

# MODERN BUSINESS ANALYTICS

Practical Data Science for Decision Making

Matt TADDY

Leslie HENDRIX

Matthew HARDING

Mc  
Graw  
Hill



# MODERN BUSINESS ANALYTICS

---

**Mc  
Graw  
Hill**





# MODERN BUSINESS ANALYTICS

---

**Practical Data Science for Decision Making**

**Matt Taddy**

*Amazon, Inc.*

**Leslie Hendrix**

*University of South Carolina*

**Matthew C. Harding**

*University of California, Irvine*

**Mc  
Graw  
Hill**





## MODERN BUSINESS ANALYTICS

Published by McGraw Hill Education, 1325 Avenue of the Americas, New York, NY 10019. Copyright ©2023 by McGraw Hill Education. All rights reserved. Printed in the United States of America. No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written consent of McGraw Hill Education, including, but not limited to, in any network or other electronic storage or transmission, or broadcast for distance learning.

Some ancillaries, including electronic and print components, may not be available to customers outside the United States.

This book is printed on acid-free paper.

1 2 3 4 5 6 7 8 9 LKV 27 26 25 24 23 22

ISBN 978-1-264-07167-8 (SE)

MHID 1-264-07167-1 (SE)

ISBN 978-1-264-07165-4 (Loose leaf)

MHID 1-264-07165-5 (Loose leaf)

Portfolio Manager: *Rebecca Olson, Eric Weber*

Product Development Manager: *Michele Janicek*

Product Developer: *Christina Verigan*

Digital Product Developer: *Katherine Ward*

Marketing Manager: *Harper Christopher*

Content Project Managers: *Amy Gehl (Core), Emily Windelborn (Assessment)*

Buyer: *Susan K. Culbertson*

Design: *Matt Diamond*

Content Licensing Specialist: *Melissa Homer*

Cover Image Credit: *MirageC/Getty Images*

Compositor: *Straive*

### Library of Congress Cataloging-in-Publication Data

Names: Taddy, Matt, author.

Title: Modern business analytics: practical data science for  
decision making / Matt Taddy, Amazon, Inc., Leslie Hendrix, University  
of South Carolina, Matthew Harding, University of California, Irvine.

Description: New York, NY: McGraw Hill Education, [2023]

Identifiers: LCCN 2021058431 | ISBN 9781264071678 (hardcover) | ISBN  
9781264071654 (Loose leaf)

Subjects: LCSH: Decision making—Econometric models. | Machine learning.

Classification: LCC HD30.23 .T3245 2023 | DDC 658.4/03—dc23/eng/20220125

LC record available at <https://lccn.loc.gov/2021058431>

The Internet addresses listed in the text were accurate at the time of publication. The inclusion of a website does not indicate an endorsement by the authors or McGraw Hill Education, and McGraw Hill Education does not guarantee the accuracy of the information presented at these sites.



Courtesy of Leslie Hendrix

Courtesy of Matthew  
C. Harding

# BRIEF CONTENTS

<i>About the Authors</i> .....	v
<i>Preface</i> .....	x
<i>Guided Tour</i> .....	xi
<i>Acknowledgments</i> .....	xvii

<b>1</b> Regression .....	1
<b>2</b> Uncertainty Quantification .....	55
<b>3</b> Regularization and Selection .....	100
<b>4</b> Classification .....	151
<b>5</b> Causal Inference with Experiments .....	175
<b>6</b> Causal Inference with Controls .....	215
<b>7</b> Trees and Forests .....	259
<b>8</b> Factor Models .....	282
<b>9</b> Text as Data .....	316
<b>10</b> Deep Learning .....	345
Appendix: R Primer .....	383

Bibliography 419

Glossary 424

Acronyms 433

Index 435

# CONTENTS

About the Authors ..... V

Preface ..... X

*What Is This Book About?*   x

Guided Tour ..... xi

*Practical Data Science for Decision Making*   xi

*An Introductory Example*   xii

*Machine Learning*   xiv

*Computing with R*   xv

Acknowledgments ..... xvii

**1 Regression ..... 1**

    Linear Regression   3

    Residuals   15

    Logistic Regression   21

    Likelihood and Deviance   26

    Time Series   30

    Spatial Data   46

**2 Uncertainty Quantification ..... 55**

    Frequentist Uncertainty   56

    False Discovery Rate Control   67

    The Bootstrap   72

    More on Bootstrap Sampling   86

    Bayesian Inference   91

**3 Regularization and Selection ..... 100**

    Out-of-Sample Performance   101

    Building Candidate Models   108

    Model Selection   130

    Uncertainty Quantification for the Lasso   144

**4 Classification ..... 151**

    Nearest Neighbors   152

    Probability, Cost, and Classification   158

    Classification via Regression   160

    Multinomial Logistic Regression   163



<b>5</b>	<b>Causal Inference with Experiments</b> .....	<b>175</b>
	Notation for Causal Inference	177
	Randomized Controlled Trials	179
	Regression Adjustment	184
	Regression Discontinuity Designs	192
	Instrumental Variables	199
	Design of Experiments	208
<b>6</b>	<b>Causal Inference with Controls</b> .....	<b>215</b>
	Conditional Ignorability	216
	Double Machine Learning	224
	Heterogeneous Treatment Effects	231
	Using Time Series as Controls	241
<b>7</b>	<b>Trees and Forests</b> .....	<b>259</b>
	Decision Trees	261
	Random Forests	268
	Causal Inference with Random Forests	275
	Distributed Computing for Random Forests	277
<b>8</b>	<b>Factor Models</b> .....	<b>282</b>
	Clustering	283
	Factor Models and PCA	290
	Factor Regression	302
	Partial Least Squares	308
<b>9</b>	<b>Text as Data</b> .....	<b>316</b>
	Tokenization	317
	Text Regression	325
	Topic Models	329
	Word Embedding	338
<b>10</b>	<b>Deep Learning</b> .....	<b>345</b>
	The Ingredients of Deep Learning	346
	Working with Deep Learning Frameworks	352
	Stochastic Gradient Descent	367
	The State of the Art	374
	Intelligent Automation	380

**Appendix: R Primer ..... 383**

Getting Started with R 384

Working with Data 396

Advanced Topics for Functions 408

Organizing Code, Saving Work, and Creating Reports 414

Bibliography 419

Glossary 424

Acronyms 433

Index 435

# PREFACE

## What Is This Book About?

The practice of data analytics is changing and modernizing. Innovations in computation and machine learning are creating new opportunities for the data analyst: exposing previously unexplored data to scientific analysis, scaling tasks through automation, and allowing deeper and more accurate modeling. Spreadsheet models and pivot tables are being replaced by code scripts in languages like R and Python. There has been massive growth in digitized information, accompanied by development of systems for storage and analysis of this data. There has also been an intellectual convergence across fields—machine learning and computer science, statistics, and social sciences and economics—that has raised the breadth and quality of applied analysis everywhere. This is the *data science approach to analytics*, and it allows leaders to go deeper than ever to understand their operations, products, and customers.

This book is a primer for those who want to gain the skills to use data science to help make decisions in business and beyond. The modern business analyst uses tools from machine learning, economics, and statistics to not only track what has happened but predict the future for their businesses. Analysts may need to identify the variables important for business policy, run an experiment to measure these variables, and mine social media for information about public response to policy changes. A company might seek to connect small changes in a recommendation system to changes in customer experience and use this information to estimate a demand curve. And any analysis will need to scale to companywide data, be repeatable in the future, and quantify uncertainty about the model estimates and conclusions.

This book focuses on business and economic applications, and we expect that our core audience will be looking to apply these tools as data scientists and analysts inside companies. But we also cover examples from health care and other domains, and the practical material that you learn in this book applies far beyond any narrow set of business problems.

This is not a book about *one of* machine learning, economics, or statistics. Rather, this book pulls from all of these fields to build a toolset for modern business analytics. The material in this book is designed to be useful for *decision making*. Detecting patterns in past data can be useful—we will cover a number of pattern recognition topics—but the necessary analysis for deeper business problems is about *why* things happen rather than what has happened. For this reason, this book will spend the time to move beyond correlation to causal analysis. This material is closer to economics than to the mainstream of data science, which should help you have a bigger practical impact through your work.

We can't cover everything here. This is not an encyclopedia of data analysis. Indeed, for continuing study, there are a number of excellent books covering different areas of contemporary machine learning and data science. For example, Hastie et al. (2009) is a comprehensive modern statistics reference and James et al. (2021) is a less advanced text from a similar viewpoint. You can view this current text as a stepping stone to a career of continued exploration and learning in statistics and machine learning. We want you to leave with a set of best practices that make you confident in what to trust, how to use it, and how to learn more.

# GUIDED TOUR




This book is based on the *Business Data Science* text by Taddy (2019), which was itself developed as part of the MBA data science curriculum at the University of Chicago Booth School of Business. This new adaptation creates a more accessible and course-ready textbook, and includes a major expansion of the examples and content (plus an appendix tutorial on computing with R). Visit [Connect](#) for digital assignments, code, datasets, and additional resources.

## Practical Data Science for Decision Making

Our target readership is anyone who wants to get the skills to use modern large-scale data to make decisions, whether they are working in business, government, science, or anywhere else.

In the past 10 years, we've observed the growth of a class of generalists who can understand business problems and also dive into the (big) data and run their own analyses. There is a massive demand for people with these capabilities, and this book is our attempt to help grow more of these sorts of people. You may be reading this book from a quantitative undergraduate course, as part of your MBA degree at a business school, or in a data science or other graduate program. Or, you may just be reading the book on your own accord. As data analysis has become more crucial and exciting, we are seeing a boom in people switching into data analysis careers from a wide variety of backgrounds. Those self-learners and career-switchers are as much our audience here as students in a classroom.

All of this said, this is not an *easy* book. We have tried to avoid explanations that require calculus or advanced linear algebra, but you will find the book a tough slog if you do not have a solid foundation in first-year mathematics and probability. Since the book includes a breadth of material that spans a range of complexity, we begin each chapter with a summary that outlines each section and indicates their difficulty according to a *ski-hill* scale:

-  The easiest material, requiring familiarity with some transformations like logarithms and exponents, and an understanding of the basics of probability.
-  Moderately difficult material, involving more advanced ideas from probability and statistics or ideas that are going to be difficult to intuit without some linear algebra.
-  The really tough stuff, involving more complex modeling ideas (and notation) and tools from linear algebra and optimization.

The black diamond material is not necessary for understanding future green or blue sections, and so instructors may wish to set their courses to cover the easy and moderately difficult sections while selecting topics from the hardest sections.

The book is designed to be self-contained, such that you can start with little prerequisite background in data science and learn as you go. However, the pace of content on introductory probability and statistics and regression is such that you may struggle if this is your first-ever course on these ideas. If you find this to be the case, we recommend spending some time working through a dedicated introductory statistics book to build some of this understanding before diving into the more advanced data science material.

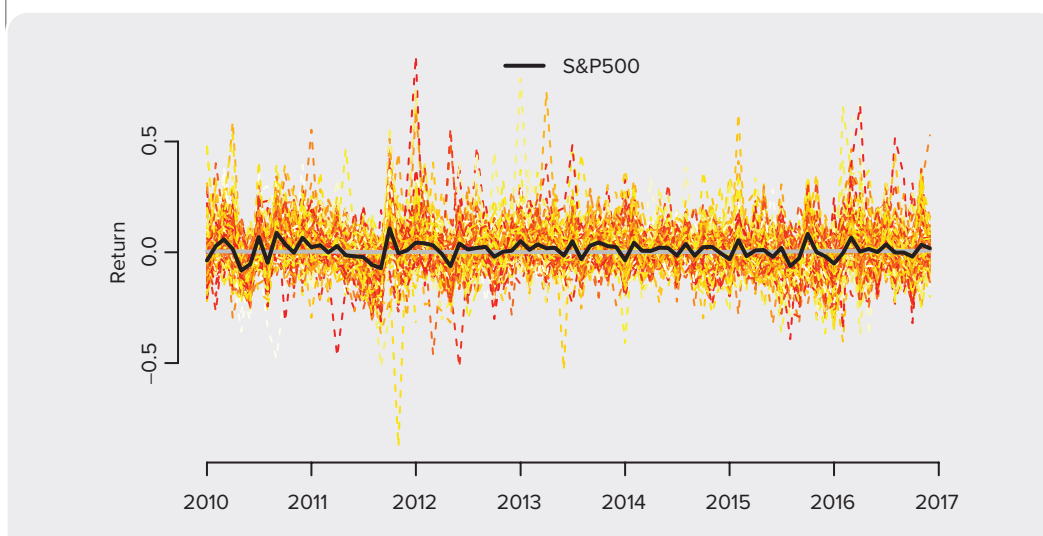
It is also important to recognize that data science can be learned only by doing. This means writing the code to run analysis routines on really messy data. We will use the R scripting language for all of our examples. All example code and data is available online, and one of the most important skills you will get out of this book will be an advanced education in this powerful and widely used statistical software. For those who are completely new to R, we have also included an extensive R primer. The skills you learn here will also prepare you well for learning how to program in other languages, such as Python, which you will likely encounter in your business analysis career.

This is a book about how to *do* modern business analytics. We will lay out a set of core principles and best practices that come from statistics, machine learning, and economics. You will be working through many real data analysis examples as you learn by doing. It is a book designed to prepare scientists, engineers, and business professionals to use data science to improve their decisions.

## An Introductory Example

Before diving into the core material, we will work through a simple finance example to illustrate the difference between data processing or description and a deeper *business analysis*. Consider the graph in Figure 0.1. This shows seven years of monthly returns for stocks in the S&P 500 index (a return is the difference between the current and previous price divided by the prior value). Each line ranging from bright yellow to dark red denotes an individual stock's return series. Their weighted average—the value of the S&P 500—is marked with a bold line. Returns on three-month U.S. treasury bills are in gray.

This is a fancy plot. It looks cool, with lots of different lines. It is the sort of plot that you might see on a computer screen in a TV ad for some online brokerage platform. *If only I had that information, I'd be rich!*



**FIGURE 0.1** A fancy plot: monthly stock returns for members of the S&P 500 and their average (the bold line). *What can you learn?*

But what can you actually learn from Figure 0.1? You can see that returns do tend to bounce around near zero (although the long-term average is reliably much greater than zero). You can also pick out periods of higher volatility (variance) where the S&P 500 changes more from month to month and the individual stock returns around it are more dispersed. That's about it. You don't learn *why* these periods are more volatile or when they will occur in the future. More important, you can't pull out useful information about any individual stock. There is a ton of *data* on the graph but little useful information.

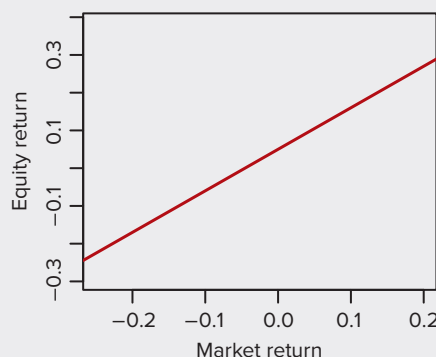
Instead of plotting raw data, let's consider a simple *market model* that relates individual stock returns to the market average. The capital asset pricing model (CAPM) regresses the returns of an individual asset onto a measure of overall market returns, as shown here:

$$r_{jt} = \alpha_j + \beta_j m_t + \varepsilon_{jt} \quad (0.1)$$

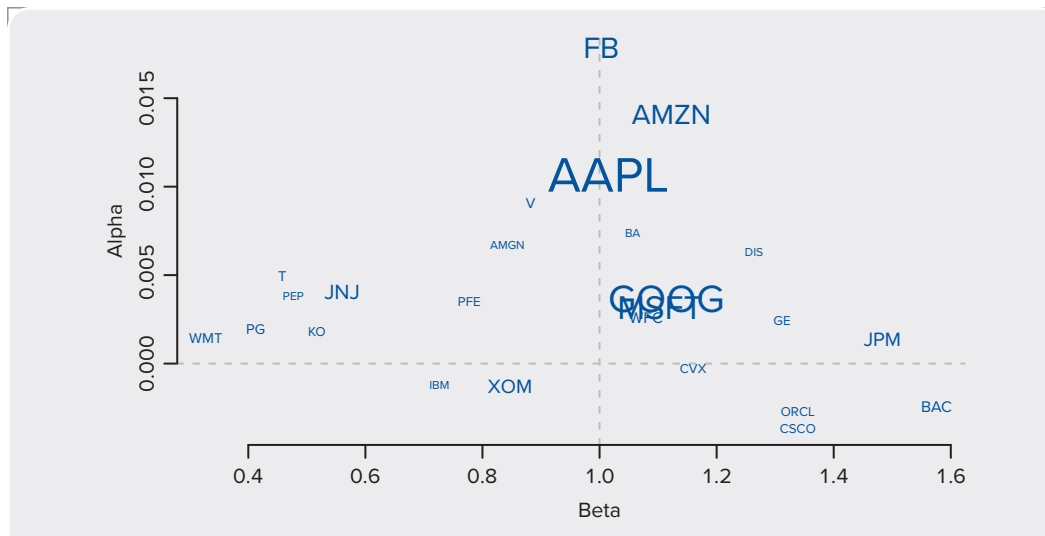
The *output*  $r_{jt}$  is equity  $j$  return at time  $t$ . The *input*  $m_t$  is a measure of the average return—the “market”—at time  $t$ . We take  $m_t$  as the return on the S&P 500 index that weights 500 large companies according to their market capitalization (the total value of their stock). Finally,  $\varepsilon_{jt}$  is an *error* that has mean zero and is uncorrelated with the market.

Equation (0.1) is the first regression model in this book. You'll see many more. This is a simple linear regression that should be familiar to most readers. The Greek letters define a line relating each individual equity return to the market, as shown in Figure 0.2. A small  $\beta_j$ , near zero, indicates an asset with low market sensitivity. In the extreme, fixed-income assets like treasury bills have  $\beta_j = 0$ . On the other hand, a  $\beta_j > 1$  indicates a stock that is more volatile than the market, typically meaning growth and higher-risk stocks. The  $\alpha_j$  is free money: assets with  $\alpha_j > 0$  are adding value regardless of wider market movements, and those with  $\alpha_j < 0$  destroy value.

Figure 0.3 represents each stock “ticker” in the two-dimensional space implied by the market model's fit on the seven years of data in Figure 0.1. The tickers are sized proportional to each firm's market capitalization. The two CAPM parameters— $[\alpha, \beta]$ —tell you a huge amount about the behavior and performance of individual assets. This picture immediately allows you to assess market sensitivity and arbitrage opportunities. For example, the big tech stocks of Facebook (FB), Amazon (AMZN), Apple (AAPL), Microsoft (MSFT), and Google (GOOGL) all have market sensitivity  $\beta$  values close to one. However, Facebook, Amazon, and Apple generated more money independent of the market over this time period compared to Microsoft and Google (which have nearly identical  $\alpha$  values and are overlapped on the plot). Note



**FIGURE 0.2** A scatterplot of a single stock's returns against market returns, with the fitted regression line for the model of Equation (0.1) shown in red.



**FIGURE 0.3** Stocks positioned according to their fitted market model, where  $\alpha$  is money you make regardless of what the market does and  $\beta$  summarizes sensitivity to market movements. The tickers are sized proportional to market capitalization. Production change alpha to  $\alpha$  and beta to  $\beta$  in the plot axis labels.

that Facebook’s CAPM parameters are estimated from a shorter time period, since it did not have its IPO until May of 2012. Some of the older technology firms, such as Oracle (ORCL), Cisco (CSCO), and IBM, appear to have destroyed value over this period (negative alpha). Such information can be used to build portfolios that maximize mean returns and minimize variance in the face of uncertain future market conditions. It can also be used in strategies like pairs-trading where you find two stocks with similar betas and buy the higher alpha while “shorting” the other.

CAPM is an old tool in financial analysis, but it serves as a great illustration of what to strive toward in practical data science. An interpretable model translates raw data into information that is directly relevant to decision making. The challenge in data science is that the data you’ll be working with will be larger and less structured (e.g., it will include text and image data). Moreover, CAPM is derived from assumptions of efficient market theory, and in many applications you won’t have such a convenient simplifying framework on hand. But the basic principles remain the same: you want to turn raw data into useful information that has direct relevance to business policy.

## Machine Learning

Machine learning (ML) is the field of using algorithms to automatically detect and predict patterns in complex data. The rise of machine learning is a major driver behind data science and a big part of what differentiates today’s analyses from those of the past. ML is closely related to modern statistics, and indeed many of the best ideas in ML have come from statisticians. But whereas statisticians have often focused on *model inference*—on understanding the parameters of their models (e.g., testing on individual coefficients in a regression)—the ML community has historically been more focused on the single goal of maximizing *predictive performance* (i.e., predicting future values of some response of interest, like sales or prices).

A focus on prediction tasks has allowed ML to quickly push forward and work with larger and more complex data. If all you care about is predictive performance, then you don't need to worry about whether your model is "true" but rather just test how well it performs when predicting future values. This single-minded focus allows rapid experimentation on alternative models and estimation algorithms. The result is that ML has seen massive success, to the point that you can now expect to have available for almost any type of data an algorithm that will work out of the box to recognize patterns and give high-quality predictions.

However, this focus on prediction means that ML on its own is less useful for many *decision-making* tasks. ML algorithms learn to predict *a future that is mostly like the past*. Suppose that you build an ML algorithm that looks at how customer web browser history predicts how much they spend in your e-commerce store. A purely prediction-focused algorithm will discern what web traffic tends to spend more or less money. It will not tell you what will happen to the spending if you *change* a group of those websites (or your prices) or perhaps make it easier for people to browse the Web (e.g., by subsidizing broadband). That is where this book comes in: we will use tools from economics and statistics in combination with ML techniques to create a platform for using data to make decisions.

Some of the material in this book will be focused on pure ML tasks like prediction and pattern recognition. This is especially true in the earlier chapters on regression, classification, and regularization. However, in later chapters you will use these prediction tools as parts of more structured analyses, such as understanding subject-specific treatment effects, fitting consumer demand functions, or as part of an artificial intelligence system. This typically involves a mix of domain knowledge and analysis tools, which is what makes the data scientist such a powerful figure. The ML tools are useless for policy making without an understanding of the business problems, but a policy maker who can deploy ML as part of their analysis toolkit will be able to make better decisions faster.

## Computing with R

You don't need to be a software engineer to work as a data scientist, but you need to be able to write and understand computer code. To learn from this book, you will need to be able to read and write in a high-level *scripting* language, in other words, flexible code that can be used to describe recipes for data analysis. In particular, you will need to have a familiarity with R ([r-project.org](http://r-project.org)).

The ability to interact with computers in this way—by typing commands rather than clicking buttons or choosing from a menu—is a basic data analysis skill. Having a script of commands allows you to rerun your analyses for new data without any additional work. It also allows you to make small changes to existing scripts to adapt them for new scenarios. Indeed, making small changes is how we recommend you work with the material in this book. The code for every in-text example is available on-line, and you can alter and extend these scripts to suit your data analysis needs. In the examples for this book, all of the analysis will be conducted in R. This is an open-source high-level language for data analysis. R is used widely throughout industry, government, and academia. Companies like RStudio sell enterprise products built around R. This is not a toy language used simply for teaching purposes—R is the real industrial-strength deal.

For the fundamentals of statistical analysis, R is tough to beat: all of the tools you need for linear modeling and uncertainty quantification are mainstays. R is also relatively forgiving for



the novice programmer. A major strength of R is its ecosystem of contributed packages. These are add-ons that increase the capability of core R. For example, almost all of the ML tools that you will use in this book are available via packages. The quality of the packages is more varied than it is for R's core functionality, but if a package has high usage you should be confident that it works as intended.

The Appendix of this book contains a tutorial that is dedicated to getting you started in R. It focuses on the topics and algorithms that are used in the examples in this book. You don't need to be an expert in R to learn from this book; you just need to be able to understand the fundamentals and be willing to mess around with the coded examples. If you have no formal background in coding, worry not: many in the field started out in this position. The learning curve can be steep initially, but once you get the hang of it, the rest will come fast. The tutorial in the Appendix should help you get started. We also provide extensive examples throughout the book, and all code, data, and homework assignments are available through Connect. Every chapter ends with a *Quick Reference* section containing the basic R recipes from that chapter. When you are ready to learn more, there are many great places where you can supplement your understanding of the basics of R. If you simply search for *R* or *R statistics* books on-line, you will find a huge variety of learning resources.

# ACKNOWLEDGMENTS

We are grateful for the reviewers who provided feedback on this first edition:

Sue Abdinour, Wichita State University

Anil Aggarwal, University of Baltimore

Goutam Chakraborty, Oklahoma State  
University

Rick Cleary, Babson College

John Daniels, Central Michigan University

John Draper, Ohio State University

Janet Frasier, West Virginia University

Phillip C. Fry, Boise State University

Marina Girju, California Baptist University

Richard Hauser, East Carolina University

Kuo-Ting “Ken” Hung, Suffolk University

Aimee Jacobs, California State University,  
Fresno

Jaehwan Jeong, Radford University

Patrick Johanns, University of Iowa

Barry King, Butler University

Lauren Kleitz, Xavier University

Su “Sue” Kong, Kutztown University

Min Li, California State University,  
Sacramento

Jiajuan Liang, University of New Haven

Vic Matta, Ohio University

Ebrahim Mortaz, Pace University

Bob Myers, Georgia Tech University

Robert Nauss, University of  
Missouri-St. Louis

Arash Negahban, California State  
University, Chico

Yasin Ozcelik, Fairfield University

Brad Price, West Virginia University

Xingye Qiao, Binghamton University

Roman Rabinovich, Boston University

Bharatendra Rai, UMass-Dartmouth

Balaraman Rajan, California State  
University, East Bay

Rouzbah Razavi, Kent State University

John Repede, Queens University of Charlotte

Thomas R. Robbins, East Carolina  
University

Wendy Swenson Roth, Georgia State  
University

Seokwoo Song, Weber State University

John R. Stevens, Utah State University

Jie Tao, Fairfield University

Vicar S. Valencia, Indiana University South  
Bend

Nikhil Varaiya, San Diego State University

Gang Wang, UMass-Dartmouth

K. Matthew Wong, St. John’s University

Chase Wu, New Jersey Institute of  
Technology

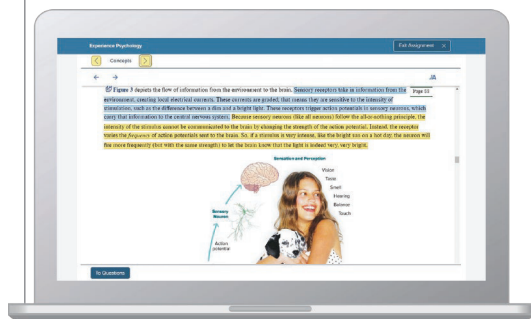
Yajiong “Lucky” Xue, East Carolina  
University

## Instructors: Student Success Starts with You

## Tools to enhance your unique voice

Want to build your own course? No problem. Prefer to use an OLC-aligned, prebuilt course? Easy. Want to make changes throughout the semester? Sure. And you'll save time with Connect's auto-grading too.

**65%**  
Less Time  
Grading



Laptop: McGraw Hill; Woman/dog: George Doyle/Getty Images

## Study made personal

Incorporate adaptive study resources like SmartBook® 2.0 into your course and help your students be better prepared in less time. Learn more about the powerful personalized learning experience available in SmartBook 2.0 at [www.mheducation.com/highered/connect/smartbook](http://www.mheducation.com/highered/connect/smartbook)

**www.mheducation.com/highered/connect/smartbook**

**www.mheducation.com/highered/connect/smartbook**

## Affordable solutions, added value



Make technology work for you with LMS integration for single sign-on access, mobile access to the digital textbook, and reports to quickly show you how each of your students is doing. And with our Inclusive Access program you can provide all these tools at a discount to your students. Ask your McGraw Hill representative for more information.

Padlock: Jobalou/Getty Images

## Solutions for your challenges



A product isn't a solution. Real solutions are affordable, reliable, and come with training and ongoing support when you need it and how you want it. Visit **[www.supportateverystep.com](http://www.supportateverystep.com)** for videos and resources both you and your students can use throughout the semester.

Checkmark: Jobalou/Getty Images

## Students: Get Learning that Fits You

### Effective tools for efficient studying

Connect is designed to help you be more productive with simple, flexible, intuitive tools that maximize your study time and meet your individual learning needs. Get learning that works for you with Connect.

### Study anytime, anywhere

Download the free ReadAnywhere app and access your online eBook, SmartBook 2.0, or Adaptive Learning Assignments when it's convenient, even if you're offline. And since the app automatically syncs with your Connect account, all of your work is available every time you open it. Find out more at [www.mheducation.com/readanywhere](http://www.mheducation.com/readanywhere)

*"I really liked this app—it made it easy to study when you don't have your textbook in front of you."*

- Jordan Cunningham,  
Eastern Washington University



Calendar: owattaphotos/Getty Images

### Everything you need in one place

Your Connect course has everything you need—whether reading on your digital eBook or completing assignments for class, Connect makes it easy to get your work done.

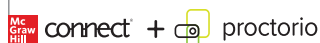
### Learning for everyone

McGraw Hill works directly with Accessibility Services Departments and faculty to meet the learning needs of all students. Please contact your Accessibility Services Office and ask them to email [accessibility@mheducation.com](mailto:accessibility@mheducation.com), or visit [www.mheducation.com/about/accessibility](http://www.mheducation.com/about/accessibility) for more information.

Top: Jenner Images/Getty Images, Left: Hero Images/Getty Images, Right: Hero Images/Getty Images



## Proctorio Remote Proctoring & Browser-Locking Capabilities



Remote proctoring and browser-locking capabilities, hosted by Proctorio within Connect, provide control of the assessment environment by enabling security options and verifying the identity of the student.

Seamlessly integrated within Connect, these services allow instructors to control students' assessment experience by restricting browser activity, recording students' activity, and verifying students are doing their own work.

Instant and detailed reporting gives instructors an at-a-glance view of potential academic integrity concerns, thereby avoiding personal bias and supporting evidence-based claims.



## ReadAnywhere

Read or study when it's convenient for you with McGraw Hill's free ReadAnywhere app. Available for iOS or Android smartphones or tablets, ReadAnywhere gives users access to McGraw Hill tools including the eBook and SmartBook 2.0 or Adaptive Learning Assignments in Connect. Take notes, highlight, and complete assignments offline—all of your work will sync when you open the app with WiFi access. Log in with your McGraw Hill Connect username and password to start learning—anytime, anywhere!

## OLC-Aligned Courses

### **Implementing High-Quality Online Instruction and Assessment through Preconfigured Courseware**

In consultation with the Online Learning Consortium (OLC) and our certified Faculty Consultants, McGraw Hill has created pre-configured courseware using OLC's quality scorecard to align with best practices in online course delivery. This turnkey courseware contains a combination of formative assessments, summative assessments, homework, and application activities, and can easily be customized to meet an individual's needs and course outcomes. For more information, visit <https://www.mheducation.com/highered/olc>.

## Tegrity: Lectures 24/7

Tegrity in Connect is a tool that makes class time available 24/7 by automatically capturing every lecture. With a simple one-click start-and-stop process, you capture all computer screens and corresponding audio in a format that is easy to search, frame by frame. Students can replay any part of any class with easy-to-use, browser-based viewing on a PC, Mac, iPod, or other mobile device.

Educators know that the more students can see, hear, and experience class resources, the better they learn. In fact, studies prove it. Tegrity's unique search feature helps students

efficiently find what they need, when they need it, across an entire semester of class recordings. Help turn your students' study time into learning moments immediately supported by your lecture. With Tegrity, you also increase intent listening and class participation by easing students' concerns about note-taking. Using Tegrity in Connect will make it more likely you will see students' faces, not the tops of their heads.

## Test Builder in Connect

Available within Connect, Test Builder is a cloud-based tool that enables instructors to format tests that can be printed, administered within a Learning Management System, or exported as a Word document of the test bank. Test Builder offers a modern, streamlined interface for easy content configuration that matches course needs, without requiring a download.

Test Builder allows you to:

- access all test bank content from a particular title.
- easily pinpoint the most relevant content through robust filtering options.
- manipulate the order of questions or scramble questions and/or answers.
- pin questions to a specific location within a test.
- determine your preferred treatment of algorithmic questions.
- choose the layout and spacing.
- add instructions and configure default settings.

Test Builder provides a secure interface for better protection of content and allows for just-in-time updates to flow directly into assessments.

## Writing Assignment

Available within Connect and Connect Master, the Writing Assignment tool delivers a learning experience to help students improve their written communication skills and conceptual understanding. As an instructor you can assign, monitor, grade, and provide feedback on writing more efficiently and effectively.

## Application-Based Activities in Connect

Application-Based Activities in Connect are highly interactive, assignable exercises that provide students a safe space to apply the concepts they have learned to real-world, course-specific problems. Each Application-Based Activity involves the application of multiple concepts, allowing students to synthesize information and use critical thinking skills to solve realistic scenarios.

 **create<sup>®</sup>** Your Book, Your Way

McGraw Hill's Content Collections Powered by Create<sup>®</sup> is a self-service website that enables instructors to create custom course materials—print and eBooks—by drawing upon

McGraw Hill’s comprehensive, cross-disciplinary content. Choose what you want from our high-quality textbooks, articles, and cases. Combine it with your own content quickly and easily, and tap into other rights-secured, third-party content such as readings, cases, and articles. Content can be arranged in a way that makes the most sense for your course and you can include the course name and information as well. Choose the best format for your course: color print, black-and-white print, or eBook. The eBook can be included in your Connect course and is available on the free ReadAnywhere app for smartphone or tablet access as well. When you are finished customizing, you will receive a free digital copy to review in just minutes! Visit McGraw Hill Create®—[www.mcgrawhillcreate.com](http://www.mcgrawhillcreate.com)—today and begin building!



# 1 REGRESSION

This chapter develops the framework and language of regression: building models that predict response outputs from feature inputs.

- **Section 1.1 Linear Regression:** Specify, estimate, and predict from a linear regression model for a quantitative response  $y$  as a function of inputs  $\mathbf{x}$ . Use log transforms to model multiplicative relationships and *elasticities*, and use interactions to allow the effect of inputs to depend on each other.
- **Section 1.2 Residuals:** Calculate the residual errors for your regression fit, and understand the key fit statistics *deviance*,  $R^2$ , and *degrees of freedom*.
- **Section 1.3 Logistic Regression:** Build logistic regression models for a binary response variable, and understand how logistic regression is related to linear regression as a *generalized linear model*. Translate the concepts of deviance, likelihood, and  $R^2$  to logistic regression, and be able to interpret logistic regression coefficients as effects on the log odds that  $y = 1$ .
- **Section 1.4 Likelihood and Deviance:** Relate likelihood maximization and deviance minimization, use the generalized linear models to determine residual deviance, and use the `predict` function to integrate new data with the same variable names as the data used to fit your regression.
- **Section 1.5 Time Series:** Adapt your regression models to allow for dependencies in data that has been observed over time, and understand time series concepts including seasonal trends, autoregression, and panel data.
- ◆ **Section 1.6 Spatial Data:** Add spatial fixed effects to your regression models and use Gaussian process models to estimate spatial dependence in your observations.



The vast majority of problems in applied data science require regression modeling. You have a *response* variable ( $y$ ) that you want to model or predict as a function of a vector of *input features*, or covariates ( $\mathbf{x}$ ). This chapter introduces the basic framework and language of regression. We will build on this material throughout the rest of the book.

Regression is all about understanding the *conditional* probability distribution for “ $y$  given  $\mathbf{x}$ ,” which we write as  $p(y|\mathbf{x})$ . Figure 1.1 illustrates the conditional distribution in contrast to a *marginal* distribution, which is so named because it corresponds to the unconditional distribution for a single margin (i.e., column) of a data matrix.

A variable that has a probability distribution (e.g., number of bathrooms in Figure 1.1) is called a *random variable*. The *mean* for a random variable is the average of random draws from its probability distribution. While the marginal mean is a simple number, the conditional mean is a function. For example, from Figure 1.1b, you can see that the average home selling price takes different values indexed by the number of bathrooms. The data is distributed randomly around these means, and the way that you model these distributions drives your estimation and prediction strategies.

## Conditional Expectation

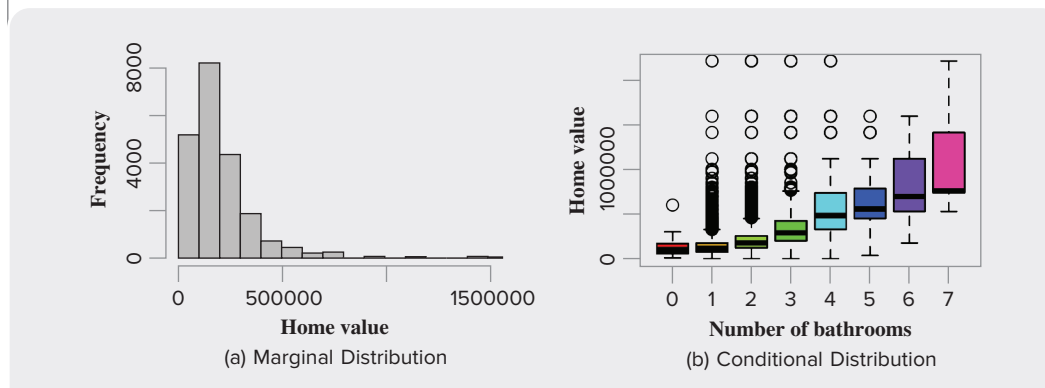
A basic but powerful regression strategy is to build models in terms of *averages* and *lines*. That is, we will model the conditional mean for our output variable as a linear function of inputs. Other regression strategies can sometimes be useful, such as *quantile regression* that models percentiles of the conditional distribution. However for the bulk of applications you will find that *mean* regression is a good approach.

There is some important notation that you need to familiarize yourself with for the rest of the book. We model the conditional mean for  $y$  given  $\mathbf{x}$  as

$$\mathbb{E}[y|\mathbf{x}] = f(\mathbf{x}'\boldsymbol{\beta}) \quad (1.1)$$

where

- $\mathbb{E}[\cdot]$  denotes the taking of the expectation or average of whatever random variable is inside the brackets. It is an extremely important operation, and we will use this notation to define many of our statistical models.



**FIGURE 1.1** Illustration of marginal versus conditional distributions for home prices. On the left, we have the marginal distribution for all of the home prices. On the right, home price distributions are conditional on the number of bathrooms.

- The vertical bar  $|$  means “given” or “conditional upon,” so that  $\mathbb{E}[y|\mathbf{x}]$  is read as “the average for  $y$  given inputs  $\mathbf{x}$ .”
- $f(\cdot)$  is a “link” function that transforms from the linear model to your response.
- $\mathbf{x} = [1, x_1, x_2, \dots, x_p]$  is the vector of covariates and  $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2, \dots, \beta_p]$  are the corresponding coefficients.

The *vector* notation,  $\mathbf{x}'\boldsymbol{\beta}$ , is shorthand for the sum of elementwise products:

$$\mathbf{x}'\boldsymbol{\beta} = [1x_1x_2\cdots x_p] \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} = \beta_0 + x_1\beta_1 + x_2\beta_2 + \dots + x_p\beta_p \quad (1.2)$$

This shorthand notation will be used throughout the book. Here we have used the convention that  $x_0 = 1$ , such that  $\beta_0$  is the intercept.

The link function,  $f(\cdot)$ , defines the relationship between your linear function  $\mathbf{x}'\boldsymbol{\beta}$  and the response. The link function gives you a huge amount of modeling flexibility. This is why models of the kind written in Equation (1.1) are called *generalized linear models* (GLMs). They allow you to make use of linear modeling strategies after some simple transformations of your output variable of interest. In this chapter we will outline the two most common GLMs: linear regression and logistic regression. These two models will serve you well for the large majority of analysis problems, and through them you will become familiar with the general principles of GLM analysis.

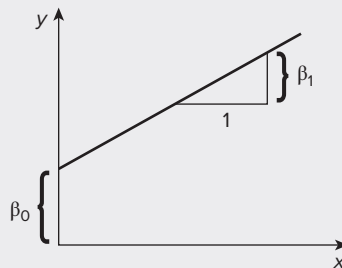
## 1.1 Linear Regression

Linear regression is the workhorse of analytics. It is fast to fit (in terms of both analyst and computational time), it gives reasonable answers in a variety of settings (so long as you know how to ask the right questions), and it is easy to interpret and understand. The model is as follows:

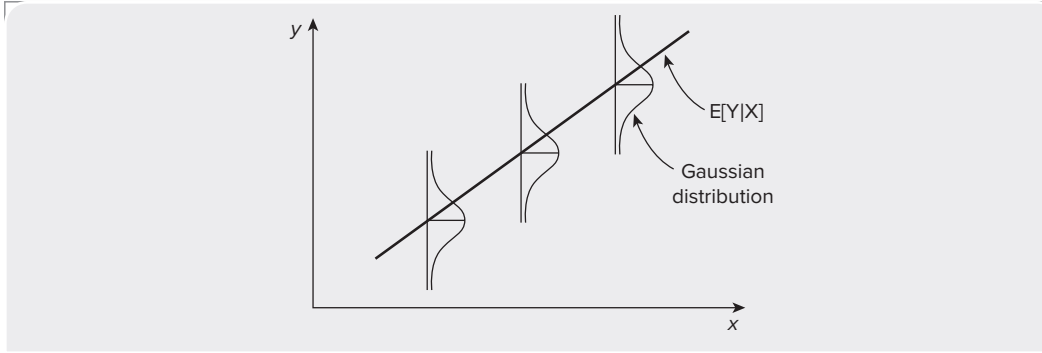
$$\mathbb{E}[y|\mathbf{x}] = \beta_0 + x_1\beta_1 + x_2\beta_2 + \dots + x_p\beta_p \quad (1.3)$$

This corresponds to using the link function  $f(z) = z$  in Equation (1.1).

With just one input  $x$ , you can write the model as  $\mathbb{E}[y|x] = \beta_0 + x\beta_1$  and plot it as in Figure 1.2.  $\beta_0$  is the intercept. This is where the line crosses the  $y$  axis when  $x$  is 0.  $\beta_1$  is the



**FIGURE 1.2** Simple linear regression with a positive slope  $\beta_1$ . The plotted line corresponds to  $\mathbb{E}[y|x]$ .



**FIGURE 1.3** Using simple linear regression to picture the Gaussian conditional distribution for  $y|x$ . Here  $E[y|x]$  are the values on the line and the variation parallel to the  $y$  axis (i.e., within each narrow vertical strip) is assumed to be described by a Gaussian distribution.

slope and describes how  $E[y|x]$  changes as  $x$  changes. If  $x$  increases by 1 unit,  $E[y|x]$  changes by  $\beta_1$ . For a two predictor model, we are fitting a plane. Higher dimensions are more difficult to imagine, but the basic intuition is the same.

When fitting a regression model—i.e., when estimating the  $\beta$  coefficients—you make some assumptions about the conditional distribution beyond its mean at  $E[y|x]$ . Linear regression is commonly fit for Gaussian (normal) conditional distributions. We write this conditional distribution as

$$y | \mathbf{x} \sim N(\mathbf{x}'\beta, \sigma^2) \quad (1.4)$$

This says that the distribution for  $y$  as a function of  $\mathbf{x}$  is normally distributed around  $E[y|\mathbf{x}] = \mathbf{x}'\beta$  with variance  $\sigma^2$ . The same model is often written with an additive error term:

$$y = \mathbf{x}'\beta + \varepsilon, \varepsilon \sim N(0, \sigma^2) \quad (1.5)$$

where  $\varepsilon$  are the “independent” or “idiosyncratic” errors. These errors contain the variations in  $y$  that are not correlated with  $\mathbf{x}$ . Equations (1.4) and (1.5) describe the same model. Figure 1.3 illustrates this model for single-input simple linear regression. The line is the average  $E[y|x]$  and vertical variation around the line is what is assumed to have a normal distribution.

You will often need to transform your data to make the linear model of Equation (1.5) realistic. One common transform is that you need to take a *logarithm* of the response, say, “ $r$ ,” such that your model becomes

$$\log(r) = \mathbf{x}'\beta + \varepsilon, \varepsilon \sim N(0, \sigma^2) \quad (1.6)$$

Of course this is the same as the model in Equation (1.5), but we have just made the replacement  $y = \log(r)$ . You will likely also consider transformations for the input variables, such that elements of  $\mathbf{x}$  include logarithmic and other functional transformations. This is often referred to as *feature engineering*.

**Example 1.1 Orange Juice Sales: Exploring Variables and the Need for a log-log Model** As a concrete example, consider sales data for orange juice (OJ) from Dominick’s grocery stores. Dominick’s was a Chicago-area chain. This data was collected in the 1990s and is publicly available from the Kilts Center at the University of Chicago’s Booth School of Business. The data includes weekly prices and unit sales (number of cartons sold) for three OJ

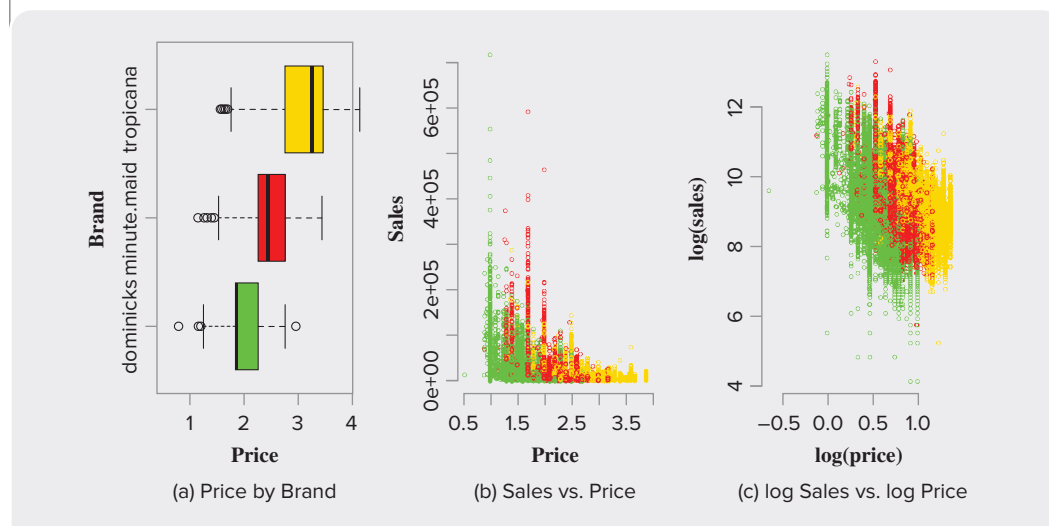
brands—Tropicana, Minute Maid, Dominick’s—at 83 stores in the Chicago area, as well as an indicator, `ad`, showing whether each brand was advertised (in store or flyer) that week.

```
> oj <- read.csv("oj.csv", strings=T)
> head(oj)
  sales price   brand ad
1  8256  3.87 tropicana 0
2  6144  3.87 tropicana 0
3  3840  3.87 tropicana 0
4  8000  3.87 tropicana 0
5  8896  3.87 tropicana 0
6  7168  3.87 tropicana 0
> levels(oj$brand)
[1] "dominicks" "minute.maid" "tropicana"
```

Notice the argument `strings=T` in `read.csv` as shorthand for “`stringsAsFactors = TRUE`.” This converts our `brand` column into a factor variable. This was the default behavior of `read.csv` prior to version 4.0.0 of R, but you now need to specify it explicitly. Otherwise you will get an error when you try to make the plots or fit the regression models below.

The code-printout above is our first example showing R code and output. We will include a ton of code and output snippets like this throughout the book: they are an integral part of the material. If this output looks unfamiliar to you, you should break here and take the time to work through the R-primer in the Appendix.

Figure 1.4 shows the prices and sales broken out by brand. You can see in Figure 1.4a that each brand occupies a different price range: Dominick’s is the budget option, Tropicana is the luxury option, and Minute Maid lives between.



**FIGURE 1.4** Dominick’s OJ prices by brand and monthly sales, both raw and after a log transformation.

price. This makes sense: demand is *downward* sloping, and if you charge more, you sell less. More specifically, it appears that *log* sales has a roughly linear relationship with *log* price. This is an important point. Whenever you are working with linear (i.e., additive) models, it is crucial that you try to work in the space where you expect to find linearity. For variables that change *multiplicatively* with other factors, this is usually the log scale (see the nearby box for a quick review on logarithms). For comparison, the raw (without log) values in Figure 1.4b show a nonlinear relationship between prices and sales.

### log-log Models and Elasticities

Another common scenario models against each other two variables that *both* move multiplicatively. For example, Figure 1.5 shows the national gross domestic product (GDP) versus imports for several countries. Fitting a line to the left panel would be silly; its slope will be entirely determined by small changes in the U.S. values. In contrast, the right panel shows that GDP and imports follow a neat linear relationship in log space.

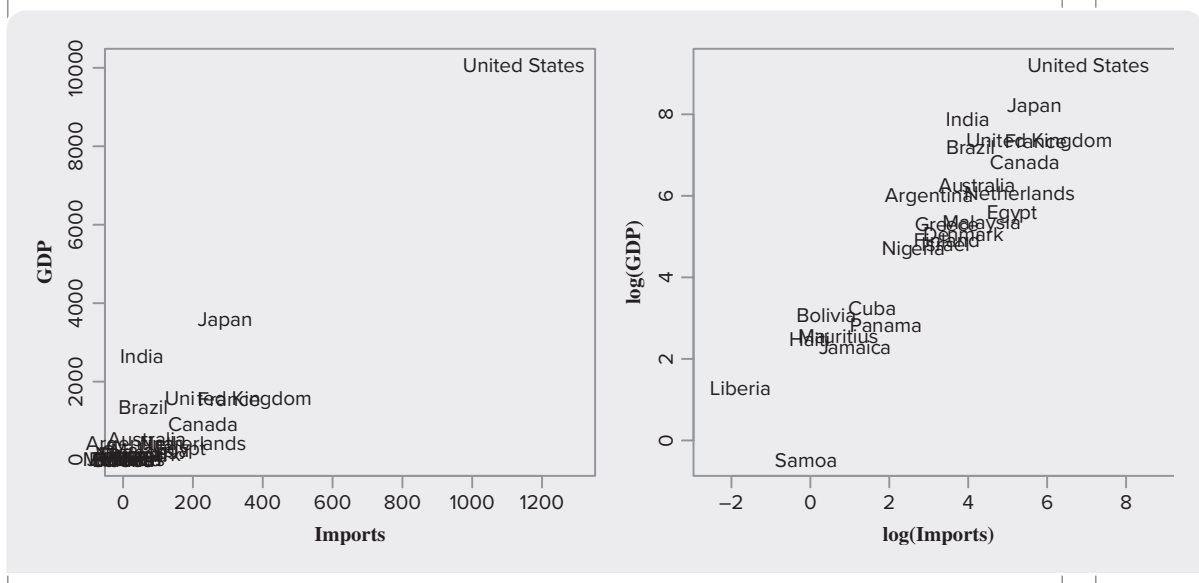
Returning to our OJ example, Figure 1.4c indicates that this *log-log* model might be appropriate for the orange juice sales versus price analysis. One possible regression model is

$$\log(\text{sales}) = \beta_0 + \beta_1 \log(\text{price}) + \varepsilon \quad (1.7)$$

Here,  $\log(\text{sales})$  increase by  $\beta_1$  for every unit increase in  $\log(\text{price})$ . Conveniently, log-log models have a much more intuitive interpretation: sales increase by  $\beta_1\%$  for every 1% increase in price. To see this, you need a bit of calculus. Write  $y = \exp[\beta_0 + \beta_1 \log(x) + \varepsilon]$  and differentiate with respect to  $x$ :

$$\frac{dy}{dx} = \frac{\beta_1}{x} e^{\beta_0 + \beta_1 \log(x) + \varepsilon} = \frac{\beta_1}{x} y \Rightarrow \beta_1 = \frac{dy/y}{dx/x} \quad (1.8)$$

This shows that  $\beta_1$  is the proportional change in  $y$  over the proportional change in  $x$ . In economics there is a special name for such an expression: *elasticity*. The concept of elasticity will play an important role in many of our analyses.



**FIGURE 1.5** National GDP against imports, in original and log scale.



fits only linear regression models, so you could use that here also (it takes the same arguments), but we will get in the habit of using `glm` since it works for many different GLMs. The function is straightforward to use: you give it a data frame with the `data` argument and provide a formula that defines your regression.

```
> fit <- glm( y ~ var1 + ... + varP, data=mydata)
```

The fitted object `fit` is a list of useful things (type `names(fit)` to see them), and there are functions to access the results. For example,

- `summary(fit)` prints the model, information about residual errors, the estimated coefficients and uncertainty about these estimates (we will cover the uncertainty in detail in the next chapter), and statistics related to model fit.
- `coef(fit)` supplies just the coefficient estimates.
- `predict(fit, newdata=mynewdata)` predicts  $y$  where `mynewdata` is a data frame with the same variables as `mydata`.

The formula syntax in the `glm` call is important. The `~` symbol is read as “regressed onto” or “as a function of.” The variable you want to predict, the  $y$  response variable, comes before the `~`, and the input features,  $x$ , come after. This model formula notation will be used throughout the remainder of the book, and we note some common specifications in Table 1.1.

The R formula for (1.13) is `log(sales) ~ brand + log(price)`. You can fit this with `glm` using the `oj` data, and then use the `coef` function to view the fitted coefficients. (More on this in Section 1.4.)

```
> fit<-glm( log(sales) ~ brand + log(price), data=oj)
> coef(fit) # fitted coefficients
      (Intercept) brandminute.maid  brandtropicana  log(price)
      10.8288216      0.8701747      1.5299428      -3.1386914
```

There are a few things to notice here. First, you can see that  $\hat{\beta} = -3.1$  for the estimated coefficient on log price. Throughout this book we use the convention that  $\hat{\theta}$  denotes the estimated value for some parameter  $\theta$ . So  $\hat{\beta}$  is the estimated “sales-price elasticity,” and it says that expected sales drop by about 3% for every 1% price increase. Second, notice that there are distinct model coefficients for Minute Maid and Tropicana but not for Dominick’s. This is due

$y \sim x_1$	model by $x_1$
$y \sim .$	include all other columns
$y \sim .-x_3$	include all except $x_3$
$y \sim .-1$	include all, but no intercept
$y \sim 1$	intercept only
$y \sim x_1*x_2$	include interaction for $x_1$ and $x_2$ and lower order terms
$y \sim x_1:x_2$	include interaction only
$y \sim .^2$	all possible 2 way interactions and lower order terms

**TABLE 1.1** Some common syntax for use in formulas.





The first step of `glm` is to create a *model matrix* (also called a *design matrix*) that defines the numeric inputs  $\mathbf{x}$ . It does this with a call to the `model.matrix` function, and you can pull that step out to see what happens.

```
> x <- model.matrix( ~ log(price) + brand, data=oj)
> x[c(100,200,300),]
      (Intercept)  log(price) brandminute.maid brandtropicana
100             1    1.1600209             0             1
200             1    1.0260416             1             0
300             1    0.3293037             0             0
```

The `model.matrix` function has expanded these brand factor levels into a couple of binary, or “dummy,” variables that are one when the observation is from that brand and zero otherwise. For example, `brandtropicana` is 1 for the Tropicana observation in row 100 and zero otherwise. There is no `branddominicks` indicator because you need only two variables to represent three categories: when both `brandminute.maid` and `brandtropicana` are zero, the *intercept* gives the value for Dominick’s expected log sales at a log price of zero. Each factor’s reference level is absorbed by the intercept and the other coefficients represent “change relative to reference” (here, Dominick’s). To check the reference level of your factors, type `levels(myfactor)`. The first level is the reference and by default this will be the first in the alphabetized list of levels. To change this, you can do `myfactor = relevel(myfactor, “myref”)`.

## 1.1.1 Interactions

All of the lines in Figure 1.6 have the same slope. In economic terms, the model assumes that consumers of the three brands have the same price sensitivity. This seems unrealistic: money is probably less of an issue for Tropicana customers than it is for the average Dominick’s consumer. You can build this information into your regression by having log price *interact* with brand.

An interaction term is the product of two inputs. Including an interaction between, say,  $x_j$  and  $x_k$  inputs, implies that your linear regression equation includes the product  $x_j x_k$  as an input.

$$\mathbb{E}[y|\mathbf{x}] = \beta_0 + \dots + \beta_k x_k + \beta_j x_j + x_j x_k \beta_{jk} \quad (1.15)$$

Here, “...” just denotes whatever else is in your multiple linear regression model. Equation (1.15) says that the effect on the expected value for  $y$  due to a unit increase in  $x_j$  is  $\beta_j + x_k \beta_{jk}$ , such that it depends upon  $x_k$ .

Interactions are central to scientific and business questions. For example,

- How does drug effectiveness change with patient age?
- Does gender change the effect of education on wages?
- How does consumer price sensitivity change across brands?

In each case here, you want to know whether one variable changes the effect of another. You don't want to know the average price sensitivity of customers; you want to know whether they are more price sensitive for one product versus another.

**Example 1.3 OJ Sales: Interaction** In the OJ sales regression, to get brand-specific price elasticity terms you need to include an interaction between each of the brand indicator terms and the log price. We can write this as a model with a separate intercept and slope for each brand:

$$\log(\text{sales}) = \alpha_{\text{brand}} + \beta_{\text{brand}} \log(\text{price}) + \varepsilon \quad (1.16)$$

We can also expand this notation out to write the exact model that `glm` will be estimating:

$$\begin{aligned} \log(\text{sales}) = & \alpha_0 + \alpha_1 \mathbb{1}_{[\text{minute.maid}]} + \alpha_2 \mathbb{1}_{[\text{tropicana}]} + \\ & (\beta_0 + \beta_1 \mathbb{1}_{[\text{minute.maid}]} + \beta_2 \mathbb{1}_{[\text{tropicana}]}) \log(\text{price}) + \varepsilon \end{aligned} \quad (1.17)$$

As before, `dominicks` is the reference level for brand and so it is absorbed into both the intercept and baseline slope on log price. For an observation from Dominick's, the indicator functions are all zero so that  $\mathbb{E}[\log(\text{sales})] = \alpha_0 + \beta_0 \log(\text{price})$ .

You can fit this model in `glm` with the `*` symbol, which is syntax for "interacted with." Note that `*` also adds the *main effects*—all of the terms from our earlier model in Equation (1.13).

```
> fit2way <- glm(log(sales) ~ log(price)*brand, data=oj)
> coef(fit2way)
```

(Intercept)	log(price)
10.95468173	-3.37752963
brandminute.maid	brandtropicana
0.88825363	0.96238960
log(price):brandminute.maid	log(price):brandtropicana
0.05679476	0.66576088

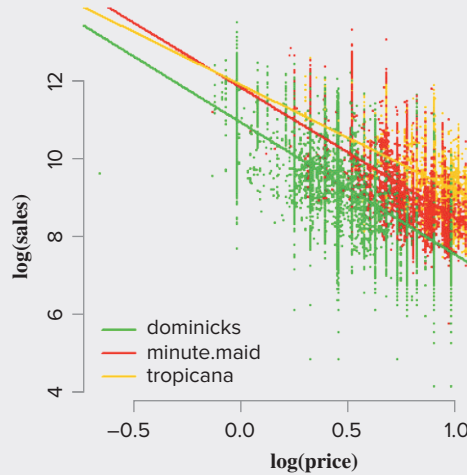
The fitted regression is pictured in Figure 1.7.

In the `glm` output, the `log(price):brand` coefficients are the interaction terms. Plugging in 0 for both Tropicana and Minute Maid indicators yields the equation of the line for Dominick's:

$$\mathbb{E}[\log(\text{sales})] = 10.95 - 3.38 \log(\text{price})$$

Plugging in one for Minute Maid terms and zero for Tropicana terms yields the equation for Minute Maid:

$$\begin{aligned} \mathbb{E}[\log(\text{sales})] &= 10.95 - 3.38 \log(\text{price}) + 0.89 + 0.06 \log(\text{price}) \\ &= 11.84 - 3.32 \log(\text{price}) \end{aligned}$$



**FIGURE 1.7** Fit for the model where we allow interaction between price and brand. Note that if you extrapolate too far, the linearity assumption implies Tropicana selling less than Minute Maid at the same price. This is a reminder that linear models are approximations and should be used with care away from the center of the observed data.

And plugging in one for Tropicana and zero for Minute Maid yields the regression line for Tropicana:

$$\begin{aligned}\mathbb{E}[\log(\text{sales})] &= 10.95 - 3.38 \log(\text{price}) + 0.96 + 0.67 \log(\text{price}) \\ &= 11.91 - 2.71 \log(\text{price})\end{aligned}$$

We see that Tropicana customers are indeed less sensitive than the others: they have a sales-price elasticity of  $-2.7$  versus around  $-3.3$  for both Dominick's and Minute Maid. This means, for example, that the store should expect a smaller sales increase for price cuts or coupons on Tropicana relative to use of the same promotion on the other brands. The price sensitivity that we estimated for model Equation (1.13),  $-3.1$ , was the result of averaging across the three distinct brand elasticities.

### Advertising and Price Elasticity

We conclude this introduction to linear regression—and the study of orange juice—with a look at the role of advertising in the relationship between sales and prices. Recall that the OJ data includes an ad dummy variable, indicating that a given brand was promoted with either an in-store display or a flier ad during the week that sales and prices were recorded. The ads can increase sales at all prices, they can change price sensitivity, and they can do both of these things in a brand-specific manner. To model this, we specify a three-way interaction between price, brand, and ad:

$$\log(\text{sales}) = \alpha_{\text{brand, ad}} + \beta_{\text{brand, ad}} \log(\text{price}) + \varepsilon \quad (1.18)$$

By subsetting on brand, ad we are indicating that there are different intercepts and slopes for each combination of the two factors. To fit this model with `glm`, you interact brand, ad, and `log(price)` with each other in the formula. Again, `glm` automatically includes the lower-level

interactions and main effects—all of the terms from our model in (1.18)—in addition to the new three-way interactions.

```
> fit3way <- glm(log(sales) ~ log(price)*brand*ad, data=oj)
> coef(fit3way)
```

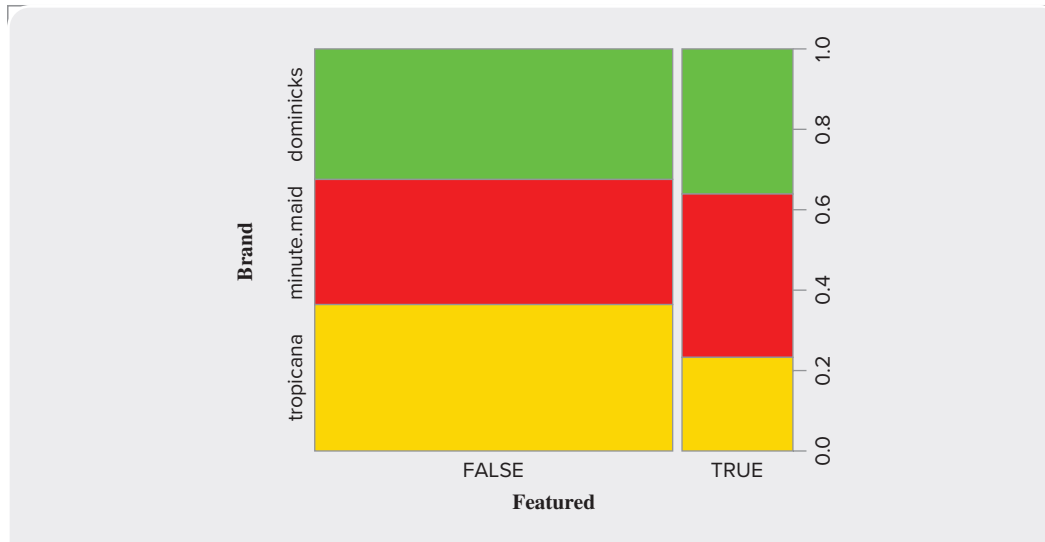
(Intercept)	log(price)
10.40657579	-2.77415436
brandminute.maid	brandtropicana
0.04720317	0.70794089
ad	log(price):brandminute.maid
1.09440665	0.78293210
log(price):brandtropicana	log(price):ad
0.73579299	-0.47055331
brandminute.maid:ad	brandtropicana:ad
1.17294361	0.78525237
log(price):brandminute.maid:ad	log(price):brandtropicana:ad
-1.10922376	-0.98614093

The brand and ad specific elasticities are compiled in Table 1.2. We see that being featured always leads to more price sensitivity. Minute Maid and Tropicana elasticities drop from  $-2$  to below  $-3.5$  with ads, while Dominick's drops from  $-2.8$  to  $-3.2$ . Why does this happen? One possible explanation is that advertisement increases the population of consumers who are considering your brand. In particular, it can increase your market beyond brand loyalists, to people who will be more price sensitive than those who reflexively buy your orange juice every week. Indeed, if you observe increased price sensitivity, it can be an indicator that your marketing efforts are expanding your consumer base. This is why Marketing 101 dictates that ad campaigns should usually be accompanied by price cuts. There is also an alternative explanation. Since the featured products are often also discounted, it could be that at lower price points the average consumer is more price sensitive (i.e., that the price elasticity is also a function of price). The truth is probably a combination of these effects.

Finally, notice that in our two-way interaction model (without including ad) Minute Maid's elasticity of  $-3.3$  was roughly the same as Dominick's—it behaved like a budget product where its consumers are focused on value. However, in Table 1.2, you can see that Minute Maid and Tropicana have nearly identical elasticities and that both are different from Dominick's. Minute Maid is looking more similar now to the other national brand product. What happened?

	Dominick's	Minute Maid	Tropicana
Not featured	$-2.8$	$-2.0$	$-2.0$
Featured	$-3.2$	$-3.6$	$-3.5$

**TABLE 1.2** Brand and ad dependent elasticities. Test that you can recover these numbers from the R output.



**FIGURE 1.8** A mosaic plot of the amount of advertisement by brand. In a mosaic plot, the size of the boxes is proportional to the amount of data contained in that category. For example, the plot indicates that most sales are not accompanied by advertising (the featured=FALSE column is wider than for featured=TRUE) and that Minute Maid is featured (i.e.,  $ad=1$ ) more often than Tropicana.

The answer is that the simpler model in Equation (1.16) led to a *confounding* between advertisement and brand effects. Figure 1.8 shows that Minute Maid was featured more often than Tropicana. Since being featured leads to more price sensitivity, this made Minute Maid artificially appear more price sensitive when you don't account for the ad's effect. The model in Equation (1.18) corrects this by including ad in the regression. This phenomenon, where variable effects can get confounded if you don't *control* for them correctly (i.e., include those effects in your regression model), will play an important role in our later discussions of causal inference.

## 1.1.2 Prediction with `glm`

Once you have decided on a fitted model, using it for prediction is easy in R. The `predict` function takes the model you want to use for prediction and a data frame containing the new data you want to predict with.

**Example 1.4 Orange Juice Sales: Predicting Sales** We can use our fitted model, `fit3way`, to make predictions of sales of orange juice. Suppose you want to predict sales for all three brands when orange juice is featured at a price of \$2.00 per carton. The first step is to create a data frame containing the observations to predict from. Be sure to specify a value for each predictor in the model.

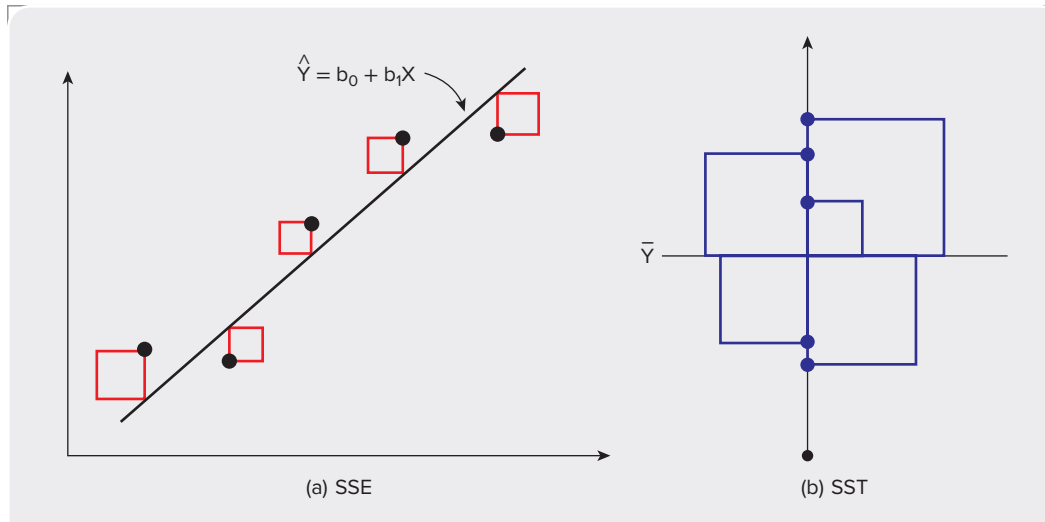
```
> newdata <- data.frame(price=rep(2,3),
+   brand=factor(c("tropicana","minute.maid","dominicks"),
+   levels=levels(oj$brand)),
+   ad=rep(1,3))
> newdata #our data frame of 3 new observations
```

	price	brand	ad
1	2	tropicana	1
2	2	minute.maid	1
3	2	dominicks	1









**FIGURE 1.10** Figure 1.10a shows the squared residual errors. These are the components of the SSE, the quantity that linear regression minimizes and is output in `glm` as `Residual deviance`. Figure 1.10b shows the squared vertical distance for each observation to the overall mean,  $\bar{y}$ . The sum of these squared areas is the sum square total (SST) and is output as `Null deviance`.

This residual deviance for linear regression is known as the *sum squared residual error*, or SSE. It measures the tightness of your model fit. For example, a common metric for model fit takes the SSE and scales it by the number of observations to get mean squared error:  $MSE = SSE/n$ , where  $n$  is the sample size.

### Proportion of Deviance Explained

The calculations behind SSE and SST are illustrated for simple linear regression in Figure 1.10. Comparison between these two deviances tells you how the variability has been *reduced* due to the information in your regression inputs. A common and useful statistic, one that we will use throughout the book, is the  $R^2$  equal to one minus the residual deviance over the null deviance. In linear regression this is

$$R^2 = 1 - \frac{SSE}{SST} \quad (1.20)$$

The  $R^2$  is the *proportion of variability explained by the regression*. It is the proportional reduction in squared errors due to your regression inputs. The name  $R^2$  is derived from the fact that, for linear regression only, it is equal to the square of the correlation (usually denoted  $r$ ) between fitted  $\hat{y}_i$  and observed values  $y_i$ .

**Example 1.5 Orange Juice Sales:  $R^2$**  We can calculate the  $R^2$  a couple of different ways for our three-way interaction OJ regression.

```
# using the glm object attributes
> 1-fit3way$deviance/fit3way$null.deviance
```

```
[1] 0.5353939
# using the SSE and SST calculated above
> 1 - SSE/SST
[1] 0.5353939
# correlation squared
> cor(fit3way$fitted, log(oj$sales))^2
[1] 0.5353939
```

However you calculate it, the regression model explains around 54% of the variability in log orange juice sales. The interpretation of  $R^2$  as squared correlation can help you get a sense of what this means: if  $R^2 = 1$ , a plot of fitted vs. observed values should lie along the perfectly straight line  $\hat{y} = y$ . As  $R^2$  decreases the scatter around this line increases.

The residual, or “fitted,” deviance plays a crucial role in how models are fit. The concept of deviance minimization is crucial to all model estimation and machine learning. In the case of linear regression, you are minimizing the sum of squared residual errors. This gives linear regression its common name: *ordinary least squares*, or OLS (the “ordinary” is in contrast to “weighted least squares” in which some observations are given more weight than others). Our readers coming from an economics or social sciences background might be more familiar with this terminology. We will use the terms OLS and linear regression interchangeably.

## 1.2.2 Degrees of Freedom

Reprinting the relevant summary output, we have one final concept to decipher.

```
> summary(fit3way)
...
(Dispersion parameter for gaussian family taken to be 0.4829706)
    Null deviance: 30079   on 28946   degrees of freedom
Residual deviance: 13975   on 28935   degrees of freedom
...
```

The *degrees of freedom* are crucial for mapping from your deviance to the estimated dispersion parameter,  $\hat{\sigma}^2$ . Unfortunately, the way that `summary.glm` uses this term is confusing because it doesn’t differentiate between two different types of degrees of freedom: those used in the model fit, and those left for calculating the error variance. These concepts are important in statistics and machine learning, so we’ll take the time to pull them apart.

To understand degrees of freedom, take a step back from regression and consider one of the most basic estimation problems in statistics: estimating the variance of a random variable.

Say you have a sample  $\{z_1 \dots z_n\}$  drawn independently from the probability distribution  $p(z)$ . Recall your usual formula for estimating the variance of this distribution:

$$\text{var}(z) \approx \sum_{i=1}^n \frac{(z_i - \bar{z})^2}{n-1} \quad (1.21)$$

where  $\bar{z} = (1/n) \sum_{i=1}^n z_i$  is the sample mean. Why are we dividing by  $(n-1)$  instead of  $n$ ? If we divide by  $n$  our estimate of the variance will be *biased low*—it will tend to underestimate  $\text{var}(z)$ . To get the intuition behind this, consider an  $n=1$  sample that consists of a single draw:  $\bar{z} = z_1$ , and thus  $z_1 - \bar{z} = 0$  by construction. Since you are estimating the mean from your sample, you have the flexibility to fit perfectly a single observation. In other words, when  $n=1$  you have zero opportunities to view any actual variation around the mean. Extending to a larger sample of size  $n$ , you have only  $n-1$  opportunities to observe variation around  $\bar{z}$ .

To use the language of statistics, “opportunities to observe variation” are called degrees of freedom. In our simple variance example, we used one *model degree of freedom* to estimate  $\mathbb{E}[z]$ , and that leaves us with  $n-1$  *residual degrees of freedom* to observe error variance. More generally:

- The **model degrees of freedom** are the number of random observations your model could fit perfectly. In regression models, this is the number of coefficients. For example, given a model with two coefficients (an intercept and slope) you can fit a line directly through two points.
- The **residual degrees of freedom** are equal to the number of opportunities that you have to observe variation around the fitted model mean. This is the sample size,  $n$ , minus the model degrees of freedom.

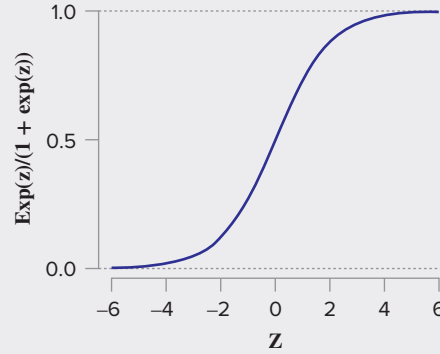
The model degrees of freedom are used for fitting the model and the residual degrees of freedom are what is left over for calculating variability after fitting the model. Throughout the rest of the book, we will follow the convention of using *degrees of freedom* (or *df*) to refer to the model degrees of freedom unless stated otherwise. Somewhat confusingly, the `summary.glm` output uses `degrees of freedom` to refer to the residual degrees of freedom, or  $n - df$  in our notation.

Once we have the terminology straight, we can now complete our original mission to understand how `glm` has calculated  $\hat{\sigma}^2$ . In fitting the linear regression, the number of model degrees of freedom used is equal to the number of parameters in the regression line. For our model in `fit3way`, there are a total of 12 parameters in the model (use `length(coef(fit3way))` to verify). So we would say  $df = 12$  for this model. And since there are 28,947 observations in the OJ dataset, the residual degrees of freedom for this model are  $28,947 - 12 = 28,935$ . This is the number that `glm` outputs next to the residual deviance. It is the number of opportunities that we have to view variation around the regression line. So, to estimate the residual variance, we take the sum of the squared residuals (the SSE) and divide by 28,935.

```
> SSE/fit3way$df.residual
[1] 0.4829706
```

This gives you  $\hat{\sigma}^2$ , or what `summary.glm` calls the “dispersion.” The summary output also provides a degrees of freedom for the null deviance. Since the null model fits only a single mean value,  $\mathbb{E}[y] = \bar{y}$ , this is equal to  $n-1$ , the denominator in our simple variance equation (1.21). For the OJ example this is 28,946.





**FIGURE 1.11** The logit link function,  $f(z) = e^z/(1 + e^z)$ .

Using a logit link, the GLM equation for  $\mathbb{E}[y|\mathbf{x}]$  is defined as

$$\mathbb{E}[y|\mathbf{x}] = p(y = 1|\mathbf{x}) = \frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{1 + e^{\mathbf{x}'\boldsymbol{\beta}}} = \frac{e^{\beta_0 + \beta_1 x_1 + \dots + \beta_k x_k}}{1 + e^{\beta_0 + \beta_1 x_1 + \dots + \beta_k x_k}} \quad (1.22)$$

A common alternate way to write (1.22) results from dividing the numerator and denominator by  $e^{\mathbf{x}'\boldsymbol{\beta}}$ :

$$\mathbb{E}[y|\mathbf{x}] = \frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{1 + e^{\mathbf{x}'\boldsymbol{\beta}}} = \frac{\frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{e^{\mathbf{x}'\boldsymbol{\beta}}}}{\frac{1}{e^{\mathbf{x}'\boldsymbol{\beta}}} + \frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{e^{\mathbf{x}'\boldsymbol{\beta}}}} = \frac{1}{e^{-\mathbf{x}'\boldsymbol{\beta}} + 1} \quad (1.23)$$

How do we interpret the  $\beta$  coefficients in this model? We need to start with the relationship between probability and *odds*. The odds of an event are defined as the probability that it happens over the probability that it doesn't.

$$\text{odds} = \frac{p}{1 - p} \quad (1.24)$$

For example, if an event has a 0.25 probability of happening, then its odds are 0.25/0.75, or 1/3. If an event has a probability of 0.9 of happening, the odds of its happening are 0.9/0.1 = 9. Odds transform from probabilities, which take values between zero and one, to the space of all positive values from zero to infinity.

Looking at (1.22), we can do some algebra and then take the log to derive an interpretation for the  $\beta_j$  coefficients. Using the shorthand  $p = p(y = 1|\mathbf{x})$ , we have

$$\begin{aligned} p &= \frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{1 + e^{\mathbf{x}'\boldsymbol{\beta}}} \\ \Rightarrow p + pe^{\mathbf{x}'\boldsymbol{\beta}} &= e^{\mathbf{x}'\boldsymbol{\beta}} \\ \Rightarrow \frac{p}{1 - p} &= e^{\mathbf{x}'\boldsymbol{\beta}} \\ \Rightarrow \log\left(\frac{p}{1 - p}\right) &= \beta_0 + \beta_1 x_1 + \dots + \beta_k x_k \end{aligned}$$

Thus, logistic regression is a *linear model for log odds*. Using what we know about logs and exponentiation, you can interpret  $e^{\beta_j}$  as the *multiplicative effect* for a unit increase in  $x_j$  on the

odds for the event  $y=1$ . For example, consider a logistic regression model with a single predictor  $x$ , such that  $\text{odds}(x) = \exp[\beta_0 + \beta_1 x]$ .

### 1.3.2 Fitting Logistic Regression in R

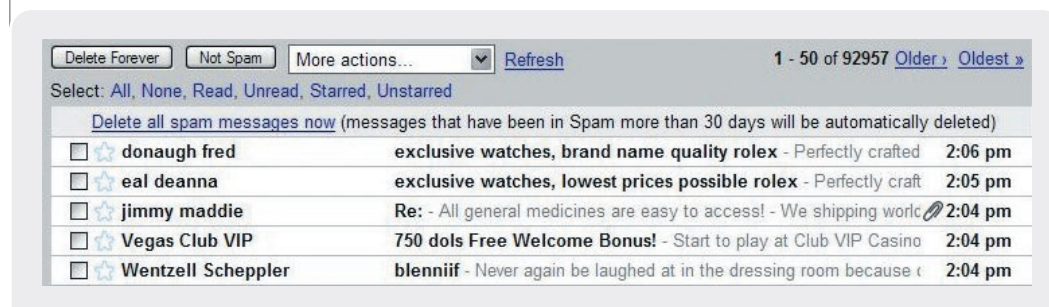
You can use `glm` to fit logistic regressions in R. The syntax is exactly the same as for linear regression, you just add the argument `family="binomial"`. Recall that the binomial distribution is the distribution for random trials with a binary outcome. The classic binomial distribution is a coin toss. Telling `glm` that you are working with a binomial distribution implies that you will be working with a binary response and want to estimate probabilities. The logit link is how `glm` fits probabilities. The response variable can take a number of forms including numeric 0 or 1, logical TRUE or FALSE, or a two level factor such as win vs. lose.

**Example 1.6 Logistic Regression: Detecting Spam** For our first logistic regression example, we'll build a filter for email spam—junk mail that can be ignored. Every time an email arrives, your email client performs a binary classification: is this *spam* or *not spam*? The email that is classified as spam gets automatically moved to a spam folder (like that in Figure 1.12), keeping your inbox free for important messages. We'll train our own spam filter by fitting logistic regression to previous emails.

Our training data `spam.csv` has 4601 emails, 1813 of which are spam. It contains 57 email features including indicators for the presence of 54 keywords or characters (e.g., free or !), counts for capitalized letters (total number and longest continuous block length), and a numeric spam variable for whether each email has been tagged as spam by a human reader (spam is one for true spam, zero for important emails). We read this data into R as a data frame named `spammy`.

```
> spammy <- read.csv("spam.csv")
> spammy[c(1,4000), c(16,56,58)]
```

	word_free	capital_run_length_longest	spam
1	1	61	1
4000	0	26	0



**FIGURE 1.12** An email folder filled with spam.

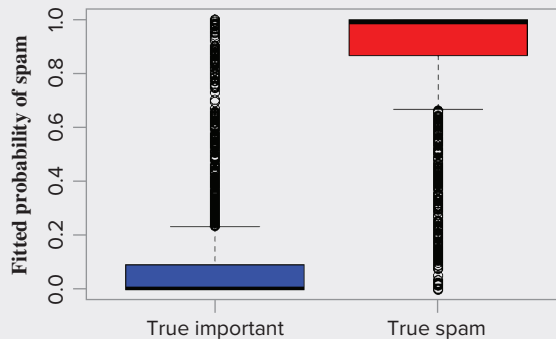
Notice that the first email, which contained the word free and had a block of 61 capitalized letters, was tagged as spam. Email 4000, with its more modest sequence of 26 capital letters, is not spam.

Our logistic regression will use all of the features in `spammy` as inputs. The R formula “`y ~ .`” tells `glm` to regress onto all variables in the data frame except for the response.

```
> spamFit <- glm(spam ~ ., data=spammy, family='binomial')
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
```







**FIGURE 1.13** Fit plot of  $\hat{y}$  versus  $y$  for the spam logistic regression. Since the true  $y$  is binary for spam, you get a boxplot rather than a scatterplot. As a test of your intuition, imagine what a *perfect* fit (i.e.,  $\hat{y} = y$ ) would look like for this regression.

```
> predict(spamFit,newdata=spammy[c(1,4000),],type="response")
      1      4000
0.8839073 0.1509989
```

The first email (true spam) has an 88% chance of being spam, while email 4000 (not spam) has a 15% chance of being spam—in other words, an 85% chance of being important email that George wants to read. Figure 1.13 shows predicted probabilities of spam by actual spam status for every email in the dataset. Note the long tails of small spam probabilities for true spam and of large spam probabilities for truly important mail: any spam classifier that you construct based on this model will occasionally make a mistake on how it treats the email. See Chapter 4 for material on designing and evaluating classification rules.

## 1.4 Likelihood and Deviance

Earlier in this chapter, you learned that deviance is the distance between the *model* and the *data*. In the case of linear regression, the deviance is the sum of squared errors. Logistic regression also has a deviance, and this is the metric that `glm` minimizes to fit the model. But what is the deviance for logistic regression? It is not a sum of squared errors. Instead, the logistic regression deviance is derived from the assumed binomial distribution for the response.

How this works relies on two complementary concepts: the likelihood and the deviance. These concepts are a bit abstract, but they play a key role in the statistical learning algorithms that we will be working with throughout this book.

- *Likelihood* is the probability of your data given the estimated model. When you maximize the likelihood, you are fitting the parameters to “make the data look most likely.”
- *Deviance* is a measure of the distance between the data and the estimated model. When you minimize the deviance, you are fitting the parameters to make the model and data look as close as possible.

## Likelihood

To unravel these concepts we'll start with the likelihood function. Consider a dataset, say  $Z$ , with probability  $p(Z|\Theta)$ . This probability is a function of both the data  $Z$  and the parameters  $\Theta$ . The likelihood function takes a given dataset as fixed and represents how this probability changes as a function of the parameters. Thus we write the likelihood as  $\text{lhd}(\Theta; Z)$ , or sometimes just  $\text{lhd}(\Theta)$  for short, to indicate that it is a function of the parameters  $\Theta$ . But there is nothing complicated going on: the likelihood is just a probability. In particular,  $\text{lhd}(\Theta) = p(Z|\Theta)$  in our imaginary setup here.

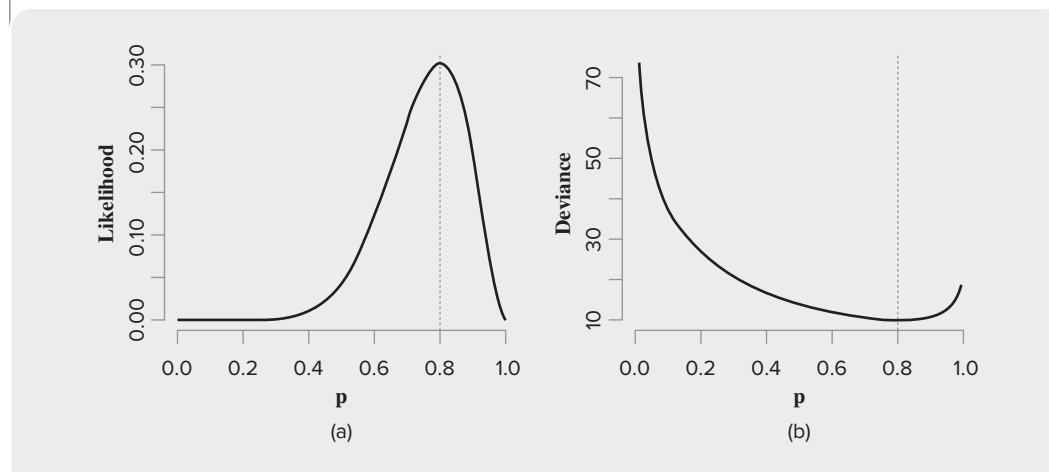
Consider a simple binomial example. You have a weighted coin with probability  $p$  of coming up heads. You have flipped the coin ten times: it has come up heads eight times and tails twice. We could write our dataset as  $Z = \{\text{heads} = 8, \text{tails} = 2\}$ . The probability of this dataset, and the likelihood, is written

$$p(Z|p) = \binom{10}{8} p^8 (1-p)^2 = \text{lhd}(p) \quad (1.25)$$

We can evaluate and plot this likelihood in R (see Figure 1.14a).

```
> p <- seq(0,1,length=100)
> plot(p, dbinom(8, size=10, prob=p), type="l", ylab="Likelihood")
```

Every time `glm` fits a model, it is choosing the parameters to maximize the likelihood. This is a very common estimation strategy with many great properties. Although we will look at other techniques in the next chapter, in particular adding *penalties* on parameter size during estimation, everything will still be built around the foundation of likelihood maximization. In our coin-flipping example, the maximum likelihood estimate (the MLE) is  $\hat{p} = 0.8$ . This is marked with a vertical line in Figure 1.14a, and it corresponds to the highest point on the likelihood curve.



**FIGURE 1.14** The likelihood (a) and deviance (b) for the probability of success,  $p$ , in a binomial trial with eight successes and two failures.

## Deviance

The deviance—the distance between your model and the data—is a simple transformation of the likelihood. In particular,

$$\text{Deviance} = -2 \log[\text{Likelihood}] + C \quad (1.26)$$

Here,  $C$  is a constant that you can ignore. The precise definition for deviance is  $-2$  times the difference between log likelihoods for your fitted model and for a “fully saturated” model where you have as many parameters as observations. The term corresponding to this fully saturated model gets wrapped into the constant,  $C$ , but again you can ignore this in most situations. In practice, we will often use the  $\propto$ , or proportional to, symbol when working with the deviance and only keep track of the parts that change as a function of the parameters. For example, in our coin tossing example, the deviance is

$$\text{dev}(p) \propto -2 \log(p^8(1-p)^2) = -16 \log(p) - 4 \log(1-p) \quad (1.27)$$

This is plotted in Figure 1.14b, with the deviance minimizing solution marked at  $\hat{p} = 0.8$ . Deviance minimization is the mirror image of likelihood maximization. With `glm` you have been fitting models by minimizing the deviance, just the same as you have been fitting models to maximize the likelihood.

**Example 1.7 Gaussian Deviance** Let’s work through an example with linear regression and Gaussian (normal) errors. The probability model is  $y \sim N(\mathbf{x}'\boldsymbol{\beta}, \sigma^2)$ , where the Gaussian probability density function is

$$N(\mathbf{x}'\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(y - \mathbf{x}'\boldsymbol{\beta})^2}{2\sigma^2}\right] \quad (1.28)$$

Recall that independent random variables have the property that  $p(y_1, \dots, y_n) = p(y_1) \times p(y_2) \times \dots \times p(y_n)$ . Given  $n$  independent observations, the likelihood (i.e., the probability density of the data) is

$$\prod_{i=1}^n p(y_i | \mathbf{x}_i) = \prod_{i=1}^n N(y_i; \mathbf{x}_i' \boldsymbol{\beta}, \sigma^2) = (2\pi\sigma^2)^{-\frac{n}{2}} \exp\left[-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mathbf{x}_i' \boldsymbol{\beta})^2\right] \quad (1.29)$$

Taking a log and multiplying by  $-2$  (and removing terms that don’t involve  $\boldsymbol{\beta}$ ), you get

$$\text{dev}(\boldsymbol{\beta}) = \frac{1}{\sigma^2} \sum_{i=1}^n (y_i - \mathbf{x}_i' \boldsymbol{\beta})^2 + C \propto \sum_{i=1}^n (y_i - \mathbf{x}_i' \boldsymbol{\beta})^2 \quad (1.30)$$

Thus, for linear regression with Gaussian errors, the deviance is proportional to the sum of squared errors (the SSE). We stated this fact earlier in the chapter, but now you can derive it for yourself. This is why linear regression is also “least-squares” regression: deviance minimization is the same thing as minimizing the SSE.

**Example 1.8 Logistic Deviance** We can do a similar derivation for logistic regression. For binary response with probabilities  $p_i = p(y_i = 1)$ , the likelihood is

$$\prod_{i=1}^n P(y_i | \mathbf{x}_i) = \prod_{i=1}^n p_i^{y_i} (1 - p_i)^{1-y_i} \quad (1.31)$$

Using your logistic regression equation for  $p_i$ , this becomes

$$\text{lhs}(\boldsymbol{\beta}) = \prod_{i=1}^n \left( \frac{\exp(\mathbf{x}_i' \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})} \right)^{y_i} \left( \frac{1}{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})} \right)^{1-y_i} \quad (1.32)$$

Taking log and multiplying by -2 gives you the logistic regression deviance:

$$\begin{aligned} \text{dev}(\boldsymbol{\beta}) &= -2 \sum_{i=1}^n [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)] \\ &\propto \sum_{i=1}^n [\log(1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})) - y_i \mathbf{x}_i' \boldsymbol{\beta}] \end{aligned} \quad (1.33)$$

This is the function that `glm` minimizes for logistic regression.

### Deviance in summary.glm

Returning to our output from `summary.glm` (which is the function that is called when you apply `summary` to a fitted `glm` object), we have deviances for each of the OJ and spam regressions. For the three-way interaction OJ linear regression:

```
> summary(fit3way)
...
Null deviance: 30079 on 28946 degrees of freedom
Residual deviance: 13975 on 28935 degrees of freedom
...
```

And for the spam filter logistic regression:

```
> summary(spamFit)
...
Null deviance: 6170.2 on 4600 degrees of freedom
Residual deviance: 1548.7 on 4543 degrees of freedom
...
```

From Equations (1.30) and (1.33), we now know how to calculate these residual deviance values. The null deviances come from the same models, and so have the same functional form, but they replace the regression fitted values for  $y$  with simple sample averages. With  $D_0$  as the symbol for null deviance, we have

- $D_0 = \sum (y_i - \bar{y})^2$  in linear regression
- $D_0 = -2 \sum [y_i \log(\bar{y}) + (1 - y_i) \log(1 - \bar{y})]$  in logistic regression

While some statistics texts restrict the concept of  $R^2$  to linear regression, we find it useful to generalize it as *the proportion of deviance that is reduced due to the regression model*. Using the symbol  $D$  to denote the residual deviance, our  $R^2$  formula is

$$R^2 = 1 - \frac{D}{D_0} \quad (1.34)$$

This  $R^2$  formula is often called McFadden's Pseudo  $R^2$  when it is used outside of linear regression.

In the case of our OJ example, we previously calculated an  $R^2$  of 0.54. For the spam regression, we can apply (1.34) to calculate that  $R^2 = 1 - 1549/6170 = 0.75$  such that around three-quarters of the variability in spam occurrence is explained by our logistic regression. We introduced  $R^2$  in the context of linear regression, as a function of the SST and SSE, but expressing it in terms of deviance means that it applies to any model that we fit.

## ■ 1.5 Time Series

We close this chapter with an introduction to working with *dependent* data. The models we have looked at so far all assume that you have *independent* observations. However, events that occur one after the other in time, or say geographically near to each other, can be correlated. For example, lawn furniture sales are always higher in spring and summer, the weather today gives you information about what the weather will be tomorrow, or when a popular restaurant has a busy night its neighboring restaurants also gain traffic from those who couldn't get a table. In this section we'll figure out how to work with data that occurs in *time*, and in the next section we consider data that occurs in *space*.

Fortunately, the main tools for dealing with dependence all fit within a standard regression framework. For the most part, you simply include the variables that cause dependence in your set of inputs. By engineering the right input features, you can control for underlying trends (e.g., monthly trends or regional effects) and for *autoregression*, which is the dependence between neighboring outcomes. In this section we will focus on *time series* dependence. This is the sort of dependence that you get for data that are observed over time, and it is common in business analysis settings. The tools you learn for time series extend to other dependence settings, and we give some pointers on this at the end of the section.

The traditional statistics approach to time series emphasizes careful testing for different forms of time series structure. Through the regularization and machine learning material from later chapters in this book, we can avoid a lot of this manual feature selection. Although it won't work for all types of time series dependence, a powerful modeling strategy is to simply include a large set of time series features and rely on the data to tell you what works best. Thus, this section will focus on helping you understand how to construct the features that are useful for modeling time series data rather than on techniques for testing for time series dependence. If you have a good intuition about the ingredients of a time series model, you will be in good shape to use these features in your applied analysis work.

### 1.5.1 Regression for Time Series Data

A time series dataset contains observations of a response variable, and input features, taken over time. Typical time intervals are daily, weekly, monthly, quarterly, or yearly. In business settings, the response of interest is typically sales numbers, revenue, profit, active users, or prices. The response variables are almost always correlated over time.

**Example 1.9 Airline Passenger Data: Regression for Time Series** As an introductory example, consider a series of monthly total international airline passengers between the years 1949 and 1960.

```
> air<-read.csv("airline.csv")
> air[c(1,70,144),]
      Year Month Passengers
1      49      1         112
70     54     10         229
144    60     12         432
```

To work with time series data in R, your first step is to create a time variable. We can use the `as.Date` function to build a Date class vector. Note that if you are working on a finer time scale, R has the `POSIXct` class that can be used to represent dates and times down to fractions of a second. To create a Date variable in our air travel example, we need to translate from the Year and Month variables in our data frame. The first step is to paste these two variables into a single year-month string for each observation, and we then call `as.Date` to tell R that this is date information. The default format to read in dates is year-month-day, and that is what we will use here. We set the day to the first of each month for convenience; these are monthly counts so it doesn't matter what day we use.

```
> air$date<-paste("19",air$Year,"-",air$Month,"-01", sep="")
> air$date[c(1,70,144)]
[1] "1949-1-01" "1954-10-01" "1960-12-01"
> air$date<-as.Date(air$date)
> air$date[c(1,70,144)]
[1] "1949-01-01" "1954-10-01" "1960-12-01"
> class(air$date)
[1] "Date"
```

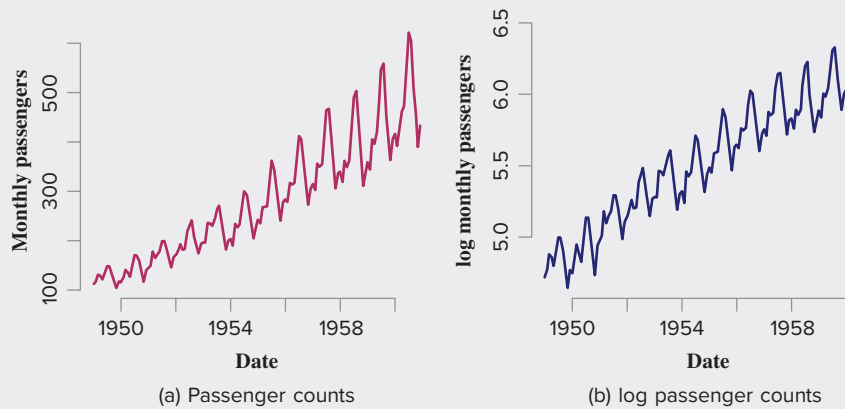
We now have the date variable, which R knows to treat as a calendar date. These dates are represented internally as days relative to January 1, 1970. You can convert them to numeric to see how R tracks the date.

```
> as.numeric(air$date[c(1,70,144)])
[1] -7670 -5571 -3318
```

```
> as.numeric(as.Date("1970-01-01"))
[1] 0
```

The two lines of code below produce plots of this data as in Figure 1.15.

```
> plot(Passengers ~ date, data=air, type="l")
> plot(log(Passengers) ~ date, data=air, type="l")
```



**FIGURE 1.15** Time series data for 12 years of the monthly total count of international air passengers, 1949 through 1960.

Unlike our usual approach of plotting scatters of data points, here we have drawn a line plot (using the `type="l"` argument) to indicate dependence over time. In Figure 1.15a, you see an overall trend of an increasing number of passengers with time. In addition to this upward trend, you also see a repeated annual oscillation around the annual average. It is evident from Figure 1.15a that the oscillations around this upward trend are getting larger with time. This is a hallmark of a time series that is changing on a percentage scale with each observation. Recalling our work with sales data earlier in this chapter, that is an indication that you will want to be building a linear model on the log scale. Figure 1.15b shows the log monthly passenger volume. You can see that the log transformation yields consistently sized annual oscillations around a roughly linear-looking trend.

### Linear Time Trend

We could fit a simple linear regression model to this data, say

$$\log(y_t) = \alpha + \beta t + \varepsilon_t$$

If you use the date variable as the input to `glm` to fit this regression, then from our `as.numeric` representations above you can see that the time trend will be counted in terms of days. This means that the impact of  $\beta$  on the monthly change will be a function of the number of the days of the month. This might be desirable in some applications, but to keep things simple here we will instead regress onto a simple index variable  $t$  that tracks the counts of months since the beginning of the dataset.

```
> air$t <- 1:nrow(air)
> fitAirSLR <- glm(log(Passengers)~t, data=air)
> coef(fitAirSLR)
```