple and relatively inflexible statistical learning meth ods. In some settings, however, we are only interested in prediction, and the interpretability of the predictive model is simply not of interest. For instance, if we seek to develop an algorithm to predict the price of a stock, our sole requirement for the algorithm is that it predict accurately— interpretability is not a concern. In this setting, we might expect that it will be best to use the most flexible model available. Surprisingly, this is not always the case! We will often obtain more accurate predictions using a less flexible method. This phenomenon, which may seem counterintuitive at first glance, has to do with the potential for overfitting in highly flexible methods. We saw an example of overfitting in Figure 2.6. We will discuss this very important concept further in Section 2.2 and throughout this book.

*2.1.4 Supervised Versus Unsupervised Learning*

support vector

machine

Most statistical learning problems fall into one of two categories: *supervised* supervised or *unsupervised*. The examples that we have discussed so far in this chap- unsupervised ter all fall into the supervised learning domain. For each observation of the predictor measurement(s) *xi*, *i* = 1*,...,n* there is an associated response measurement *yi*. We wish to fit a model that relates the response to the predictors, with the aim of accurately predicting the response for future observations (prediction) or better understanding the relationship between the response and the predictors (inference). Many classical statistical learn ing methods such as linear regression and *logistic regression* (Chapter 4), as logistic well as more modern approaches such as GAM, boosting, and support vec- regression tor machines, operate in the supervised learning domain. The vast majority of this book is devoted to this setting.

By contrast, unsupervised learning describes the somewhat more chal lenging situation in which for every observation *i* = 1*,...,n*, we observe a vector of measurements *xi* but no associated response *yi*. It is not pos sible to fit a linear regression model, since there is no response variable to predict. In this setting, we are in some sense working blind; the sit uation is referred to as *unsupervised* because we lack a response vari able that can supervise our analysis. What sort of statistical analysis is possible? We can seek to understand the relationships between the variables or between the observations. One statistical learning tool that we may use

2

*X*

2 1

0 1

8 6

|  |  |
| --- | --- |
|  |  |

2.1 What Is Statistical Learning? 27

8

6

2

*X*

4

4

2

2

0 2 4 6 8 10 12

|  |
| --- |
|  |

0246

*X*1 *X*1

FIGURE 2.8. *A clustering data set involving three groups. Each group is shown using a different colored symbol.* Left: *The three groups are well-separated. In this setting, a clustering approach should successfully identify the three groups.* Right: *There is some overlap among the groups. Now the clustering task is more challenging.*

in this setting is *cluster analysis*, or clustering. The goal of cluster analysis cluster analysis is to ascertain, on the basis of *x*1*,...,xn*, whether the observations fall into relatively distinct groups. For example, in a market segmentation study we might observe multiple characteristics (variables) for potential customers, such as zip code, family income, and shopping habits. We might believe that the customers fall into different groups, such as big spenders versus low spenders. If the information about each customer’s spending patterns were available, then a supervised analysis would be possible. However, this information is not available—that is, we do not know whether each poten tial customer is a big spender or not. In this setting, we can try to cluster the customers on the basis of the variables measured, in order to identify distinct groups of potential customers. Identifying such groups can be of interest because it might be that the groups differ with respect to some property of interest, such as spending habits.

Figure 2.8 provides a simple illustration of the clustering problem. We have plotted 150 observations with measurements on two variables, *X*1 and *X*2. Each observation corresponds to one of three distinct groups. For illustrative purposes, we have plotted the members of each group using dif ferent colors and symbols. However, in practice the group memberships are unknown, and the goal is to determine the group to which each observa tion belongs. In the left-hand panel of Figure 2.8, this is a relatively easy task because the groups are well-separated. By contrast, the right-hand panel illustrates a more challenging setting in which there is some overlap

28 2. Statistical Learning

between the groups. A clustering method could not be expected to assign all of the overlapping points to their correct group (blue, green, or orange). In the examples shown in Figure 2.8, there are only two variables, and so one can simply visually inspect the scatterplots of the observations in order to identify clusters. However, in practice, we often encounter data sets that contain many more than two variables. In this case, we cannot easily plot the observations. For instance, if there are *p* variables in our data set, then *p*(*p −* 1)*/*2 distinct scatterplots can be made, and visual inspection is simply not a viable way to identify clusters. For this reason, automated clustering methods are important. We discuss clustering and other unsupervised learning approaches in Chapter 12.

Many problems fall naturally into the supervised or unsupervised learn ing paradigms. However, sometimes the question of whether an analysis should be considered supervised or unsupervised is less clear-cut. For in stance, suppose that we have a set of *n* observations. For *m* of the observa tions, where *m<n*, we have both predictor measurements and a response measurement. For the remaining *n − m* observations, we have predictor measurements but no response measurement. Such a scenario can arise if the predictors can be measured relatively cheaply but the corresponding responses are much more expensive to collect. We refer to this setting as a *semi-supervised learning* problem. In this setting, we wish to use a sta- semi

tistical learning method that can incorporate the *m* observations for which response measurements are available as well as the *n − m* observations for which they are not. Although this is an interesting topic, it is beyond the scope of this book.

*2.1.5 Regression Versus Classification Problems*

supervised learning

Variables can be characterized as either *quantitative* or *qualitative* (also quantitative qualitative known as *categorical*). Quantitative variables take on numerical values. categorical Examples include a person’s age, height, or income, the value of a house,

and the price of a stock. In contrast, qualitative variables take on values in one of *K* different *classes*, or categories. Examples of qualitative vari- class ables include a person’s marital status (married or not), the brand of prod uct purchased (brand A, B, or C), whether a person defaults on a debt (yes or no), or a cancer diagnosis (Acute Myelogenous Leukemia, Acute Lymphoblastic Leukemia, or No Leukemia). We tend to refer to problems with a quantitative response as *regression* problems, while those involv- regression ing a qualitative response are often referred to as *classification* problems. classification However, the distinction is not always that crisp. Least squares linear re gression (Chapter 3) is used with a quantitative response, whereas logistic regression (Chapter 4) is typically used with a qualitative (two-class, or *binary*) response. Thus, despite its name, logistic regression is a classifica- binary tion method. But since it estimates class probabilities, it can be thought of as a regression method as well. Some statistical methods, such as *K*-nearest

2.2 Assessing Model Accuracy 29

neighbors (Chapters 2 and 4) and boosting (Chapter 8), can be used in the case of either quantitative or qualitative responses.

We tend to select statistical learning methods on the basis of whether the response is quantitative or qualitative; i.e. we might use linear regres sion when quantitative and logistic regression when qualitative. However, whether the *predictors* are qualitative or quantitative is generally consid ered less important. Most of the statistical learning methods discussed in this book can be applied regardless of the predictor variable type, provided that any qualitative predictors are properly *coded* before the analysis is performed. This is discussed in Chapter 3.

2.2 Assessing Model Accuracy

One of the key aims of this book is to introduce the reader to a wide range of statistical learning methods that extend far beyond the standard linear regression approach. Why is it necessary to introduce so many different statistical learning approaches, rather than just a single *best* method? *There is no free lunch in statistics:* no one method dominates all others over all possible data sets. On a particular data set, one specific method may work best, but some other method may work better on a similar but different data set. Hence it is an important task to decide for any given set of data which method produces the best results. Selecting the best approach can be one of the most challenging parts of performing statistical learning in practice.

In this section, we discuss some of the most important concepts that arise in selecting a statistical learning procedure for a specific data set. As the book progresses, we will explain how the concepts presented here can be applied in practice.

*2.2.1 Measuring the Quality of Fit*

In order to evaluate the performance of a statistical learning method on a given data set, we need some way to measure how well its predictions actually match the observed data. That is, we need to quantify the extent to which the predicted response value for a given observation is close to the true response value for that observation. In the regression setting, the most commonly-used measure is the *mean squared error* (MSE), given by mean

squared

error *MSE* = 1*n*0*n*

(*yi −* ˆ*f*(*xi*))2*,* (2.5)

*i*=1

where ˆ*f*(*xi*) is the prediction that ˆ*f* gives for the *i*th observation. The MSE will be small if the predicted responses are very close to the true responses,

30 2. Statistical Learning

and will be large if for some of the observations, the predicted and true responses differ substantially.

The MSE in (2.5) is computed using the training data that was used to fit the model, and so should more accurately be referred to as the *training MSE*. But in general, we do not really care how well the method works training MSE on the training data. Rather, *we are interested in the accuracy of the pre dictions that we obtain when we apply our method to previously unseen test data*. Why is this what we care about? Suppose that we are interested test data in developing an algorithm to predict a stock’s price based on previous stock returns. We can train the method using stock returns from the past 6 months. But we don’t really care how well our method predicts last week’s stock price. We instead care about how well it will predict tomorrow’s price or next month’s price. On a similar note, suppose that we have clinical measurements (e.g. weight, blood pressure, height, age, family history of disease) for a number of patients, as well as information about whether each patient has diabetes. We can use these patients to train a statistical learn ing method to predict risk of diabetes based on clinical measurements. In practice, we want this method to accurately predict diabetes risk for *future patients* based on their clinical measurements. We are not very interested in whether or not the method accurately predicts diabetes risk for patients used to train the model, since we already know which of those patients have diabetes.

To state it more mathematically, suppose that we fit our statistical learn ing method on our training observations *{*(*x*1*, y*1)*,*(*x*2*, y*2)*,...,*(*xn, yn*)*}*, and we obtain the estimate ˆ*f*. We can then compute ˆ*f*(*x*1)*,* ˆ*f*(*x*2)*,...,* ˆ*f*(*xn*). If these are approximately equal to *y*1*, y*2*,...,yn*, then the training MSE given by (2.5) is small. However, we are really not interested in whether ˆ*f*(*xi*) *≈ yi*; instead, we want to know whether ˆ*f*(*x*0) is approximately equal to *y*0, where (*x*0*, y*0) is a *previously unseen test observation not used to train the statistical learning method*. We want to choose the method that gives the lowest *test MSE*, as opposed to the lowest training MSE. In other words, test MSE if we had a large number of test observations, we could compute

Ave(*y*0 *−* ˆ*f*(*x*0))2*,* (2.6)

the average squared prediction error for these test observations (*x*0*, y*0). We’d like to select the model for which this quantity is as small as possible. How can we go about trying to select a method that minimizes the test MSE? In some settings, we may have a test data set available—that is, we may have access to a set of observations that were not used to train the statistical learning method. We can then simply evaluate (2.6) on the test observations, and select the learning method for which the test MSE is smallest. But what if no test observations are available? In that case, one might imagine simply selecting a statistical learning method that minimizes the training MSE (2.5). This seems like it might be a sensible approach,

Y

2.2 Assessing Model Accuracy 31

5

.

2

2

1

0

.

2

0

r

1

o

r

r

E

5

.

d

1

8

e

r

a

u

q

S

0

.

6

1

n

a

e

M

4

5

.

0

2

0

.

0

0 20 40 60 80 100

X

2 5 10 20 Flexibility

FIGURE 2.9. Left: *Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing spline fits (blue and green curves).* Right: *Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.*

since the training MSE and the test MSE appear to be closely related. Unfortunately, there is a fundamental problem with this strategy: there is no guarantee that the method with the lowest training MSE will also have the lowest test MSE. Roughly speaking, the problem is that many statistical methods specifically estimate coefficients so as to minimize the training set MSE. For these methods, the training set MSE can be quite small, but the test MSE is often much larger.

Figure 2.9 illustrates this phenomenon on a simple example. In the left hand panel of Figure 2.9, we have generated observations from (2.1) with the true *f* given by the black curve. The orange, blue and green curves illus trate three possible estimates for *f* obtained using methods with increasing levels of flexibility. The orange line is the linear regression fit, which is rela tively inflexible. The blue and green curves were produced using *smoothing splines*, discussed in Chapter 7, with different levels of smoothness. It is smoothing

spline clear that as the level of flexibility increases, the curves fit the observed data more closely. The green curve is the most flexible and matches the data very well; however, we observe that it fits the true *f* (shown in black) poorly because it is too wiggly. By adjusting the level of flexibility of the smoothing spline fit, we can produce many different fits to this data.

We now move on to the right-hand panel of Figure 2.9. The grey curve displays the average training MSE as a function of flexibility, or more for mally the *degrees of freedom*, for a number of smoothing splines. The de- degrees of freedom grees of freedom is a quantity that summarizes the flexibility of a curve; it

32 2. Statistical Learning

is discussed more fully in Chapter 7. The orange, blue and green squares indicate the MSEs associated with the corresponding curves in the left hand panel. A more restricted and hence smoother curve has fewer degrees of freedom than a wiggly curve—note that in Figure 2.9, linear regression is at the most restrictive end, with two degrees of freedom. The training MSE declines monotonically as flexibility increases. In this example the true *f* is non-linear, and so the orange linear fit is not flexible enough to estimate *f* well. The green curve has the lowest training MSE of all three methods, since it corresponds to the most flexible of the three curves fit in the left-hand panel.

In this example, we know the true function *f*, and so we can also com pute the test MSE over a very large test set, as a function of flexibility. (Of course, in general *f* is unknown, so this will not be possible.) The test MSE is displayed using the red curve in the right-hand panel of Figure 2.9. As with the training MSE, the test MSE initially declines as the level of flex ibility increases. However, at some point the test MSE levels off and then starts to increase again. Consequently, the orange and green curves both have high test MSE. The blue curve minimizes the test MSE, which should not be surprising given that visually it appears to estimate *f* the best in the left-hand panel of Figure 2.9. The horizontal dashed line indicates Var(*ϵ*), the irreducible error in (2.3), which corresponds to the lowest achievable test MSE among all possible methods. Hence, the smoothing spline repre sented by the blue curve is close to optimal.

In the right-hand panel of Figure 2.9, as the flexibility of the statistical learning method increases, we observe a monotone decrease in the training MSE and a *U-shape* in the test MSE. This is a fundamental property of statistical learning that holds regardless of the particular data set at hand and regardless of the statistical method being used. As model flexibility increases, training MSE will decrease, but the test MSE may not. When a given method yields a small training MSE but a large test MSE, we are said to be *overfitting* the data. This happens because our statistical learning procedure is working too hard to find patterns in the training data, and may be picking up some patterns that are just caused by random chance rather than by true properties of the unknown function *f*. When we overfit the training data, the test MSE will be very large because the supposed patterns that the method found in the training data simply don’t exist in the test data. Note that regardless of whether or not overfitting has occurred, we almost always expect the training MSE to be smaller than the test MSE because most statistical learning methods either directly or indirectly seek to minimize the training MSE. Overfitting refers specifically to the case in which a less flexible model would have yielded a smaller test MSE.

Figure 2.10 provides another example in which the true *f* is approxi mately linear. Again we observe that the training MSE decreases mono tonically as the model flexibility increases, and that there is a U-shape in

Y

2.2 Assessing Model Accuracy 33

5

.

2

2

1

0

.

2

0

r

1

o

r

r

E

5

.

d

1

8

e

r

a

u

q

S

0

.

6

1

n

a

e

M

4

5

.

0

2

0

.

0

0 20 40 60 80 100

X

2 5 10 20 Flexibility

FIGURE 2.10. *Details are as in Figure 2.9, using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data.*

the test MSE. However, because the truth is close to linear, the test MSE only decreases slightly before increasing again, so that the orange least squares fit is substantially better than the highly flexible green curve. Fi nally, Figure 2.11 displays an example in which *f* is highly non-linear. The training and test MSE curves still exhibit the same general patterns, but now there is a rapid decrease in both curves before the test MSE starts to increase slowly.

In practice, one can usually compute the training MSE with relative ease, but estimating test MSE is considerably more difficult because usually no test data are available. As the previous three examples illustrate, the flexibility level corresponding to the model with the minimal test MSE can vary considerably among data sets. Throughout this book, we discuss a variety of approaches that can be used in practice to estimate this minimum point. One important method is *cross-validation* (Chapter 5), which is a cross method for estimating test MSE using the training data. validation

*2.2.2 The Bias-Variance Trade-Off*

The U-shape observed in the test MSE curves (Figures 2.9–2.11) turns out to be the result of two competing properties of statistical learning methods. Though the mathematical proof is beyond the scope of this book, it is possible to show that the expected test MSE, for a given value *x*0, can always be decomposed into the sum of three fundamental quantities: the *variance* of ˆ*f*(*x*0), the squared *bias* of ˆ*f*(*x*0) and the variance of the error variance

bias

34 2. Statistical Learning

0

2

0

2

5

r

1

o

r

r

E

d

0

e

1

r

a

0

Y

1

u

q

S

n

a

e

0

M

5

0

1

−

0

0 20 40 60 80 100

X

2 5 10 20 Flexibility

FIGURE 2.11. *Details are as in Figure 2.9, using a different f that is far from linear. In this setting, linear regression provides a very poor fit to the data.*

terms *ϵ*. That is,

1

*E*

*y*0 *−* ˆ*f*(*x*0)22= Var( ˆ*f*(*x*0)) + [Bias( ˆ*f*(*x*0))]2 + Var(*ϵ*)*.* (2.7)

1

Here the notation *E*

*y*0 *−* ˆ*f*(*x*0)22defines the *expected test MSE* at *x*0, expected

and refers to the average test MSE that we would obtain if we repeatedly test MSE estimated *f* using a large number of training sets, and tested each at *x*0. The

1

overall expected test MSE can be computed by averaging *E* over all possible values of *x*0 in the test set.

*y*0 *−* ˆ*f*(*x*0)22

Equation 2.7 tells us that in order to minimize the expected test error, we need to select a statistical learning method that simultaneously achieves *low variance* and *low bias*. Note that variance is inherently a nonnegative quantity, and squared bias is also nonnegative. Hence, we see that the expected test MSE can never lie below Var(*ϵ*), the irreducible error from (2.3).

What do we mean by the *variance* and *bias* of a statistical learning method? *Variance* refers to the amount by which ˆ*f* would change if we estimated it using a different training data set. Since the training data are used to fit the statistical learning method, different training data sets will result in a different ˆ*f*. But ideally the estimate for *f* should not vary too much between training sets. However, if a method has high variance then small changes in the training data can result in large changes in ˆ*f*. In general, more flexible statistical methods have higher variance. Consider the green and orange curves in Figure 2.9. The flexible green curve is following the observations very closely. It has high variance because changing any

2.2 Assessing Model Accuracy 35

one of these data points may cause the estimate ˆ*f* to change considerably. In contrast, the orange least squares line is relatively inflexible and has low variance, because moving any single observation will likely cause only a small shift in the position of the line.

On the other hand, *bias* refers to the error that is introduced by approxi mating a real-life problem, which may be extremely complicated, by a much simpler model. For example, linear regression assumes that there is a linear relationship between *Y* and *X*1*, X*2*,...,Xp*. It is unlikely that any real-life problem truly has such a simple linear relationship, and so performing lin ear regression will undoubtedly result in some bias in the estimate of *f*. In Figure 2.11, the true *f* is substantially non-linear, so no matter how many training observations we are given, it will not be possible to produce an accurate estimate using linear regression. In other words, linear regression results in high bias in this example. However, in Figure 2.10 the true *f* is very close to linear, and so given enough data, it should be possible for linear regression to produce an accurate estimate. Generally, more flexible methods result in less bias.

As a general rule, as we use more flexible methods, the variance will increase and the bias will decrease. The relative rate of change of these two quantities determines whether the test MSE increases or decreases. As we increase the flexibility of a class of methods, the bias tends to initially decrease faster than the variance increases. Consequently, the expected test MSE declines. However, at some point increasing flexibility has little impact on the bias but starts to significantly increase the variance. When this happens the test MSE increases. Note that we observed this pattern of decreasing test MSE followed by increasing test MSE in the right-hand panels of Figures 2.9–2.11.

The three plots in Figure 2.12 illustrate Equation 2.7 for the examples in Figures 2.9–2.11. In each case the blue solid curve represents the squared bias, for different levels of flexibility, while the orange curve corresponds to the variance. The horizontal dashed line represents Var(*ϵ*), the irreducible error. Finally, the red curve, corresponding to the test set MSE, is the sum of these three quantities. In all three cases, the variance increases and the bias decreases as the method’s flexibility increases. However, the flexibility level corresponding to the optimal test MSE differs considerably among the three data sets, because the squared bias and variance change at different rates in each of the data sets. In the left-hand panel of Figure 2.12, the bias initially decreases rapidly, resulting in an initial sharp decrease in the expected test MSE. On the other hand, in the center panel of Figure 2.12 the true *f* is close to linear, so there is only a small decrease in bias as flex ibility increases, and the test MSE only declines slightly before increasing rapidly as the variance increases. Finally, in the right-hand panel of Fig ure 2.12, as flexibility increases, there is a dramatic decline in bias because the true *f* is very non-linear. There is also very little increase in variance

36 2. Statistical Learning

5

.

2

0

.

2

5

.

1

0

.

1

5

.

0

0

.

0

2 5 10 20 Flexibility

5

.

2

0

.

2

5

.

1

0

.

1

5

.

0

0

.

0

2 5 10 20 Flexibility

0

2

5

1

0

1

5

0

2 5 10 20 Flexibility

MSE Bias Var

FIGURE 2.12. *Squared bias (blue curve), variance (orange curve), Var*(*ϵ*) *(dashed line), and test MSE (red curve) for the three data sets in Figures 2.9–2.11. The vertical dotted line indicates the flexibility level corresponding to the smallest test MSE.*

as flexibility increases. Consequently, the test MSE declines substantially before experiencing a small increase as model flexibility increases. The relationship between bias, variance, and test set MSE given in Equa tion 2.7 and displayed in Figure 2.12 is referred to as the *bias-variance trade-off*. Good test set performance of a statistical learning method re- bias-variance trade-off quires low variance as well as low squared bias. This is referred to as a trade-off because it is easy to obtain a method with extremely low bias but high variance (for instance, by drawing a curve that passes through every single training observation) or a method with very low variance but high bias (by fitting a horizontal line to the data). The challenge lies in finding a method for which both the variance and the squared bias are low. This trade-off is one of the most important recurring themes in this book. In a real-life situation in which *f* is unobserved, it is generally not pos sible to explicitly compute the test MSE, bias, or variance for a statistical learning method. Nevertheless, one should always keep the bias-variance trade-off in mind. In this book we explore methods that are extremely flexible and hence can essentially eliminate bias. However, this does not guarantee that they will outperform a much simpler method such as linear regression. To take an extreme example, suppose that the true *f* is linear. In this situation linear regression will have no bias, making it very hard for a more flexible method to compete. In contrast, if the true *f* is highly non-linear and we have an ample number of training observations, then we may do better using a highly flexible approach, as in Figure 2.11. In Chapter 5 we discuss cross-validation, which is a way to estimate the test MSE using the training data.

2.2 Assessing Model Accuracy 37

*2.2.3 The Classification Setting*

Thus far, our discussion of model accuracy has been focused on the regres sion setting. But many of the concepts that we have encountered, such as the bias-variance trade-off, transfer over to the classification setting with only some modifications due to the fact that *yi* is no longer quan titative. Suppose that we seek to estimate *f* on the basis of training obser vations *{*(*x*1*, y*1)*,...,*(*xn, yn*)*}*, where now *y*1*,...,yn* are qualitative. The most common approach for quantifying the accuracy of our estimate ˆ*f* is the training *error rate*, the proportion of mistakes that are made if we apply error rate

our estimate ˆ*f* to the training observations:

1

*n*

0*n i*=1

*I*(*yi ̸*= ˆ*yi*)*.* (2.8)

Here ˆ*yi* is the predicted class label for the *i*th observation using ˆ*f*. And *I*(*yi ̸*= ˆ*yi*) is an *indicator variable* that equals 1 if *yi ̸*= ˆ*yi* and zero if *yi* = ˆ*yi*. indicator variable If *I*(*yi ̸*= ˆ*yi*) = 0 then the *i*th observation was classified correctly by our classification method; otherwise it was misclassified. Hence Equation 2.8 computes the fraction of incorrect classifications.

Equation 2.8 is referred to as the *training error* rate because it is com- training error puted based on the data that was used to train our classifier. As in the regression setting, we are most interested in the error rates that result from applying our classifier to test observations that were not used in training. The *test error* rate associated with a set of test observations of the form test error (*x*0*, y*0) is given by Ave (*I*(*y*0 *̸*= ˆ*y*0))*,* (2.9)

where ˆ*y*0 is the predicted class label that results from applying the classifier to the test observation with predictor *x*0. A *good* classifier is one for which the test error (2.9) is smallest.

The Bayes Classifier

It is possible to show (though the proof is outside of the scope of this book) that the test error rate given in (2.9) is minimized, on average, by a very simple classifier that *assigns each observation to the most likely class, given its predictor values*. In other words, we should simply assign a test observation with predictor vector *x*0 to the class *j* for which

Pr(*Y* = *j|X* = *x*0) (2.10)

is largest. Note that (2.10) is a *conditional probability*: it is the probability conditional probability that *Y* = *j*, given the observed predictor vector *x*0. This very simple clas sifier is called the *Bayes classifier*. In a two-class problem where there are Bayes classifier only two possible response values, say *class 1* or *class 2*, the Bayes classifier

38 2. Statistical Learning

o

o

o

o

o

o

o

o o o o

o

o

o o

o

o

o

o

o

oo o o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o

o o

o

o

o

o

o

o o

o

o

o

o

o o o

o

o

o

oo

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o o o

oo o

o

o

o

o

o

o

o

o

o

o

o

o o

o

2

o

o

o o

o

o

o

o

*X*

o

o

o

o o

oo o o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o o o

o

o

o

o

o

o o

o o

o

o

o o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

*X*1

o

o o

o

o

o

o

o

o

o

FIGURE 2.13. *A simulated data set consisting of* 100 *observations in each of two groups, indicated in blue and in orange. The purple dashed line represents the Bayes decision boundary. The orange background grid indicates the region in which a test observation will be assigned to the orange class, and the blue background grid indicates the region in which a test observation will be assigned to the blue class.*

corresponds to predicting class one if Pr(*Y* = 1*|X* = *x*0) *>* 0*.*5, and class two otherwise.

Figure 2.13 provides an example using a simulated data set in a two dimensional space consisting of predictors *X*1 and *X*2. The orange and blue circles correspond to training observations that belong to two different classes. For each value of *X*1 and *X*2, there is a different probability of the response being orange or blue. Since this is simulated data, we know how the data were generated and we can calculate the conditional probabilities for each value of *X*1 and *X*2. The orange shaded region reflects the set of points for which Pr(*Y* = orange*|X*) is greater than 50 %, while the blue shaded region indicates the set of points for which the probability is below 50 %. The purple dashed line represents the points where the probability is exactly 50 %. This is called the *Bayes decision boundary*. The Bayes Bayes

classifier’s prediction is determined by the Bayes decision boundary; an observation that falls on the orange side of the boundary will be assigned to the orange class, and similarly an observation on the blue side of the boundary will be assigned to the blue class.

The Bayes classifier produces the lowest possible test error rate, called

decision boundary

the *Bayes error rate*. Since the Bayes classifier will always choose the class Bayes error rate for which (2.10) is largest, the error rate will be 1*−*max*j* Pr(*Y* = *j|X* = *x*0)

2.2 Assessing Model Accuracy 39

at *X* = *x*0. In general, the overall Bayes error rate is given by

1 *− E*

\*

max

*j* Pr(*Y* = *j|X*)

+

*,* (2.11)

where the expectation averages the probability over all possible values of *X*. For our simulated data, the Bayes error rate is 0*.*133. It is greater than zero, because the classes overlap in the true population so max*j* Pr(*Y* = *j|X* = *x*0) *<* 1 for some values of *x*0. The Bayes error rate is analogous to the irreducible error, discussed earlier.

*K*-Nearest Neighbors

In theory we would always like to predict qualitative responses using the Bayes classifier. But for real data, we do not know the conditional distri bution of *Y* given *X*, and so computing the Bayes classifier is impossi ble. Therefore, the Bayes classifier serves as an unattainable gold standard against which to compare other methods. Many approaches attempt to estimate the conditional distribution of *Y* given *X*, and then classify a given observation to the class with highest *estimated* probability. One such method is the *K-nearest neighbors* (KNN) classifier. Given a positive in- *K*-nearest

neighbors teger *K* and a test observation *x*0, the KNN classifier first identifies the *K* points in the training data that are closest to *x*0, represented by *N*0. It then estimates the conditional probability for class *j* as the fraction of points in *N*0 whose response values equal *j*:

Pr(*Y* = *j|X* = *x*0) = 1*K*0 *i∈N*0

*I*(*yi* = *j*)*.* (2.12)

Finally, KNN classifies the test observation *x*0 to the class with the largest probability from (2.12).

Figure 2.14 provides an illustrative example of the KNN approach. In the left-hand panel, we have plotted a small training data set consisting of six blue and six orange observations. Our goal is to make a prediction for the point labeled by the black cross. Suppose that we choose *K* = 3. Then KNN will first identify the three observations that are closest to the cross. This neighborhood is shown as a circle. It consists of two blue points and one orange point, resulting in estimated probabilities of 2*/*3 for the blue class and 1*/*3 for the orange class. Hence KNN will predict that the black cross belongs to the blue class. In the right-hand panel of Figure 2.14 we have applied the KNN approach with *K* = 3 at all of the possible values for *X*1 and *X*2, and have drawn in the corresponding KNN decision boundary.

Despite the fact that it is a very simple approach, KNN can often pro duce classifiers that are surprisingly close to the optimal Bayes classifier. Figure 2.15 displays the KNN decision boundary, using *K* = 10, when ap plied to the larger simulated data set from Figure 2.13. Notice that even

40 2. Statistical Learning

o

o

o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o o

o

o

o

o

FIGURE 2.14. *The KNN approach, using K* = 3*, is illustrated in a simple situation with six blue observations and six orange observations.* Left: *a test ob servation at which a predicted class label is desired is shown as a black cross. The three closest points to the test observation are identified, and it is predicted that the test observation belongs to the most commonly-occurring class, in this case blue.* Right: *The KNN decision boundary for this example is shown in black. The blue grid indicates the region in which a test observation will be assigned to the blue class, and the orange grid indicates the region in which it will be assigned to the orange class.*

KNN: K=10

o

o

o o

o

o

o

o o o o

o

o

o o

o

o

o

o

o

oo o o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o

o o

o

o

o

o

o

o o

o

o

o

o

o o o

o

o

o

oo

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o o o

oo o

o

o

o

o

o

o

o

o

o

o

o

o o

o

2

o

o

o o

o

o

o

o

*X*

o

o

o

o o

oo o o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o o o

o

o

o

o

o

o o

o

o

o

o o

o

o

o

o

o

o

o

o o

o

o

o

~~o~~

o

o

o

o

o

o

o

o

o

o

*X*1

o

o o

o

o

o

o

o

o

o

FIGURE 2.15. *The black curve indicates the KNN decision boundary on the data from Figure 2.13, using K* = 10*. The Bayes decision boundary is shown as a purple dashed line. The KNN and Bayes decision boundaries are very similar.*

2.2 Assessing Model Accuracy 41

KNN: K=1 KNN: K=100

o

o

o o

o

o

o

o

o

o o

o

o

o

o o o o

o o o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

oo o o

o

o

o

o

o

o

o

o o

o

o

o

o

o

oo o o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o o

o o

o o

o

o

o o

o

o

oo

o

o

o o

o o

o

o

o o

o

o

oo

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o

o o

o

o

o o

o

o

o o

o

o

o o

o

o

o o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o o

oo o o

o

o

o

o

o o

oo o o

o

o

o

o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o o

o o o

o

o

o

o

o

o

o

o

o

o

o o

o o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

o

o

o

o

o

o o

o

o

o

o

o

o

o

FIGURE 2.16. *A comparison of the KNN decision boundaries (solid black curves) obtained using K* = 1 *and K* = 100 *on the data from Figure 2.13. With K* = 1*, the decision boundary is overly flexible, while with K* = 100 *it is not sufficiently flexible. The Bayes decision boundary is shown as a purple dashed line.*

though the true distribution is not known by the KNN classifier, the KNN decision boundary is very close to that of the Bayes classifier. The test error rate using KNN is 0*.*1363, which is close to the Bayes error rate of 0*.*1304.

The choice of *K* has a drastic effect on the KNN classifier obtained. Figure 2.16 displays two KNN fits to the simulated data from Figure 2.13, using *K* = 1 and *K* = 100. When *K* = 1, the decision boundary is overly flexible and finds patterns in the data that don’t correspond to the Bayes decision boundary. This corresponds to a classifier that has low bias but very high variance. As *K* grows, the method becomes less flexible and produces a decision boundary that is close to linear. This corresponds to a low-variance but high-bias classifier. On this simulated data set, neither *K* = 1 nor *K* = 100 give good predictions: they have test error rates of 0*.*1695 and 0*.*1925, respectively.

Just as in the regression setting, there is not a strong relationship be tween the training error rate and the test error rate. With *K* = 1, the KNN training error rate is 0, but the test error rate may be quite high. In general, as we use more flexible classification methods, the training error rate will decline but the test error rate may not. In Figure 2.17, we have plotted the KNN test and training errors as a function of 1*/K*. As 1*/K* in creases, the method becomes more flexible. As in the regression setting, the training error rate consistently declines as the flexibility increases. However, the test error exhibits a characteristic U-shape, declining at first (with a minimum at approximately *K* = 10) before increasing again when the method becomes excessively flexible and overfits.

42 2. Statistical Learning

0

2

.

0

5

1

.

0

e

t

a

0

1

R

.

r

0

o

r

r

E

5

0

.

0

Training Errors

Test Errors

0

0

.

0

0.01 0.02 0.05 0.10 0.20 0.50 1.00 1/K

FIGURE 2.17. *The KNN training error rate (blue, 200 observations) and test error rate (orange, 5,000 observations) on the data from Figure 2.13, as the level of flexibility (assessed using* 1*/K on the log scale) increases, or equivalently as the number of neighbors K decreases. The black dashed line indicates the Bayes error rate. The jumpiness of the curves is due to the small size of the training data set.*

In both the regression and classification settings, choosing the correct level of flexibility is critical to the success of any statistical learning method. The bias-variance tradeoff, and the resulting U-shape in the test error, can make this a difficult task. In Chapter 5, we return to this topic and discuss various methods for estimating test error rates and thereby choosing the optimal level of flexibility for a given statistical learning method.

2.3 Lab: Introduction to R

In this lab, we will introduce some simple R commands. The best way to learn a new language is to try out the commands. R can be downloaded from

http://cran.r-project.org/

We recommend that you run R within an integrated development environ ment (IDE) such as RStudio, which can be freely downloaded from

http://rstudio.com

2.3 Lab: Introduction to R 43

The RStudio website also provides a cloud-based version of R, which does not require installing any software.

*2.3.1 Basic Commands*

R uses *functions* to perform operations. To run a function called funcname, function we type funcname(input1, input2), where the inputs (or *arguments*) input1 argument and input2 tell R how to run the function. A function can have any number of inputs. For example, to create a vector of numbers, we use the function c() (for *concatenate*). Any numbers inside the parentheses are joined to- c() gether. The following command instructs R to join together the numbers 1, 3, 2, and 5, and to save them as a *vector* named x. When we type x, it vector gives us back the vector.

> x <- c(1, 3, 2, 5)

> x

[1] 1 3 2 5

Note that the > is not part of the command; rather, it is printed by R to indicate that it is ready for another command to be entered. We can also save things using = rather than <-:

>x= c(1, 6, 2)

> x

[1] 1 6 2

>y= c(1, 4, 3)

Hitting the *up* arrow multiple times will display the previous commands, which can then be edited. This is useful since one often wishes to repeat a similar command. In addition, typing ?funcname will always cause R to open a new help file window with additional information about the function funcname().

We can tell R to add two sets of numbers together. It will then add the first number from x to the first number from y, and so on. However, x and y should be the same length. We can check their length using the length() length() function.

> length (x)

[1] 3

> length (y)

[1] 3

>x+y

[1] 2 10 5

The ls() function allows us to look at a list of all of the objects, such ls() as data and functions, that we have saved so far. The rm() function can be rm() used to delete any that we don’t want.

> ls ()

[1] "x" "y"

> rm(x, y)

44 2. Statistical Learning

> ls ()

character(0)

It’s also possible to remove all objects at once:

> rm(list = ls ())

The matrix() function can be used to create a matrix of numbers. Before matrix() we use the matrix() function, we can learn more about it: > ?matrix

The help file reveals that the matrix() function takes a number of inputs, but for now we focus on the first three: the data (the entries in the matrix), the number of rows, and the number of columns. First, we create a simple matrix.

> x <- matrix (data = c(1, 2, 3, 4), nrow = 2, ncol = 2) > x

[,1] [,2]

[1,] 1 3

[2,] 2 4

Note that we could just as well omit typing data=, nrow=, and ncol= in the matrix() command above: that is, we could just type

> x <- matrix (c(1, 2, 3, 4), 2, 2)

and this would have the same effect. However, it can sometimes be useful to specify the names of the arguments passed in, since otherwise R will assume that the function arguments are passed into the function in the same order that is given in the function’s help file. As this example illustrates, by default R creates matrices by successively filling in columns. Alternatively, the byrow = TRUE option can be used to populate the matrix in order of the rows.

> matrix (c(1, 2, 3, 4), 2, 2, byrow = TRUE)

[,1] [,2]

[1,] 1 2

[2,] 3 4

Notice that in the above command we did not assign the matrix to a value such as x. In this case the matrix is printed to the screen but is not saved for future calculations. The sqrt() function returns the square root of each sqrt() element of a vector or matrix. The command x^2 raises each element of x to the power 2; any powers are possible, including fractional or negative powers.

> sqrt (x)

[,1] [,2]

[1,] 1.00 1.73

[2,] 1.41 2.00

> x^2

[,1] [,2]

[1,] 1 9

[2,] 4 16

2.3 Lab: Introduction to R 45

The rnorm() function generates a vector of random normal variables, rnorm() with first argument n the sample size. Each time we call this function, we will get a different answer. Here we create two correlated sets of numbers, x and y, and use the cor() function to compute the correlation between cor() them.

> x <- rnorm (50)

> y <- x + rnorm (50, mean = 50, sd = .1)

> cor (x, y)

[1] 0.995

By default, rnorm() creates standard normal random variables with a mean of 0 and a standard deviation of 1. However, the mean and standard devi ation can be altered using the mean and sd arguments, as illustrated above. Sometimes we want our code to reproduce the exact same set of random numbers; we can use the set.seed() function to do this. The set.seed() set.seed() function takes an (arbitrary) integer argument.

> set.seed (1303)

> rnorm (50)

[1] -1.1440 1.3421 2.1854 0.5364 0.0632 0.5022 -0.0004 ...

We use set.seed() throughout the labs whenever we perform calculations involving random quantities. In general this should allow the user to re produce our results. However, as new versions of R become available, small discrepancies may arise between this book and the output from R.

The mean() and var() functions can be used to compute the mean and mean() var() variance of a vector of numbers. Applying sqrt() to the output of var() will give the standard deviation. Or we can simply use the sd() function. sd() > set.seed (3)

> y <- rnorm (100)

> mean (y)

[1] 0.0110

> var (y)

[1] 0.7329

> sqrt ( var (y))

[1] 0.8561

> sd(y)

[1] 0.8561

*2.3.2 Graphics*

The plot() function is the primary way to plot data in R. For instance, plot() plot(x, y) produces a scatterplot of the numbers in x versus the numbers in y. There are many additional options that can be passed in to the plot() function. For example, passing in the argument xlab will result in a label on the *x*-axis. To find out more information about the plot() function, type ?plot.

46 2. Statistical Learning

> x <- rnorm (100)

> y <- rnorm (100)

> plot (x, y)

> plot (x, y, xlab = " this is the x- axis ",

ylab = " this is the y- axis ",

main = " Plot of X vs Y")

We will often want to save the output of an R plot. The command that we use to do this will depend on the file type that we would like to create. For instance, to create a pdf, we use the pdf() function, and to create a jpeg, pdf() we use the jpeg() function.jpeg() > pdf (" Figure . pdf ")

> plot (x, y, col = " green ")

> dev .off ()

null device

1

The function dev.off() indicates to R that we are done creating the plot. dev.off() Alternatively, we can simply copy the plot window and paste it into an appropriate file type, such as a Word document.

The function seq() can be used to create a sequence of numbers. For seq() instance, seq(a, b) makes a vector of integers between a and b. There are many other options: for instance, seq(0, 1, length = 10) makes a sequence of 10 numbers that are equally spaced between 0 and 1. Typing 3:11 is a shorthand for seq(3, 11) for integer arguments.

> x <- seq (1, 10)

> x

[1] 1 2 3 4 5 6 7 8 9 10

> x <- 1:10

> x

[1] 1 2 3 4 5 6 7 8 9 10

> x <- seq (-pi, pi, length = 50)

We will now create some more sophisticated plots. The contour() func- contour() tion produces a *contour plot* in order to represent three-dimensional data; contour plot it is like a topographical map. It takes three arguments: 1. A vector of the x values (the first dimension),

2. A vector of the y values (the second dimension), and

3. A matrix whose elements correspond to the z value (the third dimen sion) for each pair of (x, y) coordinates.

As with the plot() function, there are many other inputs that can be used to fine-tune the output of the contour() function. To learn more about these, take a look at the help file by typing ?contour.

> y <- x

> f <- outer (x, y, function (x, y) cos (y) / (1 + x^2))

> contour (x, y, f)

> contour (x, y, f, nlevels = 45, add = T)

2.3 Lab: Introduction to R 47

> fa <- (f - t(f)) / 2

> contour (x, y, fa, nlevels = 15)

The image() function works the same way as contour(), except that it image() produces a color-coded plot whose colors depend on the z value. This is known as a *heatmap*, and is sometimes used to plot temperature in weather heatmap forecasts. Alternatively, persp() can be used to produce a three-dimensional persp() plot. The arguments theta and phi control the angles at which the plot is viewed.

> image (x, y, fa)

> persp (x, y, fa)

> persp (x, y, fa , theta = 30)

> persp (x, y, fa , theta = 30, phi = 20)

> persp (x, y, fa , theta = 30, phi = 70)

> persp (x, y, fa , theta = 30, phi = 40)

*2.3.3 Indexing Data*

We often wish to examine part of a set of data. Suppose that our data is stored in the matrix A.

> A <- matrix (1:16, 4, 4)

> A

[,1] [,2] [,3] [,4]

[1,] 1 5 9 13

[2,] 2 6 10 14

[3,] 3 7 11 15

[4,] 4 8 12 16

Then, typing

> A[2, 3]

[1] 10

will select the element corresponding to the second row and the third col umn. The first number after the open-bracket symbol [ always refers to the row, and the second number always refers to the column. We can also select multiple rows and columns at a time, by providing vectors as the indices.

> A[c(1, 3), c(2, 4)]

[,1] [,2]

[1,] 5 13

[2,] 7 15

> A[1:3, 2:4]

[,1] [,2] [,3]

[1,] 5 9 13

[2,] 6 10 14

[3,] 7 11 15

> A[1:2, ]

[,1] [,2] [,3] [,4]

[1,] 1 5 9 13

[2,] 2 6 10 14

48 2. Statistical Learning

> A[, 1:2]

[,1] [,2]

[1,] 1 5

[2,] 2 6

[3,] 3 7

[4,] 4 8

The last two examples include either no index for the columns or no index for the rows. These indicate that R should include all columns or all rows, respectively. R treats a single row or column of a matrix as a vector.

> A[1, ]

[1] 1 5 9 13

The use of a negative sign - in the index tells R to keep all rows or columns except those indicated in the index.

> A[-c(1, 3), ]

[,1] [,2] [,3] [,4]

[1,] 2 6 10 14

[2,] 4 8 12 16

> A[-c(1, 3), -c(1, 3, 4)]

[1] 6 8

The dim() function outputs the number of rows followed by the number of dim() columns of a given matrix.

> dim (A)

[1] 4 4

*2.3.4 Loading Data*

For most analyses, the first step involves importing a data set into R. The read.table() function is one of the primary ways to do this. The help file read.table() contains details about how to use this function. We can use the function write.table() to export data. write.table() Before attempting to load a data set, we must make sure that R knows to search for the data in the proper directory. For example, on a Windows system one could select the directory using the Change dir*...* option under the File menu. However, the details of how to do this depend on the oper ating system (e.g. Windows, Mac, Unix) that is being used, and so we do not give further details here.

We begin by loading in the Auto data set. This data is part of the ISLR2 library, discussed in Chapter 3. To illustrate the read.table() function, we load it now from a text file, Auto.data, which you can find on the textbook website. The following command will load the Auto.data file into R and store it as an object called Auto, in a format referred to as a *data frame*. data frame Once the data has been loaded, the View() function can be used to view

2.3 Lab: Introduction to R 49

it in a spreadsheet-like window.1 The head() function can also be used to view the first few rows of the data.

> Auto <- read . table (" Auto . data ")

> View (Auto)

> head (Auto)

V1 V2 V3 V4 V5

1 mpg cylinders displacement horsepower weight

2 18.0 8 307.0 130.0 3504.

3 15.0 8 350.0 165.0 3693.

4 18.0 8 318.0 150.0 3436.

5 16.0 8 304.0 150.0 3433.

6 17.0 8 302.0 140.0 3449.

V6 V7 V8 V9

1 acceleration year origin name 2 12.0 70 1 chevrolet chevelle malibu 3 11.5 70 1 buick skylark 320 4 11.0 70 1 plymouth satellite 5 12.0 70 1 amc rebel sst 6 10.5 70 1 ford torino

Note that Auto.data is simply a text file, which you could alternatively open on your computer using a standard text editor. It is often a good idea to view a data set using a text editor or other software such as Excel before loading it into R.

This particular data set has not been loaded correctly, because R has assumed that the variable names are part of the data and so has included them in the first row. The data set also includes a number of missing observations, indicated by a question mark ?. Missing values are a common occurrence in real data sets. Using the option header = T (or header = TRUE) in the read.table() function tells R that the first line of the file contains the variable names, and using the option na.strings tells R that any time it sees a particular character or set of characters (such as a question mark), it should be treated as a missing element of the data matrix.

> Auto <- read . table (" Auto . data ", header = T, na.strings = "?", stringsAsFactors = T)

> View (Auto)

The stringsAsFactors = T argument tells R that any variable containing character strings should be interpreted as a qualitative variable, and that each distinct character string represents a distinct level for that qualitative variable. An easy way to load data from Excel into R is to save it as a csv (comma-separated values) file, and then use the read.csv() function.

> Auto <- read . csv (" Auto . csv ", na.strings = "?",

stringsAsFactors = T)

> View (Auto)

1This function can sometimes be a bit finicky. If you have trouble using it, then try the head() function instead.

50 2. Statistical Learning

> dim (Auto)

[1] 397 9

> Auto[1:4, ]

The dim() function tells us that the data has 397 observations, or rows, and dim() nine variables, or columns. There are various ways to deal with the missing data. In this case, only five of the rows contain missing observations, and so we choose to use the na.omit() function to simply remove these rows. na.omit() > Auto <- na. omit (Auto)

> dim (Auto)

[1] 392 9

Once the data are loaded correctly, we can use names() to check the names() variable names.

> names (Auto)

[1] "mpg" "cylinders" "displacement" "horsepower" [5] "weight" "acceleration" "year" "origin" [9] "name"

*2.3.5 Additional Graphical and Numerical Summaries* We can use the plot() function to produce *scatterplots* of the quantitative scatterplot variables. However, simply typing the variable names will produce an error message, because R does not know to look in the Auto data set for those variables.

> plot (cylinders , mpg)

Error in plot(cylinders , mpg) : object ‘cylinders ’ not found

To refer to a variable, we must type the data set and the variable name joined with a $ symbol. Alternatively, we can use the attach() function in attach() order to tell R to make the variables in this data frame available by name.

> plot (Auto$cylinders , Auto$mpg)

> attach (Auto)

> plot (cylinders , mpg)

The cylinders variable is stored as a numeric vector, so R has treated it as quantitative. However, since there are only a small number of possible values for cylinders, one may prefer to treat it as a qualitative variable. The as.factor() function converts quantitative variables into qualitative as.factor() variables.

> cylinders <- as. factor (cylinders)

If the variable plotted on the *x*-axis is qualitative, then *boxplots* will boxplot automatically be produced by the plot() function. As usual, a number of options can be specified in order to customize the plots.

> plot (cylinders , mpg)

> plot (cylinders , mpg , col = " red ")

> plot (cylinders , mpg , col = " red ", varwidth = T)

2.3 Lab: Introduction to R 51

> plot (cylinders , mpg , col = " red ", varwidth = T,

horizontal = T)

> plot (cylinders , mpg , col = " red ", varwidth = T,

xlab = " cylinders ", ylab = " MPG ")

The hist() function can be used to plot a *histogram*. Note that col = 2 hist() histogram has the same effect as col = "red". > hist (mpg)

> hist (mpg , col = 2)

> hist (mpg , col = 2, breaks = 15)

The pairs() function creates a *scatterplot matrix*, i.e. a scatterplot for every pair of variables. We can also produce scatterplots for just a subset of the variables.

> pairs (Auto)

> pairs (

*∼* mpg + displacement + horsepower + weight + acceleration , data = Auto

)

In conjunction with the plot() function, identify() provides a useful identify() interactive method for identifying the value of a particular variable for points on a plot. We pass in three arguments to identify(): the *x*-axis variable, the *y*-axis variable, and the variable whose values we would like to see printed for each point. Then clicking one or more points in the plot and hitting Escape will cause R to print the values of the variable of interest. The numbers printed under the identify() function correspond to the rows for the selected points.

> plot (horsepower , mpg)

> identify (horsepower , mpg , name)

The summary() function produces a numerical summary of each variable in summary() a particular data set.

> summary (Auto)

mpg cylinders displacement

Min. : 9.00 Min. :3.000 Min. : 68.0

1st Qu.:17.00 1st Qu.:4.000 1st Qu.:105.0

Median :22.75 Median :4.000 Median :151.0

Mean :23.45 Mean :5.472 Mean :194.4

3rd Qu.:29.00 3rd Qu.:8.000 3rd Qu.:275.8

Max. :46.60 Max. :8.000 Max. :455.0

horsepower weight acceleration

Min. : 46.0 Min. :1613 Min. : 8.00

1st Qu.: 75.0 1st Qu.:2225 1st Qu.:13.78

Median : 93.5 Median :2804 Median :15.50

Mean :104.5 Mean :2978 Mean :15.54

3rd Qu.:126.0 3rd Qu.:3615 3rd Qu.:17.02

Max. :230.0 Max. :5140 Max. :24.80

year origin name

52 2. Statistical Learning

Min. :70.00 Min. :1.000 amc matador : 5

1st Qu.:73.00 1st Qu.:1.000 ford pinto : 5

Median :76.00 Median :1.000 toyota corolla : 5

Mean :75.98 Mean :1.577 amc gremlin : 4

3rd Qu.:79.00 3rd Qu.:2.000 amc hornet : 4

Max. :82.00 Max. :3.000 chevrolet chevette: 4

(Other) :365

For qualitative variables such as name, R will list the number of observations that fall in each category. We can also produce a summary of just a single variable.

> summary (mpg)

Min. 1st Qu. Median Mean 3rd Qu. Max.

9.00 17.00 22.75 23.45 29.00 46.60

Once we have finished using R, we type q() in order to shut it down, or q() quit. When exiting R, we have the option to save the current *workspace* soworkspace that all objects (such as data sets) that we have created in this R session will be available next time. Before exiting R, we may want to save a record of all of the commands that we typed in the most recent session; this can be accomplished using the savehistory() function. Next time we enter R, savehistory() we can load that history using the loadhistory() function, if we wish. loadhistory()

2.4 Exercises

*Conceptual*

1. For each of parts (a) through (d), indicate whether we would generally expect the performance of a flexible statistical learning method to be better or worse than an inflexible method. Justify your answer.

(a) The sample size *n* is extremely large, and the number of predic tors *p* is small.

(b) The number of predictors *p* is extremely large, and the number of observations *n* is small.

(c) The relationship between the predictors and response is highly non-linear.

(d) The variance of the error terms, i.e. *σ*2 = Var(*ϵ*), is extremely high.

2. Explain whether each scenario is a classification or regression prob lem, and indicate whether we are most interested in inference or pre diction. Finally, provide *n* and *p*.

(a) We collect a set of data on the top 500 firms in the US. For each firm we record profit, number of employees, industry and the

CEO salary. We are interested in understanding which factors

affect CEO salary.

2.4 Exercises 53

(b) We are considering launching a new product and wish to know whether it will be a *success* or a *failure*. We collect data on 20 similar products that were previously launched. For each prod uct we have recorded whether it was a success or failure, price charged for the product, marketing budget, competition price, and ten other variables.

(c) We are interested in predicting the % change in the USD/Euro exchange rate in relation to the weekly changes in the world stock markets. Hence we collect weekly data for all of 2012. For each week we record the % change in the USD/Euro, the % change in the US market, the % change in the British market, and the % change in the German market.

3. We now revisit the bias-variance decomposition.

(a) Provide a sketch of typical (squared) bias, variance, training er ror, test error, and Bayes (or irreducible) error curves, on a sin gle plot, as we go from less flexible statistical learning methods towards more flexible approaches. The *x*-axis should represent the amount of flexibility in the method, and the *y*-axis should represent the values for each curve. There should be five curves. Make sure to label each one.

(b) Explain why each of the five curves has the shape displayed in part (a).

4. You will now think of some real-life applications for statistical learn ing.

(a) Describe three real-life applications in which *classification* might be useful. Describe the response, as well as the predictors. Is the goal of each application inference or prediction? Explain your answer.

(b) Describe three real-life applications in which *regression* might be useful. Describe the response, as well as the predictors. Is the goal of each application inference or prediction? Explain your answer.

(c) Describe three real-life applications in which *cluster analysis* might be useful.

5. What are the advantages and disadvantages of a very flexible (versus a less flexible) approach for regression or classification? Under what circumstances might a more flexible approach be preferred to a less flexible approach? When might a less flexible approach be preferred?

54 2. Statistical Learning

6. Describe the differences between a parametric and a non-parametric statistical learning approach. What are the advantages of a para metric approach to regression or classification (as opposed to a non parametric approach)? What are its disadvantages?

7. The table below provides a training data set containing six observa tions, three predictors, and one qualitative response variable.

Obs. *X*1 *X*2 *X*3 *Y*

1 0 3 0 Red

2 2 0 0 Red

3 0 1 3 Red

4 0 1 2 Green

5 *−*1 0 1 Green

6 1 1 1 Red

Suppose we wish to use this data set to make a prediction for *Y* when *X*1 = *X*2 = *X*3 = 0 using *K*-nearest neighbors.

(a) Compute the Euclidean distance between each observation and the test point, *X*1 = *X*2 = *X*3 = 0.

(b) What is our prediction with *K* = 1? Why?

(c) What is our prediction with *K* = 3? Why?

(d) If the Bayes decision boundary in this problem is highly non linear, then would we expect the *best* value for *K* to be large or small? Why?

*Applied*

8. This exercise relates to the College data set, which can be found in the file College.csv on the book website. It contains a number of variables for 777 different universities and colleges in the US. The variables are

*•* Private : Public/private indicator

*•* Apps : Number of applications received

*•* Accept : Number of applicants accepted

*•* Enroll : Number of new students enrolled

*•* Top10perc : New students from top 10 % of high school class *•* Top25perc : New students from top 25 % of high school class *•* F.Undergrad : Number of full-time undergraduates

*•* P.Undergrad : Number of part-time undergraduates

2.4 Exercises 55

*•* Outstate : Out-of-state tuition

*•* Room.Board : Room and board costs

*•* Books : Estimated book costs

*•* Personal : Estimated personal spending

*•* PhD : Percent of faculty with Ph.D.’s

*•* Terminal : Percent of faculty with terminal degree *•* S.F.Ratio : Student/faculty ratio

*•* perc.alumni : Percent of alumni who donate

*•* Expend : Instructional expenditure per student

*•* Grad.Rate : Graduation rate

Before reading the data into R, it can be viewed in Excel or a text editor.

(a) Use the read.csv() function to read the data into R. Call the loaded data college. Make sure that you have the directory set to the correct location for the data.

(b) Look at the data using the View() function. You should notice that the first column is just the name of each university. We don’t really want R to treat this as data. However, it may be handy to have these names for later. Try the following commands:

> rownames (college) <- college[, 1]

> View (college)

You should see that there is now a row.names column with the name of each university recorded. This means that R has given each row a name corresponding to the appropriate university. R will not try to perform calculations on the row names. However, we still need to eliminate the first column in the data where the names are stored. Try

> college <- college[, -1]

> View (college)

Now you should see that the first data column is Private. Note that another column labeled row.names now appears before the Private column. However, this is not a data column but rather the name that R is giving to each row.

(c) i. Use the summary() function to produce a numerical summary of the variables in the data set.

ii. Use the pairs() function to produce a scatterplot matrix of the first ten columns or variables of the data. Recall that you can reference the first ten columns of a matrix A using A[,1:10].

56 2. Statistical Learning

iii. Use the plot() function to produce side-by-side boxplots of Outstate versus Private.

iv. Create a new qualitative variable, called Elite, by *binning* the Top10perc variable. We are going to divide universities

into two groups based on whether or not the proportion

of students coming from the top 10 % of their high school

classes exceeds 50 %.

> Elite <- rep ("No", nrow (college))

> Elite[college$Top10perc > 50] <- " Yes "

> Elite <- as. factor (Elite)

> college <- data . frame (college , Elite)

Use the summary() function to see how many elite univer

sities there are. Now use the plot() function to produce

side-by-side boxplots of Outstate versus Elite.

v. Use the hist() function to produce some histograms with differing numbers of bins for a few of the quantitative vari

ables. You may find the command par(mfrow = c(2, 2))

useful: it will divide the print window into four regions so

that four plots can be made simultaneously. Modifying the

arguments to this function will divide the screen in other

ways.

vi. Continue exploring the data, and provide a brief summary of what you discover.

9. This exercise involves the Auto data set studied in the lab. Make sure that the missing values have been removed from the data.

(a) Which of the predictors are quantitative, and which are quali tative?

(b) What is the *range* of each quantitative predictor? You can an swer this using the range() function. range() (c) What is the mean and standard deviation of each quantitative predictor?

(d) Now remove the 10th through 85th observations. What is the range, mean, and standard deviation of each predictor in the subset of the data that remains?

(e) Using the full data set, investigate the predictors graphically, using scatterplots or other tools of your choice. Create some plots highlighting the relationships among the predictors. Comment on your findings.

(f) Suppose that we wish to predict gas mileage (mpg) on the basis of the other variables. Do your plots suggest that any of the other variables might be useful in predicting mpg? Justify your answer.

2.4 Exercises 57

10. This exercise involves the Boston housing data set.

(a) To begin, load in the Boston data set. The Boston data set is part of the ISLR2 *library*.

> library (ISLR2)

Now the data set is contained in the object Boston.

> Boston

Read about the data set:

> ?Boston

How many rows are in this data set? How many columns? What do the rows and columns represent?

(b) Make some pairwise scatterplots of the predictors (columns) in this data set. Describe your findings.

(c) Are any of the predictors associated with per capita crime rate? If so, explain the relationship.

(d) Do any of the census tracts of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each predictor.

(e) How many of the census tracts in this data set bound the Charles river?

(f) What is the median pupil-teacher ratio among the towns in this data set?

(g) Which census tract of Boston has lowest median value of owner occupied homes? What are the values of the other predictors for that census tract, and how do those values compare to the overall ranges for those predictors? Comment on your findings.

(h) In this data set, how many of the census tracts average more than seven rooms per dwelling? More than eight rooms per dwelling? Comment on the census tracts that average more than eight rooms per dwelling.

3

Linear Regression



This chapter is about *linear regression*, a very simple approach for super vised learning. In particular, linear regression is a useful tool for predicting a quantitative response. It has been around for a long time and is the topic of innumerable textbooks. Though it may seem somewhat dull compared to some of the more modern statistical learning approaches described in later chapters of this book, linear regression is still a useful and widely used sta tistical learning method. Moreover, it serves as a good jumping-off point for newer approaches: as we will see in later chapters, many fancy statistical learning approaches can be seen as generalizations or extensions of linear regression. Consequently, the importance of having a good understanding of linear regression before studying more complex learning methods cannot be overstated. In this chapter, we review some of the key ideas underlying the linear regression model, as well as the least squares approach that is most commonly used to fit this model.

Recall the Advertising data from Chapter 2. Figure 2.1 displays sales (in thousands of units) for a particular product as a function of advertis ing budgets (in thousands of dollars) for TV, radio, and newspaper media. Suppose that in our role as statistical consultants we are asked to suggest, on the basis of this data, a marketing plan for next year that will result in high product sales. What information would be useful in order to provide such a recommendation? Here are a few important questions that we might seek to address:

1. *Is there a relationship between advertising budget and sales?* Our first goal should be to determine whether the data provide evi-

59

© Springer Science+Business Media, LLC, part of Springer Nature 2021 G. James et al., An Introduction to Statistical Learning, Springer Texts in Statistics, https://doi.org/10.1007/978-1-0716-1418-1\_3

60 3. Linear Regression

dence of an association between advertising expenditure and sales. If the evidence is weak, then one might argue that no money should be spent on advertising!

2. *How strong is the relationship between advertising budget and sales?* Assuming that there is a relationship between advertising and sales, we would like to know the strength of this relationship. Does knowl edge of the advertising budget provide a lot of information about product sales?

3. *Which media are associated with sales?*

Are all three media—TV, radio, and newspaper—associated with sales, or are just one or two of the media associated? To answer this question, we must find a way to separate out the individual contribu tion of each medium to sales when we have spent money on all three media.

4. *How large is the association between each medium and sales?*

For every dollar spent on advertising in a particular medium, by what amount will sales increase? How accurately can we predict this amount of increase?

5. *How accurately can we predict future sales?*

For any given level of television, radio, or newspaper advertising, what is our prediction for sales, and what is the accuracy of this prediction?

6. *Is the relationship linear?*

If there is approximately a straight-line relationship between advertis ing expenditure in the various media and sales, then linear regression is an appropriate tool. If not, then it may still be possible to trans form the predictor or the response so that linear regression can be used.

7. *Is there synergy among the advertising media?*

Perhaps spending $50*,*000 on television advertising and $50*,*000 on ra dio advertising is associated with higher sales than allocating $100*,*000 to either television or radio individually. In marketing, this is known as a *synergy* effect, while in statistics it is called an *interaction* effect. synergy

interaction It turns out that linear regression can be used to answer each of these questions. We will first discuss all of these questions in a general context, and then return to them in this specific context in Section 3.4.

3.1 Simple Linear Regression

*Simple linear regression* lives up to its name: it is a very straightforward simple linear regression

3.1 Simple Linear Regression 61

approach for predicting a quantitative response *Y* on the basis of a sin gle predictor variable *X*. It assumes that there is approximately a linear relationship between *X* and *Y* . Mathematically, we can write this linear relationship as

*Y ≈ β*0 + *β*1*X.* (3.1)

You might read “*≈*” as *“is approximately modeled as”*. We will sometimes describe (3.1) by saying that we are *regressing Y on X* (or *Y onto X*). For example, *X* may represent TV advertising and *Y* may represent sales. Then we can regress sales onto TV by fitting the model

sales *≈ β*0 + *β*1 *×* TV*.*

In Equation 3.1, *β*0 and *β*1 are two unknown constants that represent the *intercept* and *slope* terms in the linear model. Together, *β*0 and *β*1 are intercept slope known as the model *coefficients* or *parameters*. Once we have used our

training data to produce estimates *β*ˆ0 and *β*ˆ1 for the model coefficients, we can predict future sales on the basis of a particular value of TV advertising

coefficient parameter

by computing

*y*ˆ = *β*ˆ0 + *β*ˆ1*x,* (3.2)

where ˆ*y* indicates a prediction of *Y* on the basis of *X* = *x*. Here we use a *hat* symbol, ˆ , to denote the estimated value for an unknown parameter or coefficient, or to denote the predicted value of the response.

*3.1.1 Estimating the Coefficients*

In practice, *β*0 and *β*1 are unknown. So before we can use (3.1) to make predictions, we must use data to estimate the coefficients. Let

(*x*1*, y*1)*,* (*x*2*, y*2)*,...,* (*xn, yn*)

represent *n* observation pairs, each of which consists of a measurement of *X* and a measurement of *Y* . In the Advertising example, this data set consists of the TV advertising budget and product sales in *n* = 200 different markets. (Recall that the data are displayed in Figure 2.1.) Our goal is to obtain coefficient estimates *β*ˆ0 and *β*ˆ1 such that the linear model (3.1) fits the available data well—that is, so that *yi ≈ β*ˆ0 + *β*ˆ1*xi* for *i* = 1*,...,n*. In other words, we want to find an intercept *β*ˆ0 and a slope *β*ˆ1 such that the resulting line is as close as possible to the *n* = 200 data points. There are a number of ways of measuring *closeness*. However, by far the most common approach involves minimizing the *least squares* criterion, least squares and we take that approach in this chapter. Alternative approaches will be

considered in Chapter 6.

Let ˆ*yi* = *β*ˆ0 + *β*ˆ1*xi* be the prediction for *Y* based on the *i*th value of *X*. Then *ei* = *yi −y*ˆ*i* represents the *i*th *residual*—this is the difference between residual

62 3. Linear Regression

5

2

0

2

s

5

e

1

l

a

S

0

1

5

0 50 100 150 200 250 300

TV

FIGURE 3.1. *For the* Advertising *data, the least squares fit for the regression of* sales *onto* TV *is shown. The fit is found by minimizing the residual sum of squares. Each grey line segment represents a residual. In this case a linear fit captures the essence of the relationship, although it overestimates the trend in the left of the plot.*

the *i*th observed response value and the *i*th response value that is predicted by our linear model. We define the *residual sum of squares* (RSS) as residual sum of squares RSS = *e*21 + *e*22 + *···* + *e*2*n,*

or equivalently as

RSS = (*y*1*−β*ˆ0*−β*ˆ1*x*1)2+ (*y*2*−β*ˆ0*−β*ˆ1*x*2)2+*···*+ (*yn−β*ˆ0*−β*ˆ1*xn*)2*.* (3.3)

The least squares approach chooses *β*ˆ0 and *β*ˆ1 to minimize the RSS. Using some calculus, one can show that the minimizers are

)*n*

*β*ˆ1 =

*i*=1(*xi − x*¯)(*yi − y*¯) )*~~n~~*

*i*=1(*xi − x*¯)2 *,*

(3.4)

*β*ˆ0 = ¯*y − β*ˆ1*x,* ¯

where ¯*y ≡* 1*n*)*ni*=1 *yi* and ¯*x ≡* 1*n*)*ni*=1 *xi* are the sample means. In other words, (3.4) defines the *least squares coefficient estimates* for simple linear regression.

Figure 3.1 displays the simple linear regression fit to the Advertising data, where *β*ˆ0 = 7*.*03 and *β*ˆ1 = 0*.*0475. In other words, according to this approximation, an additional $1*,*000 spent on TV advertising is asso ciated with selling approximately 47*.*5 additional units of the product. In

1

β

3

2.5

6

0

.

0

5

0

.

0

4

0

.

0

3

0

.

0

2.15

2.2

2.3

2.5

3

2.11 ●

3.1 Simple Linear Regression 63

S

S

R

5 6 7 8 9 β0

β1 β0

FIGURE 3.2. *Contour and three-dimensional plots of the RSS on the* Advertising *data, using* sales *as the response and* TV *as the predictor. The red dots correspond to the least squares estimates β*ˆ0 *and β*ˆ1*, given by (3.4).*

Figure 3.2, we have computed RSS for a number of values of *β*0 and *β*1, using the advertising data with sales as the response and TV as the predic tor. In each plot, the red dot represents the pair of least squares estimates (*β*ˆ0*, β*ˆ1) given by (3.4). These values clearly minimize the RSS.

*3.1.2 Assessing the Accuracy of the Coefficient Estimates*

Recall from (2.1) that we assume that the *true* relationship between *X* and *Y* takes the form *Y* = *f*(*X*) + *ϵ* for some unknown function *f*, where *ϵ* is a mean-zero random error term. If *f* is to be approximated by a linear function, then we can write this relationship as

*Y* = *β*0 + *β*1*X* + *ϵ.* (3.5)

Here *β*0 is the intercept term—that is, the expected value of *Y* when *X* = 0, and *β*1 is the slope—the average increase in *Y* associated with a one-unit increase in *X*. The error term is a catch-all for what we miss with this simple model: the true relationship is probably not linear, there may be other variables that cause variation in *Y* , and there may be measurement error. We typically assume that the error term is independent of *X*.

The model given by (3.5) defines the *population regression line*, which population

is the best linear approximation to the true relationship between *X* and *Y* .1 The least squares regression coefficient estimates (3.4) characterize the

regression line

*least squares line* (3.2). The left-hand panel of Figure 3.3 displays these least squares line

1The assumption of linearity is often a useful working model. However, despite what many textbooks might tell us, we seldom believe that the true relationship is linear.

64 3. Linear Regression

0

0

1

1

5

5

Y

Y

0

0

5

5

−

−

0

0

1

1

−

−

−2 −1 0 1 2

X

−2 −1 0 1 2 X

FIGURE 3.3. *A simulated data set.* Left: *The red line represents the true rela tionship, f*(*X*)=2+3*X, which is known as the population regression line. The blue line is the least squares line; it is the least squares estimate for f*(*X*) *based on the observed data, shown in black.* Right: *The population regression line is again shown in red, and the least squares line in dark blue. In light blue, ten least squares lines are shown, each computed on the basis of a separate random set of observations. Each least squares line is different, but on average, the least squares lines are quite close to the population regression line.*

two lines in a simple simulated example. We created 100 random *X*s, and generated 100 corresponding *Y* s from the model

*Y* =2+3*X* + *ϵ,* (3.6)

where *ϵ* was generated from a normal distribution with mean zero. The red line in the left-hand panel of Figure 3.3 displays the *true* relationship, *f*(*X*) = 2+3*X*, while the blue line is the least squares estimate based on the observed data. The true relationship is generally not known for real data, but the least squares line can always be computed using the coefficient estimates given in (3.4). In other words, in real applications, we have access to a set of observations from which we can compute the least squares line; however, the population regression line is unobserved. In the right-hand panel of Figure 3.3 we have generated ten different data sets from the model given by (3.6) and plotted the corresponding ten least squares lines. Notice that different data sets generated from the same true model result in slightly different least squares lines, but the unobserved population regression line does not change.

At first glance, the difference between the population regression line and the least squares line may seem subtle and confusing. We only have one data set, and so what does it mean that two different lines describe the relationship between the predictor and the response? Fundamentally, the concept of these two lines is a natural extension of the standard statistical

3.1 Simple Linear Regression 65

approach of using information from a sample to estimate characteristics of a large population. For example, suppose that we are interested in knowing the population mean *µ* of some random variable *Y* . Unfortunately, *µ* is unknown, but we do have access to *n* observations from *Y* , *y*1*,...,yn*, which we can use to estimate *µ*. A reasonable estimate is ˆ*µ* = ¯*y*, where

*y*¯ = 1*n*)*ni*=1 *yi* is the sample mean. The sample mean and the population mean are different, but in general the sample mean will provide a good estimate of the population mean. In the same way, the unknown coefficients *β*0 and *β*1 in linear regression define the population regression line. We seek to estimate these unknown coefficients using *β*ˆ0 and *β*ˆ1 given in (3.4). These coefficient estimates define the least squares line.

The analogy between linear regression and estimation of the mean of a random variable is an apt one based on the concept of *bias*. If we use the bias sample mean ˆ*µ* to estimate *µ*, this estimate is *unbiased*, in the sense that unbiased on average, we expect ˆ*µ* to equal *µ*. What exactly does this mean? It means that on the basis of one particular set of observations *y*1*,...,yn*, ˆ*µ* might overestimate *µ*, and on the basis of another set of observations, ˆ*µ* might underestimate *µ*. But if we could average a huge number of estimates of *µ* obtained from a huge number of sets of observations, then this average would *exactly* equal *µ*. Hence, an unbiased estimator does not *systematically* over- or under-estimate the true parameter. The property of unbiasedness holds for the least squares coefficient estimates given by (3.4) as well: if we estimate *β*0 and *β*1 on the basis of a particular data set, then our estimates won’t be exactly equal to *β*0 and *β*1. But if we could average the estimates obtained over a huge number of data sets, then the average of these estimates would be spot on! In fact, we can see from the right hand panel of Figure 3.3 that the average of many least squares lines, each estimated from a separate data set, is pretty close to the true population regression line.

We continue the analogy with the estimation of the population mean *µ* of a random variable *Y* . A natural question is as follows: how accurate is the sample mean ˆ*µ* as an estimate of *µ*? We have established that the average of ˆ*µ*’s over many data sets will be very close to *µ*, but that a single estimate ˆ*µ* may be a substantial underestimate or overestimate of *µ*. How far off will that single estimate of ˆ*µ* be? In general, we answer this question by computing the *standard error* of ˆ*µ*, written as SE(ˆ*µ*). We have standard

error the well-known formula Var(ˆ*µ*) = SE(ˆ*µ*)2 = *σ*2

*n ,* (3.7)

where *σ* is the standard deviation of each of the realizations *yi* of *Y* .2 Roughly speaking, the standard error tells us the average amount that this estimate ˆ*µ* differs from the actual value of *µ*. Equation 3.7 also tells us how

2This formula holds provided that the *n* observations are uncorrelated.

66 3. Linear Regression

this deviation shrinks with *n*—the more observations we have, the smaller the standard error of ˆ*µ*. In a similar vein, we can wonder how close *β*ˆ0 and *β*ˆ1 are to the true values *β*0 and *β*1. To compute the standard errors associated with *β*ˆ0 and *β*ˆ1, we use the following formulas:

SE(*β*ˆ0)2= *σ*231*n*+*x*¯2

4

)*~~n~~*

*i*=1(*xi − x*¯)2

*,* SE(*β*ˆ1)2= *σ*2

)*~~n~~*

*i*=1(*xi − x*¯)2 *,* (3.8)

where *σ*2 = Var(*ϵ*). For these formulas to be strictly valid, we need to assume that the errors *ϵi* for each observation have common variance *σ*2 and are uncorrelated. This is clearly not true in Figure 3.1, but the formula still turns out to be a good approximation. Notice in the formula that SE(*β*ˆ1) is smaller when the *xi* are more spread out; intuitively we have more *leverage* to estimate a slope when this is the case. We also see that SE(*β*ˆ0) would be the same as SE(ˆ*µ*) if ¯*x* were zero (in which case *β*ˆ0 would be equal to ¯*y*). In general, *σ*2 is not known, but can be estimated from the data. This estimate of *σ* is known as the *residual standard error*, and is given by the formula residual

RSE = 5RSS*/*(*n −* 2). Strictly speaking, when *σ*2 is estimated from the data we should write SE( 6 *β*ˆ1) to indicate that an estimate has been made, but for simplicity of notation we will drop this extra “hat”.

standard error

Standard errors can be used to compute *confidence intervals*. A 95 % confidence interval confidence interval is defined as a range of values such that with 95 % probability, the range will contain the true unknown value of the param eter. The range is defined in terms of lower and upper limits computed from the sample of data. A 95% confidence interval has the following prop erty: if we take repeated samples and construct the confidence interval for each sample, 95% of the intervals will contain the true unknown value of the parameter. For linear regression, the 95 % confidence interval for *β*1 approximately takes the form

*β*ˆ1 *±* 2 *·* SE(*β*ˆ1)*.* (3.9)

That is, there is approximately a 95 % chance that the interval

7

*β*ˆ1 *−* 2 *·* SE(*β*ˆ1)*, β*ˆ1 + 2 *·* SE(*β*ˆ1)8(3.10)

will contain the true value of *β*1.3 Similarly, a confidence interval for *β*0 approximately takes the form

*β*ˆ0 *±* 2 *·* SE(*β*ˆ0)*.* (3.11)

3*Approximately* for several reasons. Equation 3.10 relies on the assumption that the errors are Gaussian. Also, the factor of 2 in front of the SE(*β*ˆ1) term will vary slightly depending on the number of observations *n* in the linear regression. To be precise, rather than the number 2, (3.10) should contain the 97.5 % quantile of a *t*-distribution with *n−*2 degrees of freedom. Details of how to compute the 95 % confidence interval precisely in R will be provided later in this chapter.

3.1 Simple Linear Regression 67

In the case of the advertising data, the 95 % confidence interval for *β*0 is [6*.*130*,* 7*.*935] and the 95 % confidence interval for *β*1 is [0*.*042*,* 0*.*053]. Therefore, we can conclude that in the absence of any advertising, sales will, on average, fall somewhere between 6*,*130 and 7*,*935 units. Furthermore, for each $1*,*000 increase in television advertising, there will be an average increase in sales of between 42 and 53 units.

Standard errors can also be used to perform *hypothesis tests* on the hypothesis test coefficients. The most common hypothesis test involves testing the *null hypothesis* of null hypothesis *H*0 : There is no relationship between *X* and *Y* (3.12)

versus the *alternative hypothesis* alternative hypothesis *Ha* : There is some relationship between *X* and *Y .* (3.13)

Mathematically, this corresponds to testing

*H*0 : *β*1 = 0

versus

*Ha* : *β*1 *̸*= 0*,*

since if *β*1 = 0 then the model (3.5) reduces to *Y* = *β*0 + *ϵ*, and *X* is not associated with *Y* . To test the null hypothesis, we need to determine whether *β*ˆ1, our estimate for *β*1, is sufficiently far from zero that we can be confident that *β*1 is non-zero. How far is far enough? This of course depends on the accuracy of *β*ˆ1—that is, it depends on SE(*β*ˆ1). If SE(*β*ˆ1) is small, then even relatively small values of *β*ˆ1 may provide strong evidence that *β*1 *̸*= 0, and hence that there is a relationship between *X* and *Y* . In contrast, if SE(*β*ˆ1) is large, then *β*ˆ1 must be large in absolute value in order for us to reject the null hypothesis. In practice, we compute a *t-statistic*, *t*-statistic given by

*t* = *β*ˆ1 *−* 0

SE(*β*ˆ1)*,* (3.14)

which measures the number of standard deviations that *β*ˆ1 is away from 0. If there really is no relationship between *X* and *Y* , then we expect that (3.14) will have a *t*-distribution with *n −* 2 degrees of freedom. The *t*-distribution has a bell shape and for values of *n* greater than approximately 30 it is quite similar to the standard normal distribution. Consequently, it is a simple matter to compute the probability of observing any number equal to *|t|* or larger in absolute value, assuming *β*1 = 0. We call this probability the *p-value*. Roughly speaking, we interpret the *p*-value as follows: a small*p*-value

*p*-value indicates that it is unlikely to observe such a substantial association between the predictor and the response due to chance, in the absence of any real association between the predictor and the response. Hence, if we

68 3. Linear Regression

Coefficient Std. error *t*-statistic *p*-value

Intercept 7.0325 0.4578 15.36 *<* 0*.*0001 TV 0.0475 0.0027 17.67 *<* 0*.*0001

TABLE 3.1. *For the* Advertising *data, coefficients of the least squares model for the regression of number of units sold on TV advertising budget. An increase of* $1*,*000 *in the TV advertising budget is associated with an increase in sales by around 50 units. (Recall that the* sales *variable is in thousands of units, and the* TV *variable is in thousands of dollars.)*

see a small *p*-value, then we can infer that there is an association between the predictor and the response. We *reject the null hypothesis*—that is, we declare a relationship to exist between *X* and *Y* —if the *p*-value is small enough. Typical *p*-value cutoffs for rejecting the null hypothesis are 5% or 1%, although this topic will be explored in much greater detail in Chap ter 13. When *n* = 30, these correspond to *t*-statistics (3.14) of around 2 and 2.75, respectively.

Table 3.1 provides details of the least squares model for the regression of number of units sold on TV advertising budget for the Advertising data. Notice that the coefficients for *β*ˆ0 and *β*ˆ1 are very large relative to their standard errors, so the *t*-statistics are also large; the probabilities of seeing such values if *H*0 is true are virtually zero. Hence we can conclude that *β*0 *̸*= 0 and *β*1 *̸*= 0.4

*3.1.3 Assessing the Accuracy of the Model*

Once we have rejected the null hypothesis (3.12) in favor of the alternative hypothesis (3.13), it is natural to want to quantify *the extent to which the model fits the data*. The quality of a linear regression fit is typically assessed using two related quantities: the *residual standard error* (RSE) and the *R*2*R*2

statistic.

Table 3.2 displays the RSE, the *R*2 statistic, and the *F*-statistic (to be described in Section 3.2.2) for the linear regression of number of units sold on TV advertising budget.

Residual Standard Error

Recall from the model (3.5) that associated with each observation is an error term *ϵ*. Due to the presence of these error terms, even if we knew the true regression line (i.e. even if *β*0 and *β*1 were known), we would not be

4In Table 3.1, a small *p*-value for the intercept indicates that we can reject the null hypothesis that *β*0 = 0, and a small *p*-value for TV indicates that we can reject the null hypothesis that *β*1 = 0. Rejecting the latter null hypothesis allows us to conclude that there is a relationship between TV and sales. Rejecting the former allows us to conclude that in the absence of TV expenditure, sales are non-zero.

3.1 Simple Linear Regression 69

Quantity Value

Residual standard error 3.26

*R*2 0.612

*F*-statistic 312.1

TABLE 3.2. *For the* Advertising *data, more information about the least squares model for the regression of number of units sold on TV advertising budget.*

able to perfectly predict *Y* from *X*. The RSE is an estimate of the standard deviation of *ϵ*. Roughly speaking, it is the average amount that the response will deviate from the true regression line. It is computed using the formula

RSE =

9 1

*n −* 2RSS =

:;;<1

*n −* 2

0*n i*=1

(*yi − y*ˆ*i*)2*.* (3.15)

Note that RSS was defined in Section 3.1.1, and is given by the formula

RSS = 0*n i*=1

(*yi − y*ˆ*i*)2*.* (3.16)

In the case of the advertising data, we see from the linear regression output in Table 3.2 that the RSE is 3*.*26. In other words, actual sales in each market deviate from the true regression line by approximately 3*,*260 units, on average. Another way to think about this is that even if the model were correct and the true values of the unknown coefficients *β*0 and *β*1 were known exactly, any prediction of sales on the basis of TV advertising would still be off by about 3*,*260 units on average. Of course, whether or not 3*,*260 units is an acceptable prediction error depends on the problem context. In the advertising data set, the mean value of sales over all markets is approximately 14*,*000 units, and so the percentage error is 3*,*260*/*14*,*000 = 23 %.

The RSE is considered a measure of the *lack of fit* of the model (3.5) to the data. If the predictions obtained using the model are very close to the true outcome values—that is, if ˆ*yi ≈ yi* for *i* = 1*,...,n*—then (3.15) will be small, and we can conclude that the model fits the data very well. On the other hand, if ˆ*yi* is very far from *yi* for one or more observations, then the RSE may be quite large, indicating that the model doesn’t fit the data well.

*R*2 Statistic

The RSE provides an absolute measure of lack of fit of the model (3.5) to the data. But since it is measured in the units of *Y* , it is not always clear what constitutes a good RSE. The *R*2 statistic provides an alternative measure of fit. It takes the form of a *proportion*—the proportion of variance

70 3. Linear Regression

explained—and so it always takes on a value between 0 and 1, and is independent of the scale of *Y* .

To calculate *R*2, we use the formula

*R*2 = TSS *−* RSS

TSS = 1 *−* RSS

TSS (3.17)

where TSS = )(*yi − y*¯)2 is the *total sum of squares*, and RSS is defined total sum of squares in (3.16). TSS measures the total variance in the response *Y* , and can be thought of as the amount of variability inherent in the response before the regression is performed. In contrast, RSS measures the amount of variability that is left unexplained after performing the regression. Hence, TSS *−* RSS measures the amount of variability in the response that is explained (or removed) by performing the regression, and *R*2 measures the *proportion of variability in Y that can be explained using X*. An *R*2 statistic that is close to 1 indicates that a large proportion of the variability in the response is explained by the regression. A number near 0 indicates that the regression does not explain much of the variability in the response; this might occur because the linear model is wrong, or the error variance *σ*2 is high, or both. In Table 3.2, the *R*2 was 0*.*61, and so just under two-thirds of the variability in sales is explained by a linear regression on TV.

The *R*2 statistic (3.17) has an interpretational advantage over the RSE (3.15), since unlike the RSE, it always lies between 0 and 1. However, it can still be challenging to determine what is a *good R*2 value, and in general, this will depend on the application. For instance, in certain problems in physics, we may know that the data truly comes from a linear model with a small residual error. In this case, we would expect to see an *R*2 value that is extremely close to 1, and a substantially smaller *R*2 value might indicate a serious problem with the experiment in which the data were generated. On the other hand, in typical applications in biology, psychology, marketing, and other domains, the linear model (3.5) is at best an extremely rough approximation to the data, and residual errors due to other unmeasured factors are often very large. In this setting, we would expect only a very small proportion of the variance in the response to be explained by the predictor, and an *R*2 value well below 0*.*1 might be more realistic!

The *R*2 statistic is a measure of the linear relationship between *X* and *Y* . Recall that *correlation*, defined as correlation )*n*

~~5)~~*~~n~~*

*i*=1(*xi − ~~x~~*)(*yi − ~~y~~*)

Cor(*X, Y* ) =

*i*=1(*xi − ~~x~~*)2~~5)~~*~~n~~i*=1(*yi − ~~y~~*)2 *,* (3.18)

is also a measure of the linear relationship between *X* and *Y* .5 This sug gests that we might be able to use *r* = Cor(*X, Y* ) instead of *R*2 in order to

5We note that in fact, the right-hand side of (3.18) is the sample correlation; thus, it would be more correct to write Cor(!*X, Y* ); however, we omit the “hat” for ease of notation.