



Sports Analytics and Data Science

Winning the Game with Methods and Models

THOMAS W. MILLER

Faculty Director of Northwestern University's Predictive Analytics Program

About This E-Book

EPUB is an open, industry-standard format for e-books. However, support for EPUB and its many features varies across reading devices and applications. Use your device or app settings to customize the presentation to your liking. Settings that you can customize often include font, font size, single or double column, landscape or portrait mode, and figures that you can click or tap to enlarge. For additional information about the settings and features on your reading device or app, visit the device manufacturer's Web site.

Many titles include programming code or configuration examples. To optimize the presentation of these elements, view the e-book in single-column, landscape mode and adjust the font size to the smallest setting. In addition to presenting code and configurations in the reflowable text format, we have included images of the code that mimic the presentation found in the print book; therefore, where the reflowable format may compromise the presentation of the code listing, you will see a “Click here to view code image” link. Click the link to view the print-fidelity code image. To return to the previous page viewed, click the Back button on your device or app.

Sports Analytics and Data Science

Winning the Game with Methods and Models

THOMAS W. MILLER

Publisher: Paul Boger
Editor-in-Chief: Amy Neidlinger
Executive Editor: Jeanne Glasser Levine
Cover Designer: Alan Clements
Managing Editor: Kristy Hart
Project Editor: Andy Beaster
Manufacturing Buyer: Dan Uhrig

©2016 by Thomas W. Miller
Published by Pearson Education, Inc.
Old Tappan, New Jersey 07675

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact international@pearsoned.com.

Company and product names mentioned herein are the trademarks or registered trademarks of their respective owners.

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

Printed in the United States of America

First Printing November 2015

ISBN-10: 0-13-388643-3

ISBN-13: 978-0-13-388643-6

Pearson Education LTD.

Pearson Education Australia PTY, Limited.

Pearson Education Singapore, Pte. Ltd.

Pearson Education Asia, Ltd.

Pearson Education Canada, Ltd.

Pearson Educación de Mexico, S.A. de C.V.

Pearson Education—Japan

Pearson Education Malaysia, Pte. Ltd.

Library of Congress Control Number: 2015954509

Contents

Preface

Figures

Tables

Exhibits

1 Understanding Sports Markets

2 Assessing Players

3 Ranking Teams

4 Predicting Scores

5 Making Game-Day Decisions

6 Crafting a Message

7 Promoting Brands and Products

8 Growing Revenues

9 Managing Finances

10 Playing What-if Games

11 Working with Sports Data

12 Competing on Analytics

A Data Science Methods

A.1 Mathematical Programming

A.2 Classical and Bayesian Statistics

A.3 Regression and Classification

A.4 Data Mining and Machine Learning

A.5 Text and Sentiment Analysis

A.6 Time Series, Sales Forecasting, and Market Response Models

A.7 Social Network Analysis

A.8 Data Visualization

A.9 Data Science: The Eclectic Discipline

B Professional Leagues and Teams

Data Science Glossary

Baseball Glossary

Bibliography

Index

Preface

“Sometimes you win, sometimes you lose, sometimes it rains.”

—TIM ROBBINS AS EBBY CALVIN LALOOSH IN *Bull Durham*
(1988)

Businesses attract customers, politicians persuade voters, websites cajole visitors, and sports teams draw fans. Whatever the goal or target, data and models rule the day.

This book is about building winning teams and successful sports businesses. Winning and success are more likely when decisions are guided by data and models. Sports analytics is a source of competitive advantage.

This book provides an accessible guide to sports analytics. It is written for anyone who needs to know about sports analytics, including players, managers, owners, and fans. It is also a resource for analysts, data scientists, and programmers. The book views sports analytics in the context of data science, a discipline that blends business savvy, information technology, and modeling techniques.

To use analytics effectively in sports, we must first understand sports—the industry, the business, and what happens on the fields and courts of play. We need to know how to work with data—identifying data sources, gathering data, organizing and preparing them for analysis. We also need to know how to build models from data. Data do not speak for themselves. Useful predictions do not arise out of thin air. It is our job to learn from data and build models that work.

The best way to learn about sports analytics and data science is through examples. We provide a ready resource and reference guide for modeling techniques. We show programmers how to solve real world problems by building on a foundation of trustworthy methods and code.

The truth about what we do is in the programs we write. The code is there for everyone to see and for some to debug. Data sets and computer programs are available from the website for the *Modeling Techniques* series at

<http://www.ftpress.com/miller/>. There is also a GitHub site at <https://github.com/mtpa/>.

When working on sports problems, some things are more easily accomplished with R, others with Python. And there are times when it is good to offer solutions in both languages, checking one against the other.

One of the things that distinguishes this book from others in the area of sports analytics is the range of data sources and topics discussed. Many researchers focus on numerical performance data for teams and players. We take a broader view of sports analytics—the view of data science. There are text data as well as numeric data. And with the growth of the World Wide Web, the sources of data are plentiful. Much can be learned from public domain sources through crawling and scraping the web and utilizing application programming interfaces (APIs).

I learn from my consulting work with professional sports organizations. Research Publishers LLC with its ToutBay division promotes what can be called “data science as a service.” Academic research and models can take us only so far. Eventually, to make a difference, we need to implement

our ideas and models, sharing them with one another.

Many have influenced my intellectual development over the years. There were those good thinkers and good people, teachers and mentors for whom I will be forever grateful.

Sadly, no longer with us are Gerald Hahn Hinkle in philosophy and Allan Lake Rice in languages at Ursinus College, and Herbert Feigl in philosophy at the University of Minnesota. I am also most thankful to David J. Weiss in psychometrics at the University of Minnesota and Kelly Eakin in economics, formerly at the University of Oregon.

My academic home is the Northwestern University School of Professional Studies. Courses in sports research methods and quantitative analysis, marketing analytics, database systems and data preparation, web and network data science, web information retrieval and real-time analytics, and data visualization provide inspiration for this book. Thanks to the many students and fellow faculty from whom I have learned. And thanks to colleagues and staff who administer excellent graduate programs, including the Master of Science in Predictive Analytics, Master of Arts in Sports Administration, Master of Science in Information Systems, and the Advanced Certificate in Data Science.

Lorena Martin reviewed this book and provided valuable feedback while she authored a companion volume on sports performance measurement and analytics ([Martin 2016](#)). Adam Grossman and Tom Robinson provided valuable feedback about coverage of topics in sports business management. Roy Sanford provided advice on statistics. Amy Hendrickson of TEXnology Inc. applied her craft, making words, tables, and figures look beautiful in print—another victory for open

source. Candice Bradley served dual roles as a reviewer and copyeditor for all books in the *Modeling Techniques* series. And Andy Beaster helped in preparing this book for final production. I am grateful for their guidance and encouragement.

Thanks go to my editor, Jeanne Glasser Levine, and publisher, Pearson/FT Press, for making this book possible. Any writing issues, errors, or items of unfinished business, of course, are my responsibility alone.

My good friend Brittney and her daughter Janiya keep me company when time permits. And my son Daniel is there for me in good times and bad, a friend for life. My greatest debt is to them because they believe in me.

Thomas W. Miller
Glendale, California
October 2015

Figures

1.1 MLB, NBA, and NFL Average Annual Salaries

1.2 MLB Team Payrolls and Win/Loss Performance (2014 Season)

1.3 A Perceptual Map of Seven Sports

2.1 Multitrait-Multimethod Matrix for Baseball Measures

3.1 Assessing Team Strength: NBA Regular Season (2014–2015)

4.1 Work of Data Science

4.2 Data and Models for Research

4.3 Training-and-Test Regimen for Model Evaluation

4.4 Training-and-Test Using Multi-fold Cross-validation

4.5 Training-and-Test with Bootstrap Resampling

4.6 Predictive Modeling Framework for Team Sports

6.1 How Sports Fit into the Entertainment Space (Or Not)

6.2 Indices of Dissimilarity Between Pairs of Binary Variables

6.3 Consumer Preferences for Dodger Stadium Seating

6.4 Choice Item for Assessing Willingness to Pay for Tickets

6.5 The Market: A Meeting Place for Buyers and Sellers

7.1 Dodgers Attendance by Day of Week

7.2 Dodgers Attendance by Month

7.3 Dodgers Weather, Fireworks, and Attendance

7.4 Dodgers Attendance by Visiting Team

7.5 Regression Model Performance: Bobbleheads and Attendance

8.1 Competitive Analysis for an NBA Team: Golden State Warriors

9.1 Cost-Volume-Profit Analysis

9.2 Higher Profits Through Increased Sales

9.3 Higher Profits Through Lower Fixed Costs

9.4 Higher Profits Through Increased Efficiency

9.5 Decision Analysis: Investing in a Sports Franchise (Or Not)

10.1 Game-day Simulation (Offense Only)

10.2 Mets' Away and Yankees' Home Data (Offense and Defense)

10.3 Balanced Game-day Simulation (Offense and Defense)

10.4 Actual and Theoretical Runs-scored Distributions

10.5 Poisson Model for Mets vs. Yankees at Yankee Stadium

10.6 Negative Binomial Model for Mets vs. Yankees at Yankee Stadium

10.7 Probability of Home Team Winning (Negative Binomial Model)

10.8 Strategic Modeling Techniques in Sports

11.1 Software Stack for a Document Search and Selection System

11.2 The Information Supply Chain of Professional Team Sports

11.3 Automated Data Acquisition by Crawling, Scraping, and Parsing

11.4 Automated Data Acquisition with an API

11.5 Gathering and Organizing Data for Analysis

A.1 Mathematical Programming Modeling Methods

A.2 Evaluating the Predictive Accuracy of a Binary Classifier

A.3 Linguistic Foundations of Text Analytics

A.4 Creating a Terms-by-Documents Matrix

A.5 Data and Plots for the Anscombe Quartet

A.6 Visualizing Many Games Across a Season: Differential Runs Plot

A.7 Moving Fraction Plot for Basketball

A.8 Visualizing Basketball Play-by-Play Data

A.9 Data Science: The Eclectic Discipline

Tables

- 1.1 Sports and Recreation Activities in the United States
- 1.2 MLB Team Valuation and Finances (March 2015)
- 1.3 NBA Team Valuation and Finances (January 2015)
- 1.4 NFL Team Valuation and Finances (August 2014)
- 1.5 World Soccer Team Valuation and Finances (May 2015)
- 2.1 Levels of Measurement
- 3.1 NBA Team Records (2014–2015 Season)
- 5.1 Twenty-five States of a Baseball Half-Inning
- 6.1 Dissimilarity Matrix for Entertainment Events and Activities
- 6.2 Consumer Preference Data for Dodger Stadium Seating
- 7.1 Bobbleheads and Dodger Dogs
- 7.2 Regression of Attendance on Month, Day of Week, and Promotion
- 9.1 Discounted Cash Flow Analysis of a Player Contract
- 9.2 Would you like to buy the Brooklyn Nets?
- 10.1 New York Mets' Early Season Games in 2007
- 10.2 New York Yankees' Early Season Games in 2007
- A.1 Three Generalized Linear Models
- A.2 Social Network Data: MLB Player Transactions
- B.1 Women's National Basketball Association (WNBA)
- B.2 Major League Baseball (MLB)

B.3 Major League Soccer (MLS)

B.4 National Basketball Association (NBA)

B.5 National Football League (NFL)

B.6 National Hockey League (NHL)

Exhibits

[1.1 MLB, NBA, and NFL Player Salaries \(R\)](#)

[1.2 Payroll and Performance in Major League Baseball \(R\)](#)

[1.3 Making a Perceptual Map of Sports \(R\)](#)

[3.1 Assessing Team Strength by Unidimensional Scaling \(R\)](#)

[6.1 Mapping Entertainment Events and Activities \(R\)](#)

[6.2 Mapping Entertainment Events and Activities \(Python\)](#)

[6.3 Preferences for Sporting Events—Conjoint Analysis \(R\)](#)

[6.4 Preferences for Sporting Events—Conjoint Analysis \(Python\)](#)

[7.1 Shaking Our Bobbleheads Yes and No \(R\)](#)

[7.2 Shaking Our Bobbleheads Yes and No \(Python\)](#)

[10.1 Team Winning Probabilities by Simulation \(R\)](#)

[10.2 Team Winning Probabilities by Simulation \(Python\)](#)

[11.1 Simple One-Site Web Crawler and Scraper \(Python\)](#)

[11.2 Gathering Opinion Data from Twitter: Football Injuries \(Python\)](#)

[A.1 Programming the Anscombe Quartet \(Python\)](#)

[A.2 Programming the Anscombe Quartet \(R\)](#)

[A.3 Making Differential Runs Plots for Baseball \(R\)](#)

[A.4 Moving Fraction Plot: A Basketball Example \(R\)](#)

[A.5 Visualizing Basketball Games \(R\)](#)

[A.6 Seeing Data Science as an Eclectic Discipline \(R\)](#)

1. Understanding Sports Markets

“Those of you on the floor at the end of the game, I’m proud of you. You played your guts out. I’m only going to say this one time. All of you have the weekend. Think about whether or not you want to be on this team under the following condition: What I say when it comes to this basketball team is the law, absolutely and without discussion.”

—GENE HACKMAN AS COACH NORMAN DALE IN *Hoosiers*
(1986)

In applying the laws of economics to professional sports, we must consider the nature of sports and the motives of owners. Professional sports are different from other forms of business. There are sellers and buyers of sports entertainment. The sellers are the players and teams within the leagues of professional sports. The buyers are consumers of sports, many of whom never go to games in person but who watch sports on television, listen to the radio, and buy sports team paraphernalia.

Sports compete with other forms of entertainment for people’s time and money. And various sports compete with one another, especially when their seasons overlap. Sports teams produce entertainment content that is distributed through the media. Sports teams license their brand names and logos to other organizations, including sports apparel manufacturers. Sports teams are not independent businesses competing with one another. While players and teams compete on the fields and courts of play, they cooperate with one another as

members of leagues. The core product of sports is the sporting contest, a joint product of two or more players or two or more teams.

Fifty-four sports and recreation activities, shown in [table 1.1](#), are tracked by the National Sporting Goods Association (2015), which serves the sporting goods industry. In recent years, participation in baseball, basketball, football, and tennis has declined, while participation in soccer has increased. There has been growth in individual recreational sports, such as skateboarding and snowboarding. Of course, levels of participation in sports are not necessarily an indicator of levels of interest in sports as entertainment.

Aerobic Exercising	Ice/Figure Skating
Archery (Target)	In-Line Roller Skating
Backpack/Wilderness Camping	Kayaking
Baseball	Lacrosse
Basketball	Martial Arts/MMA/Tae Kwon Do
Bicycle Riding	Mountain Biking (Off Road)
Billiards/Pool	Muzzleloading
Boating (Motor/Power)	Paintball Games
Bowling	Running/Jogging
Boxing	Scuba Diving (Open Water)
Camping (Vacation/Overnight)	Skateboarding
Canoeing	Skiing (Alpine)
Cheerleading	Skiing (Cross Country)
Dart Throwing	Snowboarding
Exercise Walking	Soccer
Exercising with Equipment	Softball
Fishing (Fresh Water)	Swimming
Fishing (Salt Water)	Table Tennis/Ping Pong
Football (Flag)	Target Shooting (Airgun)
Football (Tackle)	Target Shooting (Live Ammunition)
Football (Touch)	Tennis
Golf	Volleyball
Gymnastics	Water Skiing
Hiking	Weight Lifting
Hockey (Ice)	Work Out at Club/Gym/Fitness Studio
Hunting with Bow & Arrow	Wrestling
Hunting with Firearms	Yoga

Table 1.1. Sports and Recreation Activities in the United States

Sports businesses produce entertainment products by cooperating with one another. While it is illegal for businesses

in most industries to collude in setting output and prices, sports leagues engage in cooperative output and pricing as a standard part of their business model. The number of games, indeed the entire schedule of games in a sport, is determined by the league. In fact, aspects of professional sports are granted monopoly power by the federal government in the United States.

When developing a model for a typical business or firm, an economist would assume profit maximization as a motive. But for a professional sports team, an owner's motives may not be so easily understood. While one owner may operate his or her team for profit year by year, another may seek to maximize wins or overall utility. Another may look for capital appreciation—buying, then selling after a few years. Lacking knowledge of owners' motives, it is difficult to predict what they will do.

Gaining market share and becoming the dominant player is a goal of firms in many industries. Not so in the business of professional sports. If one team were assured of victory in almost all of its contests, interest in those contests could wane. A team benefits by winning more often than losing, but winning all the time may be less beneficial than winning most of the time. Professional sports leagues claim to be seeking competitive balance, although there are dominant teams in many leagues.

Sports is big business as shown by valuations and finances of the major professional sports in the United States and worldwide. Data from *Forbes* for Major League Baseball ([MLB](#)), the National Basketball Association (NBA), the National Football League (NFL), and worldwide soccer teams

are shown in tables 1.2 through 1.5.

Team Rank	Team	One-Year				Operating Income (\$ Millions)
		Current Value (\$ Millions)	Change in Value (Percentage)	Debt/Value (Percentage)	Revenue (\$ Millions)	
1	New York Yankees	3,200	28	0	508	8.1
2	Los Angeles Dodgers	2,400	20	17	403	-12.2
3	Boston Red Sox	2,100	40	0	370	49.2
4	San Francisco Giants	2,000	100	4	387	68.4
5	Chicago Cubs	1,800	50	24	302	73.3
6	St Louis Cardinals	1,400	71	21	294	73.6
7	New York Mets	1,350	69	26	263	25.0
8	Los Angeles Angels	1,300	68	0	304	16.7
9	Washington Nationals	1,280	83	27	287	41.4
10	Philadelphia Phillies	1,250	28	8	265	-39.0
11	Texas Rangers	1,220	48	13	266	3.5
12	Atlanta Braves	1,150	58	0	267	33.2
13	Detroit Tigers	1,125	65	15	254	-20.7
14	Seattle Mariners	1,100	55	0	250	26.4
15	Baltimore Orioles	1,000	61	15	245	31.4
16	Chicago White Sox	975	40	5	227	31.9

17	Pittsburgh Pirates	900	57	10	229	43.6
18	Minnesota Twins	895	48	25	223	21.3
19	San Diego Padres	890	45	22	224	35.0
20	Cincinnati Reds	885	48	6	227	2.2
21	Milwaukee Brewers	875	55	6	226	11.3
22	Toronto Blue Jays	870	43	0	227	-17.9
23	Colorado Rockies	855	49	7	214	12.6
24	Arizona Diamondbacks	840	44	17	211	-2.2
25	Cleveland Indians	825	45	9	207	8.9
26	Houston Astros	800	51	34	175	21.6
27	Oakland Athletics	725	46	8	202	20.8
28	Kansas City Royals	700	43	8	231	26.6
29	Miami Marlins	650	30	34	188	15.4
30	Tampa Bay Rays	625	29	22	188	7.9

Source. Badenhausen, Ozanian, and Settimi ([2015b](#)).

Table 1.2. *MLB Team Valuation and Finances (March 2015)*

Team Rank	Team	Current Value (\$ Millions)	One-Year Change		Operating	
			In Value (Percentage)	Debt/Value (Percentage)	Revenue (\$ Millions)	Income (\$ Millions)
1	Los Angeles Lakers	2,600	93	2	293	104.1
2	New York Knicks	2,500	79	0	278	53.4
3	Chicago Bulls	2,000	100	3	201	65.3
4	Boston Celtics	1,700	94	9	173	54.9
5	Los Angeles Clippers	1,600	178	0	146	20.1
6	Brooklyn Nets	1,500	92	19	212	99.4
7	Golden State Warriors	1,300	73	12	168	44.9
8	Houston Rockets	1,250	61	8	175	38.0
9	Miami Heat	1,175	53	8	188	12.6
10	Dallas Mavericks	1,150	50	17	168	30.4
11	San Antonio Spurs	1,000	52	8	172	40.9
12	Portland Trail Blazers	940	60	11	153	11.7
13	Oklahoma City Thunder	930	58	15	152	30.8
14	Toronto Raptors	920	77	16	151	17.9
15	Cleveland Cavaliers	915	78	22	149	20.6
16	Phoenix Suns	910	61	20	145	28.2
17	Washington Wizards	900	86	14	143	10.1
18	Orlando Magic	875	56	17	143	20.9
19	Denver Nuggets	855	73	1	136	14.0
20	Utah Jazz	850	62	6	142	32.7
21	Indiana Pacers	830	75	18	139	25.0
22	Atlanta Hawks	825	94	21	133	14.8
23	Detroit Pistons	810	80	23	144	17.6
24	Sacramento Kings	800	45	29	125	8.9
25	Memphis Grizzlies	750	66	23	135	10.5
26	Charlotte Hornets	725	77	21	130	1.2
27	Philadelphia 76ers	700	49	21	125	24.4
28	New Orleans Pelicans	650	55	19	131	19.0
29	Minnesota Timberwolves	625	45	16	128	6.9
30	Milwaukee Bucks	600	48	29	110	11.5

Source. Badenhausen, Ozanian, and Settimi (2015a).

Table 1.3. NBA Team Valuation and Finances (January 2015)

Team Rank	Team	One-Year				Operating Income (\$ Millions)
		Current Value (\$ Millions)	Change in Value (Percentage)	Debt/Value (Percentage)	Revenue (\$ Millions)	
1	Dallas Cowboys	3,200	39	6	560	245.7
2	New England Patriots	2,600	44	9	428	147.2
3	Washington Redskins	2,400	41	10	395	143.4
4	New York Giants	2,100	35	25	353	87.3
5	Houston Texans	1,850	28	11	339	102.8
6	New York Jets	1,800	30	33	333	79.5
7	Philadelphia Eagles	1,750	33	11	330	73.2
8	Chicago Bears	1,700	36	6	309	57.1
9	San Francisco 49ers	1,600	31	53	270	24.8
10	Baltimore Ravens	1,500	22	18	304	56.7
11	Denver Broncos	1,450	25	8	301	30.7
12	Indianapolis Colts	1,400	17	4	285	60.7
13	Green Bay Packers	1,375	16	1	299	25.6
14	Pittsburgh Steelers	1,350	21	15	287	52.4
15	Seattle Seahawks	1,330	23	9	288	27.3
16	Miami Dolphins	1,300	21	29	281	8.0

17	Carolina Panthers	1,250	18	5	283	55.6
18	Tampa Bay Buccaneers	1,225	15	15	275	46.4
19	Tennessee Titans	1,160	10	11	278	35.6
20	Minnesota Vikings	1,150	14	43	250	5.3
21	Atlanta Falcons	1,125	21	27	264	13.1
22	Cleveland Browns	1,120	11	18	276	35.0
23	New Orleans Saints	1,110	11	7	278	50.1
24	Kansas City Chiefs	1,100	9	6	260	10.0
25	Arizona Cardinals	1,000	4	15	266	42.8
26	San Diego Chargers	995	5	10	262	39.9
27	Cincinnati Bengals	990	7	10	258	11.9
28	Oakland Raiders	970	18	21	244	42.8
29	Jacksonville Jaguars	965	15	21	263	56.9
30	Detroit Lions	960	7	29	254	-15.9
31	Buffalo Bills	935	7	13	252	38.0
32	St Louis Rams	930	6	12	250	16.2

Source. Badenhausen, Ozanian, and Settimi (2014).

Table 1.4. *NFL Team Valuation and Finances (August 2014)*

Team Rank	Team	One-Year				Operating Income (\$ Millions)
		Current Value (\$ Millions)	Change in Value (Percentage)	Debt/Value (Percentage)	Revenue (\$ Millions)	
1	Real Madrid	3,263	-5	4	746	170
2	Barcelona	3,163	-1	3	657	174
3	Manchester United	3,104	10	20	703	211
4	Bayern Munich	2,347	27	0	661	78
5	Manchester City	1,375	59	0	562	122
6	Chelsea	1,370	58	0	526	83
7	Arsenal	1,307	-2	30	487	101
8	Liverpool	982	42	10	415	86
9	Juventus	837	-2	9	379	50
10	AC Milan	775	-10	44	339	54
11	Borussia Dortmund	700	17	6	355	55
12	Paris Saint-Germain	634	53	0	643	-1
13	Tottenham Hotspur	600	17	9	293	63
14	Schalke 04	572	-1	0	290	57
15	Inter Milan	439	-9	56	222	-41
16	Atletico de Madrid	436	33	53	231	47
17	Napoli	353	19	0	224	43
18	Newcastle United	349	33	0	210	44
19	West Ham United	309	33	12	186	54
20	Galatasaray	294	-15	17	220	-37

Source. Ozanian (2015).

**Table 1.5. World Soccer Team Valuation and Finances
(May 2015)**

Professional sports teams most certainly compete with one another in the labor market, and labor in the form of star players is in short supply. Some argue that salary caps are necessary to preserve competitive balance. Salary caps also help teams in limiting expenditures on players.

Most professional sports in the United States have salary caps.

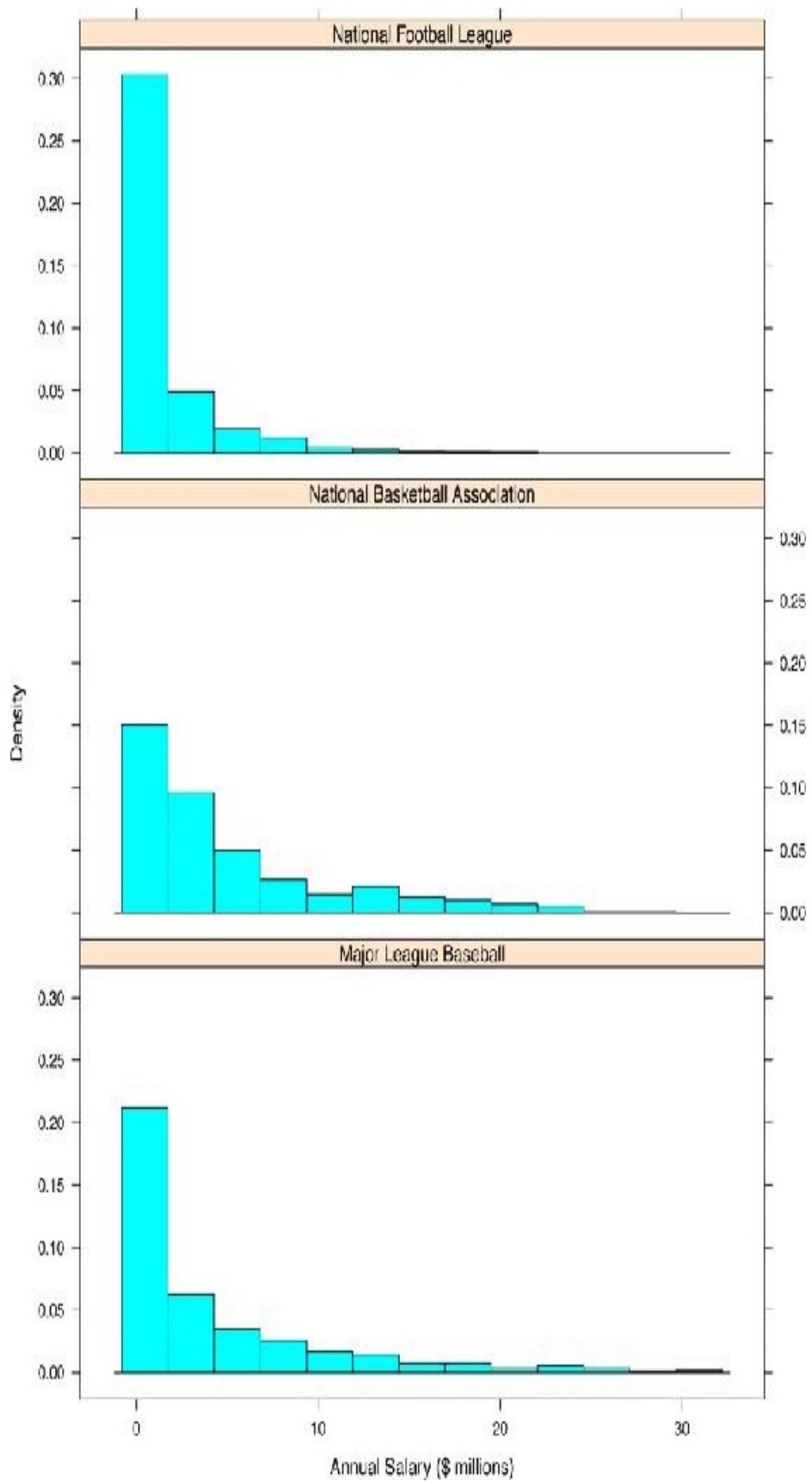
The 2015 salary cap for NFL teams, with fifty-three player rosters, is set at \$143.28 million ([Patra 2015](#)). Most teams have payrolls at or near the cap, making the average salary of an NFL player about \$2.7 million. One player on an NFL team may be designated as a *franchise player*, restricting that player from entering free agency. The league sets minimum salaries for franchise players. For example, a franchise quarterback has a minimum salary of \$18.544 million in 2015. The highest annual salary among NFL players is \$22 million for Aaron Rodgers, Green Bay Packers quarterback ([spotrac 2015c](#)). The minimum annual salary is \$420 thousand.

NBA teams have a \$70 million salary cap for the 2015–16 season, with penalties for teams going over the cap. Maximum player salaries are based on a percentage of cap and years of service. For example, LeBron James, with ten years of experience, would have a maximum salary of \$23 million ([Mahoney 2015](#)). New Orleans Pelicans Anthony Davis' average salary of \$29 million is the highest among NBA players ([spotrac 2015b](#)). Team rosters include fifteen players under contract, with as many as thirteen available to play in any particular game. The minimum annual salary is \$428,498.

Major League Baseball (MLB) has a “luxury tax” for teams with payrolls in excess of \$189 million. There is a regular-player roster of twenty-five or twenty-six players for double-header days/nights. A forty-man roster includes players under contract and eligible to play. Between September 1 and the end of the regular season the roster is expanded to forty players. The roster drops back to twenty-five players for the playoffs. The minimum MLB annual salary is \$505,700 in 2015. The highest MLB annual salary is \$31 million for

Miguel Cabrera of the Detroit Tigers ([spotrac 2015a](#)).

Figure 1.1, a histogram lattice, shows how player salaries compare across the MLB, NBA, and NFL in August 2015. Player salary distributions are positively skewed. The mean salary across NFL players is around \$1.7 million, but the median is \$630 thousand. The mean salary across NBA players is around \$5.1 million, with median salary \$2.8 million. The mean salary across MLB players is around \$4.1 million, with the median \$1.1 million.

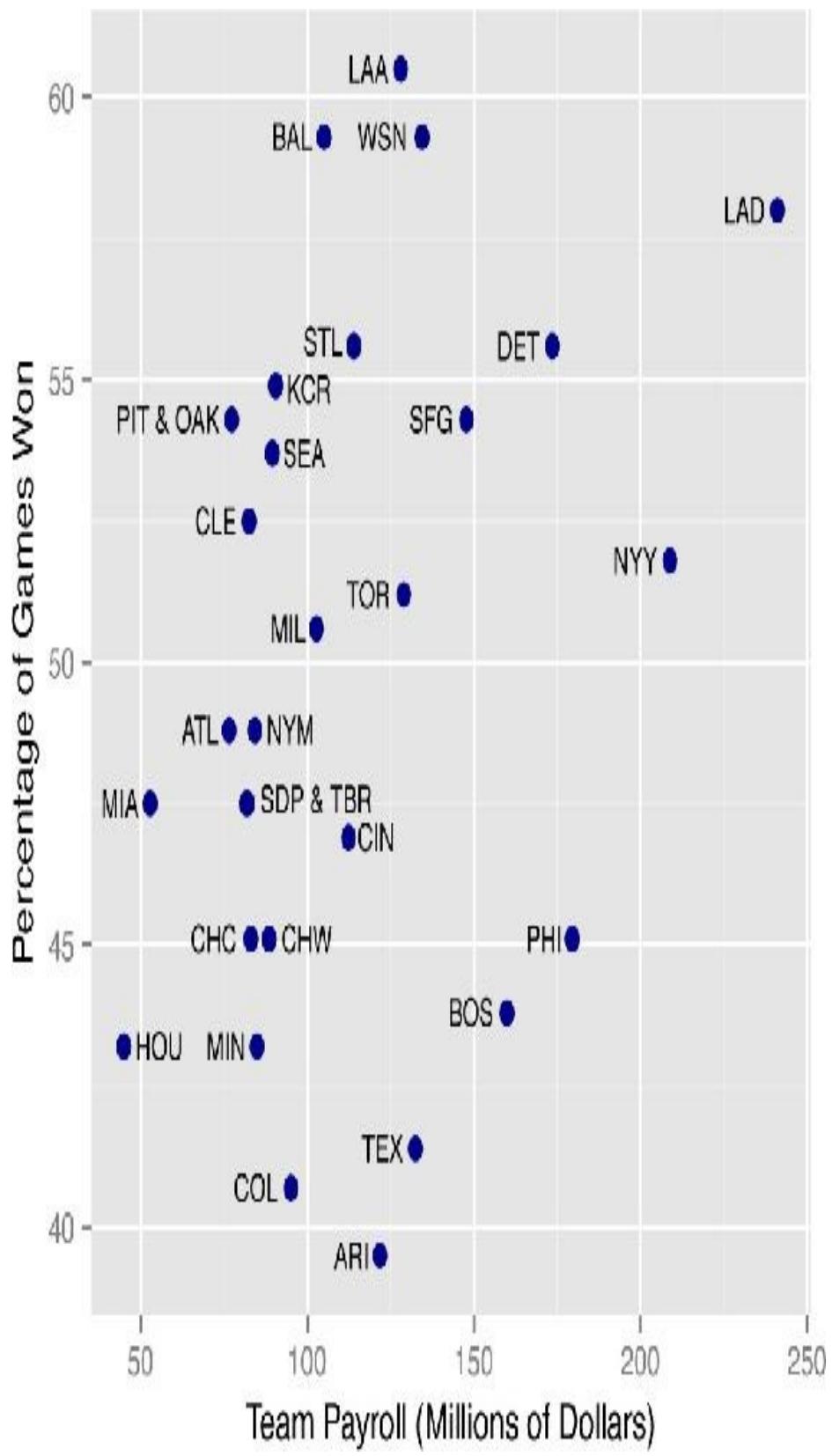


Sources. spotrac ([2015a](#), [2015b](#), [2015c](#)).

Figure 1.1. MLB, NBA, and NFL Average Annual Salaries

Do team expenditures on players buy success? This is a meaningful question to ask for leagues that have no salary caps. Szymanski ([2015](#)) reports studies showing that between 60 and 90 percent of the variability in U.K. soccer team positions may be explained by wages paid to players. Major League Baseball has a luxury tax in place of a salary cap, and team payrolls vary widely in size. The New York Yankees have been known for having the highest payrolls in baseball. Recently, the Los Angeles Dodgers have surpassed the Yankees with the highest player payroll—more than \$257 million at the end of the 2014 season ([Woody 2014](#)).

Figure 1.2 shows baseball team salaries at the beginning of the 2014 season plotted against the percentage of games won across the regular season. Notice how teams that made the playoffs in 2014, labeled with team abbreviations, have a wide range of payrolls. While the biggest spenders in baseball are often among the set of teams going to the playoffs, the relationship between team payrolls and team performance is weak at best—less than 7 percent of the variability in win/loss percentages is explained by player payrolls.



Sources. Sports Reference LLC (2015b) and USA Today (2015).

See Appendix B, page 255, for team abbreviations and names.

**Figure 1.2. MLB Team Payrolls and Win/Loss Performance
(2014 Season)**

The thesis of Michael Lewis' *Moneyball* (2003) and what has become the ethos of sports analytics is that small-market baseball teams can win by spending their money wisely. Star players demand top salaries due as much to their celebrity status as to their skills. Players with high on-base percentages, overlooked by major-market teams, can be hired at much lower salaries than star players.

Teams, although associated with particular cities, can be known nationwide or worldwide. The media of television and the Internet provide opportunities for reaching consumers across the globe. A Super Bowl at the Rose Bowl in Pasadena, California or AT&T Stadium in Arlington, Texas may be attended by around 100 thousand fans (Alder 2015), while U.S. television audiences have grown to over 100 million (statista 2015).

Media revenues are important to successful sports teams. Other revenues come from business partnerships, sponsorships, advertising, and stadium naming rights. City governments understand well the power of sports to promote business. Locating sports arenas in cities can help to revitalize downtown areas, as demonstrated by the experience of the Oklahoma City Thunder. Indianapolis, Indiana promotes itself as a sports capital with the Colts and Pacers (Rein, Shields, and Grossman 2015).

Teams seek to build their brands, developing a positive reputation in the minds of consumers. Players, like fans, are attracted to teams with a reputation for hard work, courage,

fair play, honesty, teamwork, and community service. The character of a team is often as important as its likelihood of winning. The Cubs are associated with Chicago, but Cub fans may be found from Maine to California. This is despite the fact that the Cubs have not won the World Series since 1908. Teams in U.S. professional sports vie to become “America’s team,” with fans across the land wearing their logoembossed hats and jerseys.

The demand for sports and the feelings of sports consumers are not so easily understood. Fans can be fickle and fandom fleeting. Fans can be loyal to a sport, to a team, or to individual players. Multivariate methods can help us understand how sports consumers think by revealing relationships among products or brands.

Figure 1.3 provides an example, a *perceptual map* of seven sports. Along the horizontal dimension, we move from individual, non-contact sports on the left-hand side, to team sports with little contact, to team sports with contact on the right-hand side. The vertical dimension, less easily described, may be thought of as relating to the aerobic versus anaerobic nature of sports and to other characteristics such as physicality and skill. Sports such as tennis, soccer, and basketball entail aerobic exercise. These are endurance sports, while football is an example of a sport that involves both aerobic and anaerobic exercise, including intense exercise for short durations. Sports close together on the map have similarities. Baseball and golf, for example, involve special skills, such as precision in hitting a ball. Soccer and hockey involve almost continuous movement and getting a ball through the goal. Football and hockey have high physicality or player contact.

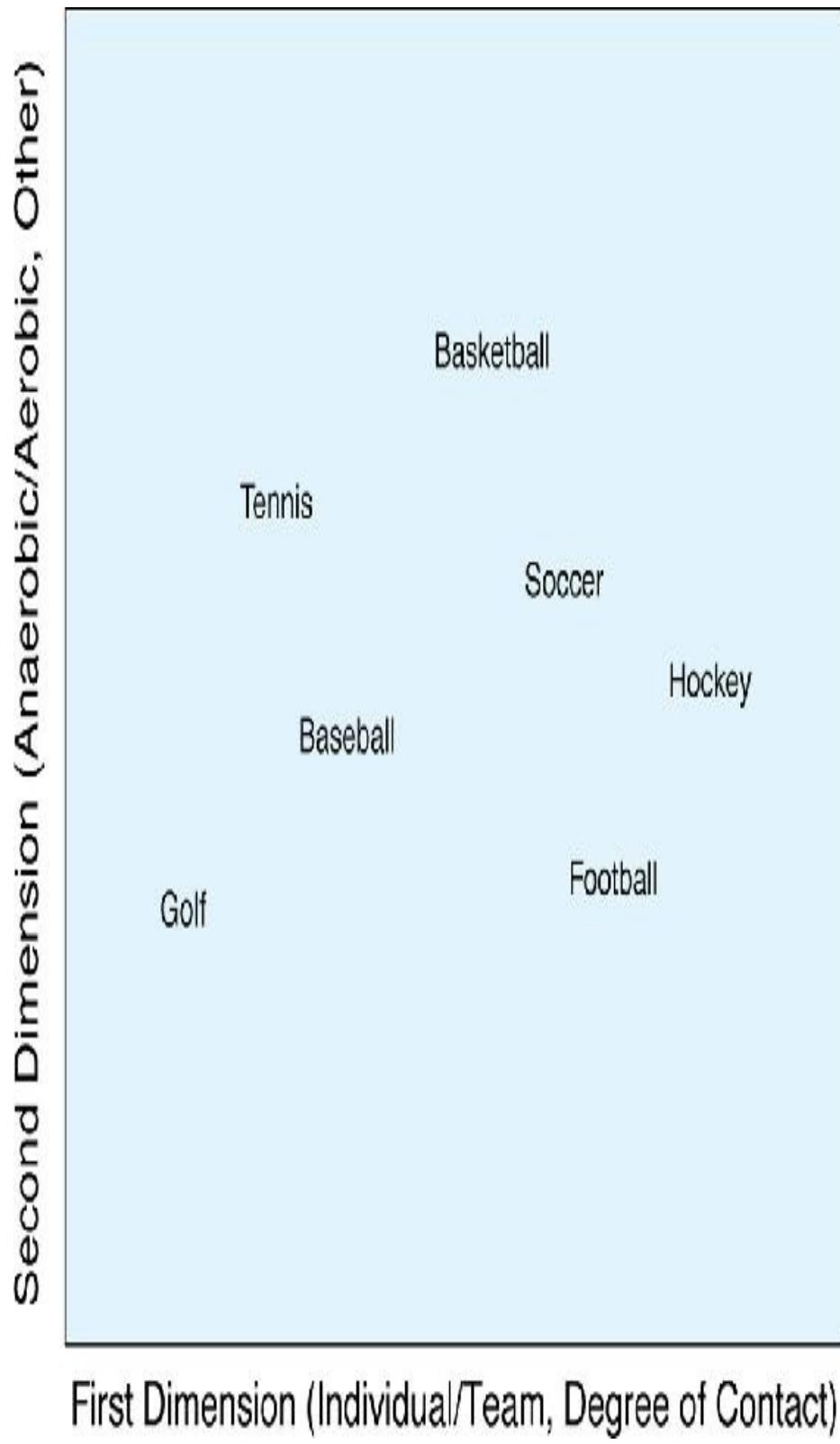


Figure 1.3. A Perceptual Map of Seven Sports

In many respects, professional sports teams are decidedly different from other businesses. They are in the public eye. They live and die in the media. And a substantial portion of their revenues come from media.

Késenne (2007), Szymanski (2009), Fort (2011), Fort and Winfree (2013), Leeds and von Allmen (2014), and the edited volumes of Humphreys and Howard (2008a, 2008b, 2008c) review sports economics and business issues.

Gorman and Calhoun (1994) and Rein, Shields, and Grossman (2015) focus on alternative sources of revenue for sports teams and how these relate to business strategy. The business of baseball has been the subject of numerous volumes (Miller 1990; Zimbalist 1992; Powers 2003; Bradbury 2007; Pessah 2015). And Jozsa (2010) reviews the history of the National Basketball Association.

An overview of sports marketing is provided by Mullin, Hardy, and Sutton (2014). Rein, Kotler, and Shields (2006) and Carter (2011) discuss the convergence of entertainment and sports. Miller (2015a) reviews methods in marketing data science, including product positioning maps, market segmentation, target marketing, customer relationship management, and competitive analysis.

Sports also represents a laboratory for labor market research. Sports is one of the few industries in which job performance and compensation are public knowledge. Economic studies examine player performance measures and value of individual players to teams (Kahn 2000; Bradbury 2007). Miller (1991), Abrams (2010), and Lowenfish (2010) review baseball labor relations. And Early (2011) provides insight into labor and racial discrimination in professional sports.

Sports wagering markets have been studied extensively by economists because they provide public information about price, volume, and rates of return. Furthermore, sports betting opportunities have fixed beginning and ending times and published odds or point spreads, making them easier to study than many financial investment opportunities. As a result, sports wagering markets have become a virtual field laboratory for the study of market efficiency. Sauer (1998) provides a comprehensive review of the economics of wagering markets.

When management objectives can be defined clearly in mathematical terms, teams use mathematical programming methods—constrained optimization. Teams attempt to maximize revenue or minimize costs subject to known situational factors. There has been extensive work on league schedules, for which the league objective may be to have teams playing one another an equal number of times while minimizing total distance traveled between cities.

Alternatively, league officials may seek home/away schedules, revenue sharing formulas, or draft lottery rules that maximize competitive balance. Briskorn (2008) reviews methods for scheduling sports competition, drawing on integer programming, combinatorics, and graph theory. Wright (2009) provides an overview of operations research in sport.

Extensive data about sports are in the public domain, readily available in newspapers and online sources. These data offer opportunities for predictive modeling and research.

Throughout the book we also identify places to apply methods of operations research, including mathematical programming and simulation.

Exhibit 1.1 shows an R program for exploring distributions of player salaries across the MLB, NBA, and NFL. The program draws on software for statistical graphics from Sarkar (2008).

Exhibit 1.2 (page 18) shows an R program for examining the relationship between MLB payrolls and win-loss performance. The program draws on software for statistical graphics from Wickham and Chang (2014).

Exhibit 1.3 (page 19) shows an R program to obtain a perceptual map of seven sports, showing their relationships with one another. The program draws on modeling software for multidimensional scaling.

Exhibit 1.1. MLB, NBA, and NFL Player Salaries (R)

[**Click here to view code image**](#)

```
# MLB, NBA, and NFL Player Salaries (R)

library(lattice) # statistical graphics

# variables in contract data from spotrac.com
(August 2015)
#   player: player name (contract years)
#   position: position on team
#   team: team abbreviation
#   teamsignedwith: team that signed the
original contract
#   age: age in years as of August 2015
#   years: years as player in league
#   contract: dollars in contract
#   guaranteed: guaranteed dollars in contract
#   guaranteedpct: percentage of contract
dollars guaranteed
#   salary: annual salary in dollars
#   yearfreeagent: year player becomes free
agent
```

```

#
#   additional created variables
#   salarymm: salary in millions
#   leaguename: full league name
#   league: league abbreviation

# read data for Major League Baseball
mlb_contract_data <-
read.csv("mlb_player_salaries_2015.csv")
mlb_contract_data$leaguename <- rep("Major
League Baseball",
length = nrow(mlb_contract_data))
for (i in seq(along =
mlb_contract_data$yearfreeagent))
  if (mlb_contract_data$yearfreeagent[i] ==
0)
    mlb_contract_data$yearfreeagent[i] <-
NA
for (i in seq(along = mlb_contract_data$age))
  if (mlb_contract_data$age[i] == 0)
    mlb_contract_data$age[i] <- NA
mlb_contract_data$salarymm <-
mlb_contract_data$salary/1000000
mlb_contract_data$league <- rep("MLB", length
= nrow(mlb_contract_data))
print(summary(mlb_contract_data))
# variables for plotting
mlb_data_plot <- mlb_contract_data[,,
c("salarymm","leaguename")]

nba_contract_data <-
read.csv("nba_player_salaries_2015.csv")
nba_contract_data$leaguename <- rep("National
Basketball Association",
length = nrow(nba_contract_data))
for (i in seq(along =
nba_contract_data$yearfreeagent))
  if (nba_contract_data$yearfreeagent[i] ==
0)
    nba_contract_data$yearfreeagent[i] <-

```

```

NA
for (i in seq(along = nba_contract_data$age))
  if (nba_contract_data$age[i] == 0)
    nba_contract_data$age[i] <- NA
nba_contract_data$salarymm <-
nba_contract_data$salary/1000000
nba_contract_data$league <- rep("NBA", length
= nrow(nba_contract_data))
print(summary(nba_contract_data))
# variables for plotting
nba_data_plot <- nba_contract_data[,,
c("salarymm","leaguename")]

nfl_contract_data <-
read.csv("nfl_player_salaries_2015.csv")
nfl_contract_data$leaguename <- rep("National
Football League",
  length = nrow(nfl_contract_data))
for (i in seq(along =
nfl_contract_data$yearfreeagent))
  if (nfl_contract_data$yearfreeagent[i] ==
0)
    nfl_contract_data$yearfreeagent[i] <-
NA
for (i in seq(along = nfl_contract_data$age))
  if (nfl_contract_data$age[i] == 0)
    nfl_contract_data$age[i] <- NA
nfl_contract_data$salarymm <-
nfl_contract_data$salary/1000000
nfl_contract_data$league <- rep("NFL", length
= nrow(nfl_contract_data))
print(summary(nfl_contract_data))
# variables for plotting
nfl_data_plot <- nfl_contract_data[,,
c("salarymm","leaguename")]

# merge contract data with variables for
plotting
plotting_data_frame <- rbind(mlb_data_plot,
nba_data_plot, nfl_data_plot)

```

```
# generate the histogram lattice for comparing
player salaries
# across the three leagues in this study
lattice_object <- histogram(~salarymm |
leaguename, plotting_data_frame,
    type = "density", xlab = "Annual Salary ($
millions)", layout = c(1,3))

# print to file
pdf(file =
"fig_understanding_markets_player_salaries.pdf",
width = 8.5, height = 11)
print(lattice_object)
dev.off()
```

**Exhibit 1.2. Payroll and Performance in Major League
Baseball (R)**

[Click here to view code image](#)

```
# Payroll and Performance in Major League
Baseball (R)

library(ggplot2) # statistical graphics

# functions used with grid graphics to split
the plotting region
# to set margins and to plot more than one
ggplot object on one page/screen
vplayout <- function(x, y)
viewport(layout.pos.row=x, layout.pos.col=y)

# user-defined function to plot a ggplot
object with margins
ggplot.print.with.margins <-
function(ggplot.object.name,
left.margin.pct=10,
```

```

    right.margin.pct=10,top.margin.pct=10,bottom.
  { # begin function for printing ggplot
objects with margins
    # margins expressed as percentages of
total... use integers
    grid.newpage()
    pushViewport(viewport(layout=grid.layout(100,1
    print(ggplot.object.name,
    vp=vplayout((0 + top.margin.pct):(100 -
bottom.margin.pct),
    (0 + left.margin.pct):(100 -
right.margin.pct)))
} # end function for printing ggplot
objects with margins

# read in payroll and performance data
# including annotation text for team
abbreviations
mlb_data <-
read.csv("mlb_payroll_performance_2014.csv")
mlb_data$millions <- mlb_data$payroll/1000000
mlb_data$winpercent <- mlb_data$wlpct * 100

cat("\nCorrelation between Payroll and
Performance:\n")
with(mlb_data, print(cor(millions,
winpercent)))

cat("\nProportion of win/loss percentage
explained by payrolls:\n")
with(mlb_data, print(cor(millions,
winpercent)^2))

pdf(file =
"fig_understanding_markets_payroll_performance.pdf
    width = 5.5, height = 5.5)
ggplot_object <- ggplot(data = mlb_data,
    aes(x = millions, y = winpercent)) +
    geom_point(colour = "darkblue", size = 3)
+

```

```
    xlab("Team Payroll (Millions of Dollars)")  
    +  
    ylab("Percentage of Games Won") +  
    geom_text(aes(label = textleft), size = 3,  
    hjust = 1.3) +  
    geom_text(aes(label = textright), size =  
    3, hjust = -0.25)  
  
    ggplot.print.with.margins(ggplot_object,  
    left.margin.pct = 5,  
    right.margin.pct = 5, top.margin.pct = 5,  
    bottom.margin.pct = 5)  
  
dev.off()
```

Exhibit 1.3. Making a Perceptual Map of Sports (R)

[Click here to view code image](#)

```
# Making a Perceptual Map of Sports (R)  
  
library(MASS) # includes functions for  
multidimensional scaling  
library(wordcloud) # textplot utility to  
avoid overlapping text  
  
USE_METRIC_MDS <- FALSE # metric versus  
nonmetric toggle  
  
# utility function for converting a distance  
structure  
# to a distance matrix as required for some  
routines and  
# for printing of the complete matrix for  
visual inspection.  
make.distance.matrix <-  
function(distance_structure)  
{ n <- attr(distance_structure, "Size")
```

```

    full <- matrix(0,n,n)
    full[lower.tri(full)] <-
distance_structure
    full+t(full)
}

# enter data into a distance structure as
required for various
# distance-based routines. That is, we enter
the upper triangle
# of the distance matrix as a single vector of
distances
distance_structure <-
as.single(c(9,11,10,5,14,4,15,6,12,13,16,1,18,

# provide a character vector of sports names
sport_names <- c("Baseball", "Basketball",
"Football",
"Soccer", "Tennis", "Hockey", "Golf")

attr(distance_structure, "Size") <-
length(sport_names) # set size attribute

# check to see that the distance structure has
been entered correctly
# by converting the distance structure to a
distance matrix
# using the utility function
make.distance.matrix, which we had defined
distance_matrix <-
unlist(make.distance.matrix(distance_structure))
cat("\n","Distance Matrix of Seven
Sports","\n")
print(distance_matrix)

if (USE_METRIC_MDS)
{
  # apply the metric multidimensional
  scaling algorithm and plot the map
  mds_solution <

```

```
cmdscale(distance_structure, k=2, eig=T)
}

# apply the nonmetric multidimensional scaling
# algorithm
# this is more appropriate for rank-order data
# and provides a more satisfactory solution
here

if (!USE_METRIC_MDS)
{
  mds_solution <- isoMDS(distance_matrix, k
= 2, trace = FALSE)

}

pdf(file =
"plot_nonmetric_mds_seven_sports.pdf",
width=8.5, height=8.5) # opens pdf
plotting device
# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))

# original solution
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2]

# set up the plot but do not plot points...
use names for points
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # first page of pdf plots
# We plot the sport names in the locations where points normally go.
text(First_Dimension, Second_Dimension, labels
= sport_names,
      offset = 0.0, cex = 1.5)
title("Seven Sports (initial solution)")
```

```

# reflect the horizontal dimension
# multiply the first dimension by -1 to get
reflected image
First_Dimension <- mds_solution$points[,1] *
-1
Second_Dimension <- mds_solution$points[,2]
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # second page of pdf plots
text(First_Dimension, Second_Dimension, labels
= sport_names,
      offset = 0.0, cex = 1.5)
title("Seven Sports (horizontal reflection)")

# reflect the vertical dimension
# multiply the section dimension by -1 to get
reflected image
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2] *
-1
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # third page of pdf plots
text(First_Dimension, Second_Dimension, labels
= sport_names,
      offset = 0.0, cex = 1.5)
title("Seven Sports (vertical reflection)")

# multiply the first and second dimensions by
-1
# for reflection in both horizontal and
vertical directions
First_Dimension <- mds_solution$points[,1] *
-1
Second_Dimension <- mds_solution$points[,2] *
-1
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,

```

```

    xlim = c(-15, 15), ylim = c(-15, 15)) #
fourth page of pdf plots
text(First_Dimension, Second_Dimension, labels
= sport_names,
    offset = 0.0, cex = 1.5)
title("Seven Sports (horizontal and vertical
reflection)")
dev.off() # closes the pdf plotting device

pdf(file =
"plot.pretty_original_mds_seven_sports.pdf",
width=8.5, height=8.5) # opens pdf
plotting device
# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] #
no reflection
Second_Dimension <-
mds_solution$points[,2] # no reflection
# wordcloud utility for plotting with no
overlapping text
textplot(x = First_Dimension,
        y = Second_Dimension,
        words = sport_names,
        show.lines = FALSE,
        xlim = c(-15, 15), # extent of horizontal
axis range
        ylim = c(-15, 15), # extent of vertical
axis range
        xaxt = "n", # suppress tick marks
        yaxt = "n", # suppress tick marks
        cex = 1.15, # size of text points
        mgp = c(0.85, 1, 0.85), # position of
axis labels
        cex.lab = 1.5, # magnification of axis
label text
        xlab = "",
        ylab = "")
dev.off() # closes the pdf plotting device

```

```
pdf(file = "fig_sports_perceptual_map.pdf",
     width=8.5, height=8.5) # opens pdf
# plotting device
# use par(mar = c(bottom, left, top, right))
# to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] *
-1 # reflect horizontal
Second_Dimension <- mds_solution$points[,2]
# wordcloud utility for plotting with no
overlapping text
textplot(x = First_Dimension,
          y = Second_Dimension,
          words = sport_names,
          show.lines = FALSE,
          xlim = c(-15, 15), # extent of horizontal
axis range
          ylim = c(-15, 15), # extent of vertical
axis range
          xaxt = "n", # suppress tick marks
          yaxt = "n", # suppress tick marks
          cex = 1.15, # size of text points
          mgp = c(0.85, 1, 0.85), # position of
axis labels
          cex.lab = 1.5, # magnification of axis
label text
          xlab = "First Dimension (Individual/Team,
Degree of Contact)",
          ylab = "Second Dimension
(Anaerobic/Aerobic, Other")
dev.off() # closes the pdf plotting device
```

2. Assessing Players

Pete: "Gus, did you ever think in a million years computers would be a part of this game?"

Gus: "Computers? Anyone uses computers doesn't know a damn thing about this game."

Pete: "You know, if you wanted to, you could access any high school or college roster, pull the stats on any player at any time. You wouldn't have to waste your time with all these papers."

Gus: "I'm not wasting my time. I enjoy doing this."

Pete: "You know, they got a special program now that can calculate a player's stats and, based on the competition he's seen, tell you whether or not he's ready for the next level. You believe that?"

Gus: "Yeah, what else does it tell you? When to scratch your ass?"

Pete: "I don't like them either, but they're part of the business now."

Gus: "Pete, scouts, good scouts are the heart of this game. They decide who's gonna play and, if they're lucky, they decide how it's gonna be played. But a computer can't tell if a kid's got instincts or not, or if he can hit a cutoff man, or hit behind the runner. . . or look into a kid's face that's just gone oh-for-four and know if he's gonna be able to come back like nothing's happened. No, a computer can't tell you all that crap, I'll tell you. No."

—JOHN GOODMAN AS PETE KLINE, AND CLINT EASTWOOD AS

GUS LOBEL IN *Trouble with the Curve* (2012)

The job of sports performance analytics is to understand how various factors contribute to success on the fields and courts of play, and this job begins with **measurement**. Many factors contribute to success in sport. There are measures that relate to physical stature, biophysics, health, fitness, and conditioning. There is athleticism and measures dealing with speed, power, strength, flexibility, and agility. There are psychological measures of intelligence, personality, and attitude. Finally, there are measures relating to proficiency in sport—knowledge, skill, and execution in practice and in games.

Let us step back from the things we see and hear about on a regular basis—the language of sportscasting—and ask basic measurement questions. What do we want from performance measures in sports? What makes a measure reliable? What makes a measure valid?

We use the term ***reliability*** to refer to the trustworthiness or repeatability of measurement procedures. We consider the degree to which repeated measures of the same trait at the same time agree with one another, as in test-retest reliability or split-half reliability. When assessed with a multi-item survey, we ask that a measure have internal consistency.

When we talk about ***validity***, we are thinking of the degree to which a measure measures what it is supposed to measure. There are subjective assessments of *face validity* or *content validity*. We examine measurements to see the degree to which they appear to measure the traits they are supposed to measure.

A more objective approach to validity assessment would be to demonstrate *predictive validity*. Knowing how two traits are

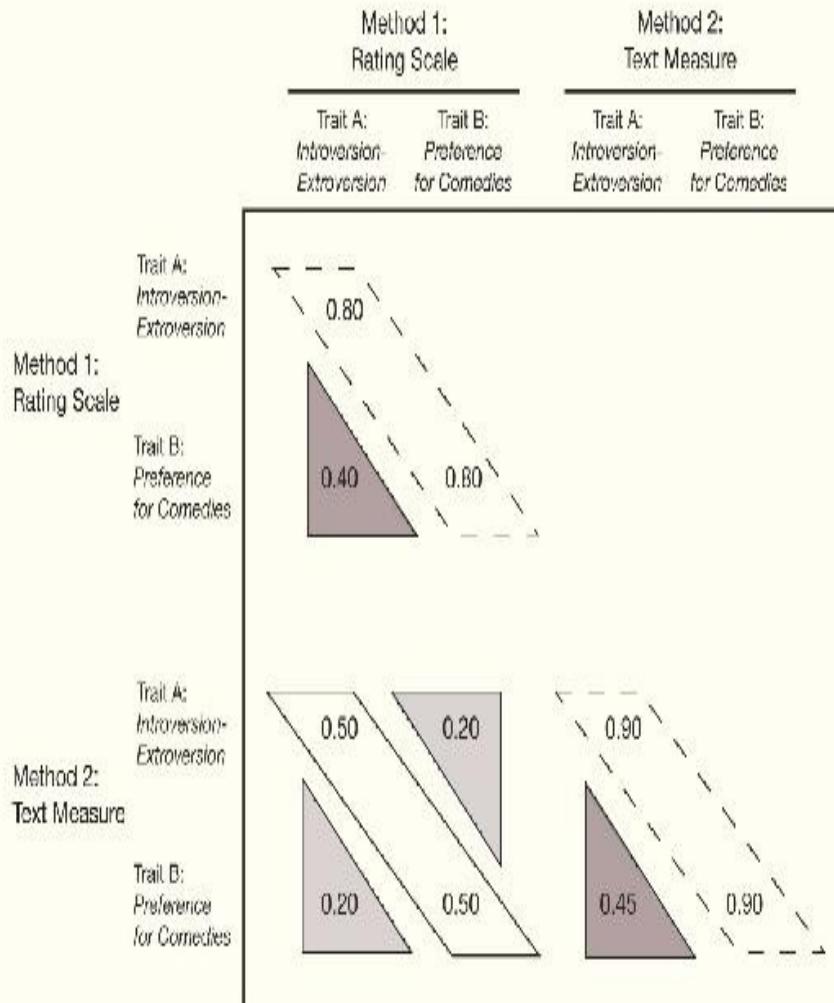
related in theory, we can create measures of those traits and examine the degree to which these measures relate as theory suggests. The meaning of a measure is defined by its relationship to other measures. This is *construct validity*, a logical extension of predictive validity.

Campbell and Fiske (1959) define reliability and validity as follows:

Reliability is the agreement between two efforts to measure the same trait through maximally similar methods. Validity is represented in the agreement between two attempts to measure the same trait through maximally different methods. (83)

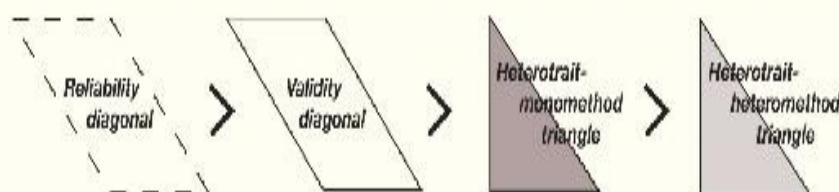
The prototypical measurement study involves the multitraitmultimethod matrix, as shown in figure 2.1

The meaning of a measure is defined by its relationships to other measures.



Convergent validity is demonstrated by high correlations in the validity diagonal.

Discriminant validity is demonstrated by the relative sizes of correlations in diagonals and triangles:



Method variance is demonstrated by relatively high heterotrait-monomethod correlations when traits are assumed to be uncorrelated.

Figure 2.1. MultitraitMultimethod Matrix for Baseball Measures

The multitraitmultimethod matrix has rows and columns associated with traits (attributes) and methods (measurement procedures). Each element of the matrix represents a trait-method unit. The components of the matrix are the reliability diagonal, validity diagonal, heterotrait-monomeasure triangles, and heterotrait-heteromeasure triangles.

Figure 2.1 shows a hypothetical multitraitmultimethod matrix with four baseball measures. It portrays two underlying traits: general hitting ability and power hitting ability, measuring them both in practice and games. In batting practice, we first ask the player to hit each ball pitched as cleanly as he can and determine the proportion of balls hit in fair territory. Then we ask him to hit each ball pitched as far as he can and determine the proportion of balls hit out of the park, much as we would see in a home run derby. For game-day measures, we refer to box scores to obtain the player's batting average and home run rate (home runs per at bat). This gives four distinct measures. We compute the correlations between all pairs of measures and show the results in the matrix.

To assess reliability for training measures, we would use the same measurement procedures on numerous days and compute the average intercorrelation across all pairs of measures. And to assess reliability for game-day measures, we would compute correlations between measures from odd-numbered games with even-numbered games (or alternatively, we could compute correlations between measures across many random split-halves of games and average those correlations).

Reliability refers to measures of the same trait in the same way

at about the same time. The reliability of sports performance measures is very high because these are objective measures based on counts. Aside from variations across official scorers, there is little subjectivity or opinion involved in these measures. We expect high correlations in the reliability diagonals of the multitraitmultimethod matrix.

What else do we expect to see in a multitraitmultimethod matrix? We should see different measures of the same trait correlating positively on the validity diagonal. Hitting in batting practice should have a positive correlation with hitting in games. We expect measures of the same trait to correlate more highly with one another than with measures of different traits. Accordingly, we should see higher correlations on the validity diagonal than in either the heterotrait-monomethod or the heterotrait-heteromethod triangles.

The meaning of a measure is defined by its relationships to other measures. This notion of construct validity is illustrated by the multitraitmultimethod matrix and what Campbell and Fiske (1959) call convergent validity and discriminant validity. Convergent validity refers to the idea that different measures of the same trait should converge. That is, different measures of the same trait or attribute should have relatively high correlations. Discriminant validity refers to the notion that measures of different traits should diverge. In other words, measures of different traits should have lower correlations than measures of the same trait. Convergent and discriminant validation are part of what we mean by construct validation. The meaning of a measure is defined in terms of its relationship to other measures.

What we expect in theory depends on the traits being

measured. If we took running instead of general hitting ability as our first trait being studied, we would expect very different set of results. In practice, we might also measure a player's time in the 40-yard dash, perhaps converting the time in milliseconds to miles per hour. For the game-day measures, we refer to box scores to obtain the player's success rate in stolen bases (stolen bases divided by stolen bases attempted). What would we expect to see for these measures in relation to hitting for power? Running ability and hitting for power would be expected to be uncorrelated or negatively correlated.

Discussions of validity touch on fundamental issues in the philosophy of science—issues of theory construction, measurement, and testability. There are no easy answers here. If the theory is correct and the measures valid, then the pattern of relationships among the measures should be similar to the pattern predicted by theory. To the extent that this is true for observed data, we have partial confirmation of the theory and, at the same time, demonstration of construct validity. But what if the predictions do not pan out? Then we are faced with a dilemma: the theory could be wrong, one or more of the measures could be invalid, or we could have observed an event of low probability with correct theory and valid measures.

Regarding measurement and philosophy of science, I like the umpire story:

After a long day of disputed calls at the ballpark, three umpires are asked to justify their methods. The first umpire, an empiricist by persuasion, says, *I call them as I see them*. The second, with the faith of a philosophical realist, replies, *I call them as they are*. Not to be outdone, the third umpire, with the self-

proclaimed authority of an operationist or logical empiricist, says, *The way I call them—that's the way they are.*

S. S. Stevens (1946) wrote *On the Theory of Scales of Measurement*, an influential article identifying four general types of measures: nominal, ordinal, interval, and ratio. It was the strength of Stevens' convictions, perhaps more than the strength of his argument, that influenced generations of researchers. The words he chose to describe levels of measurement seemed to carry the force of law. He talked about the formal properties of scales and “permissible statistics,” arguing that “the statistical manipulations that can legitimately be applied to empirical data depend on the type of scale” (Stevens 1946, 677). Stevens argued that we could compute means, standard deviations, and correlations with interval and ratio measures, but not with ordinal measures.

Table 2.1 summarizes scale types or levels of measurement from Stevens (1946). The formal definition of a scale type follows from its mathematical properties, or what Stevens called its “mathematical group structure.” This refers to the set of data transformations that, when used on the original measures, will create new measures with the same scale properties as the original measures.

<i>Level of Measurement (Scale Type)</i>	<i>Basic Empirical Operations</i>	<i>Mathematical Group Structure</i>	<i>Examples of Permissible Statistics</i>
Nominal	equality, numbers like names	one-to-one correspondence	number of cases in class, frequency table, modal class
Ordinal	greater than, less than	one-to-one monotonic	median, percentiles, rank-order correlation
Interval	equality of intervals	one-to-one linear	mean, standard deviation, product-moment correlation
Ratio	equality of ratios	one-to-one linear, preserving the zero point	same statistics as interval level

Source: Adapted from Stevens (1946).

Table 2.1. *Levels of Measurement*

For nominal scales, any one-to-one transformation will preserve the number of categories and, hence, the scale's

essential property. For ordinal scales, any one-to-one monotonic transformation will preserve the property of order. For interval scales, any one-to-one linear transformation, a function of the form $y = ax + b$, will preserve the properties of the scale. Ratio scales are similar to interval scales, except that the zero point must be preserved. Accordingly, for ratio scales, a data transformation that preserves its properties must have the form $y = ax$.

Researchers following Stevens' dictums constitute the weak measurement school. They argue that many measures are ordinal rather than interval and that statistics relying on sums or differences, including means and variances, would be inappropriate for ordinal measures. Researchers following the strong statistics school, on the other hand, argue that statistical methods make no explicit assumptions about the meaning of measurements or their relationships to underlying dimensions. Strong statistics can be used with weak measurements.

For practical purposes, we ask whether or not a variable has meaningful magnitude. If a variable is categorical, it lacks meaningful magnitude. One further observation is appropriate for categorical data: we note whether the variable is binary (taking only two possible values) or multinomial (taking more than two possible values). If we can make these simple distinctions across measures, we can do much useful research. Note that most sports performance measures begin as counts, which are ratio measures. And, despite the objections of weak measurement believers, there are many situations in which computing the mean of ranks makes perfectly good sense. Many sports analysts confuse reliability with stability. In baseball, for example, they note low correlations for player

batting averages from one year to the next or pitcher earned run averages from one year to the next, saying that this is evidence of low reliability. This is incorrect thinking. Baseball measures and sports performance measures in general have very high reliability.

Reliability concerns agreement between measures of the same trait in the same way at about the same time. The reliability of sports performance measures is very high because these are objective measures based on counts.

Many sports performance measures rely on the official scoring of events on the field, box scores, and play-by-play logs.

Official records, counts, and mathematical formulas for computing performance measures do not change from one observer or one analyst to the next. Many of the newer measures of player and ball location on the fields and courts of play, player running speed, and efficiency in getting to balls in play are obtained through electronic devices with little or no human intervention. These are highly reliable and trustworthy. What performance measures lack is not reliability, but stability from one year to the next or one game to the next.

Measurement is the assignment of numbers to attributes according to rules, and measurements themselves have certain desirable attributes:

- **Reliable.** A measure should be trustworthy and repeatable.
- **Valid.** A measure should measure the attribute it is said to measure.
- **Explicit.** Procedures should be unambiguous and defined in detail, so that each research worker obtains

the same values when using the measurement procedure.

- **Accessible.** A measure should come from data that are easily obtained.
- **Tractable.** A measure should be easy to work with and easy to utilize in methods and models.
- **Comprehensible.** A measure should be simple and straightforward, so it is easily understood and interpreted.
- **Transparent.** The method of measurement should be documented fully, so research workers can share results with one another in a spirit of open and honest scientific inquiry. There should be no trade secrets in science.

Sports performance measures vary widely in the degree to which they possess these attributes. Some measures are dependent on timing or tracking devices in stadia, and may be accessible only to leagues and teams.

Regarding performance on the field, we record play-by-play events, compute box scores, and note standings of teams. We develop general measures of offensive and defensive performance and rate players and teams. Baseball has its Sabermetrics and Moneyball. And other sports have followed suit, designing numerous measures of player performance and using them to make personnel decisions.

Psychometrics has “standardized testing,” defining explicit procedures that all test administrators must follow when making measures. Explicit, unambiguous procedures promote reliability and reproducibility.

Measures in baseball serve to illustrate measurement principles. Batting average (BA) is a simple proportion with at

bats as a divisor. We look for players with batting averages above 0.250, and batting at or above 0.300 is a goal of many hitters. Batting below 0.200, sometimes referred to as “the Mendoza line,” is not a good sign for hitters. Batting average is easily understood, but criticized as a measure of hitting ability because it fails to consider the value of walks. This is a concern about the measure’s validity.

On-base percentage (OBP) is very easy to explain. Using plate appearances rather than at bats in the divisor, OBP reflects the proportion of times that a hitter reaches first base or beyond. For OBP, we look for players whose values are around 0.333, getting on base one in every three plate appearances. OBP is well known and well understood, partly as a result of its use in *Moneyball* (Lewis 2003). But it is criticized because it fails to consider the value of extra-base hits. This, too, is a concern about measurement validity.

Slugging percentage (SLG) is a ratio: the number of total bases per at bat. It is not especially hard to understand, although it is not a percentage as its name would imply. And because it is neither a percentage nor an average, it is difficult to understand. A fan has to be “in the know” to understand that an SLG value of 0.300 is bad or a value of 0.500 is very good. Babe Ruth’s lifetime SLG was 0.690, which is very good. SLG suffers by being less comprehensible than other measures of hitting ability.

On-base percentage plus slugging (OPS) is a simple sum of OBP and SLG. It is neither a percentage like OBP, nor a ratio like SLG. The intent of OPS was to provide an index of ability that would reflect both getting on base and hitting with power. OPS is another measure for which a person needs to be “in the

know” to understand. The average regular-season OPS across the thirty MLB teams in the 2014 season was 0.700 (Sports Reference LLC 2015a). OPS makes little intuitive sense.

There is no justification for weighting OBP and SLG equally in measuring hitting ability, and adding a proportion to a ratio gives a measure with no known units. Accordingly, OPS is neither comprehensible nor valid.

Tango, Lichtman, and Dolphin (2007) have created an alternative to OPS called the on-base average (OBA), computed as a weighted linear combination of various hitting measures. Their intent is to define a measure that gives reasonable, data-based weights to getting on base and hitting for power, a measure that is then scaled so it has values to conform to OBP. Tango, Lichtman, and Dolphin (2007) make a strong case for using OBA instead of OPS. But their method for calculating OBA is complicated.¹ OBA cannot be easily explained in words, so it fails as a general index of hitting ability.

¹ Tango, Lichtman, and Dolphin (2007) provide the formula for weighted on-base average (OBA):

$$wOBA = \frac{0.72 \times NIBB + 0.75 \times HBP + 0.90 \times 1B + 0.92 \times RBOE + 1.24 \times 2B + 1.56 \times 3B + 1.95 \times HR}{PA}$$

where NIBB is the number of intentional bases on balls, HBP is number of times a player is hit by a pitch, 1B is the number of singles, RBOE is the number of times a batter reaches base on an error, 2B is the number of doubles, 3B is the number of triples, and HR is the number of home runs. PA refers to plate appearances, which may or may not exclude bunts, intentional bases on balls, and other events described as “obscure.”

Simple, comprehensible measures are preferred to complex measures because simple measures are easier to explain to fans, coaches, and managers. Among the most comprehensible measures are simple percentages or proportions computed

from the events of a game.

In baseball, much time and effort has been devoted to attempts at finding the best single measure of player prowess. A five-tool player in baseball is a player with strong skills for running, fielding, throwing, hitting, and hitting with power. How can we combine measures of these five traits into a single measure reflecting a player's contribution to his team?

Comprehensive player evaluation is illustrated by measures of points or wins above replacement. The general idea is to assess player abilities in hitting, fielding, base running, and throwing/pitching relative to a norm group (referred to as replacement players) and then to combine those norm-group-scaled assessments.

We ask, "What is a player's value to his team? What if he were replaced by another player who is available to play, a player of average ability at the same position?" Wins above replacement is usually expressed in units of wins across the regular season, with ten runs being equivalent to one win. If a player's wins-above-replacement value is 5, say, then that player's team can expect to win five fewer games across the entire season if he must be replaced.

Wins-above-replacement measures fail the transparency test when methods of calculation are closely held company secrets. This is a special problem for norm-group-based measures because their meaning rests on the choice of norm group. If we do not know who the replacement players are, then we cannot accurately interpret wins-above-replacement. Furthermore, there is no way of checking the calculations of for-profit companies that refuse to publish their formulas and data. These measures are not in keeping with the spirit of scientific

inquiry. They are neither comprehensible nor transparent and should be rejected by fans and teams.

For a transparent wins-above-replacement method, we can use openWAR from Baumer, Jensen, and Matthews (2015). Data and programs for this metric are in the public domain.

What about player performance over time? There are truisms in life, and one of those truisms is that the body ages. Much is understood about age effects in baseball and how to model them (Fair 2008). PECOTA, a well known measurement and prediction system, uses player-comparable age curves as its base data.² Sadly, PECOTA is another method lacking in transparency.

² We know about PECOTA from Silver (2004, 2012), who developed the method for a Baseball Prospectus in 2002–2003. Silver (2012) explains that the name comes from “a marginal infielder with the Kansas City Royals during the 1980s who was nevertheless a constant thorn in the side of my favorite Detroit Tigers. . . . Although Bill Pecota hit just .249 for his career overall, he hit .303 in games against the Tigers.” (Silver 2012, 88)

Fortunately, there are alternatives to PECOTA. Teams desiring age-based measures can compute them directly, obtaining predictions about performance over the course of a player’s career (Albert and Bennett 2001; Marchi and Albert 2014).

Some methods build on Bayesian inference (Albert 2009). Age-based models are most easily developed using tractable measures such as proportions.³ Age-based measures and predictions are especially useful in salary negotiations.

³ For an age-based measure, we could work with the proportion of hits in at bats (batting average BA), on-base percentage (OBP), or the proportion of home runs in at bats. We think of observations at time t as probabilities p_t and employ the logit transform, using the log of the odds ratio. The resulting model takes the form of a logistic regression with a quadratic term in the linear predictor:

$$\log\left(\frac{p_t}{1-p_t}\right) = \beta_0 + \beta_1 x_t + \beta_2 x_t^2$$

where x_t is a player's age at time t . With fitted parameters in hand (or posterior distributions for those parameters in a Bayesian context), we work backwards to obtain a player's age curve. We find the value of the performance measure next year $p_{(t+1)}$ associated with the player's age next year $x_{(t+1)}$.

How do we go beyond individual player performance to look at a player's contribution to his team? Therein lies a fundamental question in sports analytics. Anyone who knows sports knows that a good team is worth more than the sum of its parts. And it should come as no surprise that a dysfunctional team is worth less than the sum of its parts. This is to say that team effects should be considered when predicting winners and losers.

Baseball may be less susceptible to team effects than other sports. On a baseball diamond, Tinker-to-Evers-to-Chance works fine, even when Tinkers, Evers, and Chance are not speaking with one another. Baseball is distinct from many other team sports in being defined by many individual matchups, one batter facing one pitcher, then another pitcher facing another batter—the events are discrete and easily identifiable as belonging to one player or another.

In most team sports, players complement one another. Some players are described as “team players,” because they help their teammates play better. Stockton and Malone worked together as a unit on the Utah Jazz. Their classic pick-and-roll made the Jazz a difficult team to beat for many years. Their individual performance measures from those years are inextricably intertwined (Oliver 2004).

By their very nature, basketball and many other team sports present special problems in evaluating individual players. A

player's value on one team may be quite different from his value on another team. This is revealed in the free-agency market where bids for a player vary widely across bidding teams. And this explains why player trades can benefit all teams involved.

Player evaluation has become a hot topic among the consumers of sport due in large measure to the growth in fantasy sports. Fantasy sports has an individual-player focus, with fantasy teams being little more than the sum of their player component parts. In this regard, fantasy sports are pure fantasy. They do not reflect the way players work together as a team.

Performance measures can be quite useful to coaches, managers, and owners. Measures of athleticism and sport proficiency are well documented in Martin (2016), with summary principles outlined in Martin and Miller (in press).

For references on measurement reliability and validity, we refer to the literature of psychometrics (Gulliksen 1950; Cronbach 1951; Ghiselli 1964; Nunnally 1967; Nunnally and Bernstein 1994; Lord and Novick 1968; Fiske 1971; Brown 1976). Betz and Weiss (2001) and Allen and Yen (2002) introduce concepts of measurement theory. Item response theory is discussed by Rogers, Swaminathan, and Hambleton (1991). Articles in the volume edited by Shrout and Fiske (1995) provide many examples of multitraitmultimethod matrices and review quantitative methods available for the analysis of such matrices. Lumley (2010) discusses sample survey design and analysis in the R programming environment. For R functions in psychometrics, see Revelle (2014). Reviews of the weak measurements versus strong

statistics controversy and its relevance (or lack of relevance) to science and statistical inference have been presented by Baker, Hardyck, and Petrinvich (1966) and Velleman and Wilkinson (1993).

Measurement theory applies equally well to text measures and designed surveys. We work with term frequencies within documents and term frequencies adjusted by overall corpus term frequencies. We assign scores to text messages and posts. We utilize methods of natural language processing to detect features in text and to annotate documents (Bird, Klein, and Loper 2009; Pustejovsky and Stubbs 2013). These are measurements.

3. Ranking Teams

Gillon: "So you saying that our boxing here in Diggstown is not to your satisfaction, Mister . . . ?"

Caine: "Caine. Gabriel Caine?"

Gillon: "John Gillon. Nice to meet you."

Caine: "Hey, can I be frank with you?"

Gillon: "Please."

Caine: "It's never too satisfying knowing who's gonna win every time. You know what I mean? Take this mamluke in the white trunks over here. Half way through the first fight, I knew he'll be kissing canvas. And varoom, he's already done it twice. So what do you think? Is he going to kiss canvas a third time? Yes."

Gillon: "What you're saying is that you think this man in the red trunks is going to win this fight."

Caine: "Is there like an acoustical problem in here? I didn't say I think he's going to win this fight. I said I know he's going to win this fight. Hey, I gotta split. By the way, I'd bet a thousand on it."

Gillon: "But would you bet two thousand bucks on it?"

Cain: "What? Are you kidding?"

Gillon: "There's two things we never joke about in Diggstown, Mr. Caine: our boxing and our betting."

—BRUCE DERN AS GILLON AND JAMES WOODS AS CAINE IN
Diggstown (1992)

The meaning of a measure is defined by its relationships with

other measures, and the strength of a team is defined in relation to other teams. When picking winning teams, it is not sufficient to consider individual player statistics. We must see how teams compete with one another.

Measurement is the assignment of numbers to attributes according to rules. And just as we assign numbers to player performance attributes, we can assign numbers to teams. When working with team performance, we usually generate ratings and rankings.

There is debate about how to rank teams, especially when teams have limited opportunity to play one another. Most of us remember the extensive controversies surrounding the Bowl Championship Series (BCS) for college football prior to the introduction of a limited playoff program. And there remains controversy about which teams should qualify for the playoffs due to strength-of-schedule differences across teams.

Compared with college athletics, professional team sports have more balance across league schedules, but perfect balance does not exist. Divisions and conferences are not equal in player abilities, and teams play more of their games with other teams in their own conferences and divisions.

Consider the problem of assessing team strength in the National Basketball Association. Table 3.1 shows the NBA team win/loss records at the close of the regular season 2014–2015. Eight teams from each conference make the playoffs. Western Conference teams are at a disadvantage because there are more strong teams in that conference. NBA schedules are not balanced.

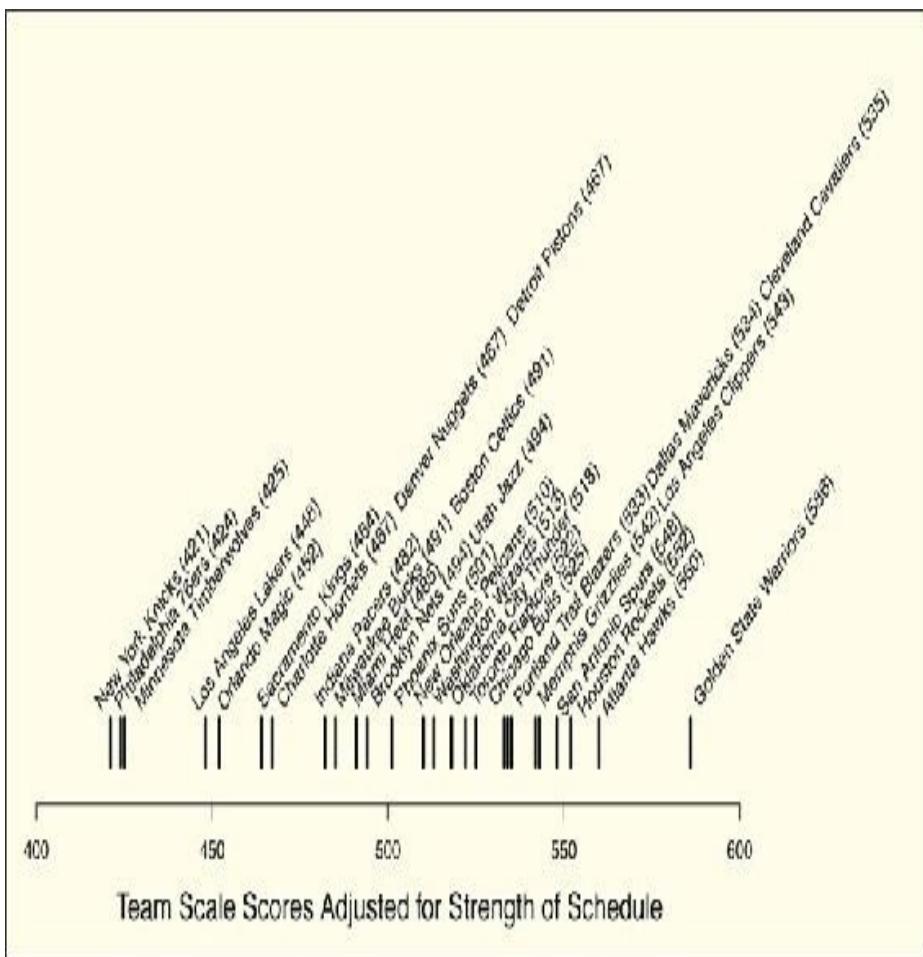
Conference	Division	Team Name (Playoff *)	Abbreviation	Wins	Losses	P(Win)
Eastern	Atlantic	Boston Celtics*	BOS	40	42	0.488
		Brooklyn Nets*	BKN	38	44	0.463
		New York Knicks	NYK	17	65	0.207
		Philadelphia 76ers	PHI	18	64	0.220
		Toronto Raptors*	TOR	49	33	0.598
	Central	Chicago Bulls*	CHI	50	32	0.610
		Cleveland Cavaliers*	CLE	53	29	0.646
		Detroit Pistons	DET	32	50	0.390
		Indiana Pacers	IND	38	44	0.463
		Milwaukee Bucks*	MIL	41	41	0.500
Western	Southeast	Atlanta Hawks*	ATL	60	22	0.732
		Charlotte Hornets	CHA	33	49	0.402
		Miami Heat	MIA	37	45	0.451
		Orlando Magic	ORL	25	57	0.305
		Washington Wizards*	WAS	46	36	0.561
	Southwest	Dallas Mavericks*	DAL	50	32	0.610
		Houston Rockets*	HOU	56	26	0.683
		Memphis Grizzlies*	MEM	55	27	0.671
		New Orleans Pelicans*	NOP	45	37	0.549
		San Antonio Spurs*	SAS	55	27	0.671
Western	Northwest	Denver Nuggets	DEN	30	52	0.366
		Minnesota Timberwolves	MIN	16	66	0.195
		Oklahoma City Thunder	OKC	45	37	0.549
		Portland Trail Blazers*	POR	51	31	0.622
		Utah Jazz	UTA	38	44	0.463
	Pacific	Golden State Warriors*	GSW	67	15	0.817
		Los Angeles Clippers*	LAC	56	26	0.683
		Los Angeles Lakers	LAL	21	61	0.256
		Phoenix Suns	PHX	39	43	0.476
		Sacramento Kings	SAC	29	53	0.354

Table 3.1. NBA Team Records (2014–2015 Season)

How shall we judge or rank teams, given imbalance in team schedules? One way employs unidimensional scaling from psychometrics, which uses team win/loss data from games. Win/loss data are paired comparisons, one team over another in each game.

To employ unidimensional scaling, we construct a matrix with the number of times each team beats each other team. This matrix is converted to a matrix of proportions, which are then averaged across rows or columns and referred to a standard normal distribution. As a final step in the process, we set a desired mean and standard deviation for scale scores. The method automatically adjusts for strength of schedule while defining interval-level measures from what were originally paired comparisons.

Figure 3.1 shows the results of a unidimensional scaling of box score data from the 2014–2015 NBA season. A team of average strength would have a score of 500 on this scale, with the standard deviation across teams set to 100. Unidimensional scales reflect differences between teams in strength, not just rank orders of teams. Note the strength of the Golden State Warriors relative to other teams. From the unidimensional scaling alone, we would have predicted the Golden State Warriors to be victorious in the playoffs.



**Figure 3.1. Assessing Team Strength: NBA Regular Season
(2014–2015)**

Another method for rating teams based on wins and losses comes from work on chess player rankings by Árpád Élö (1903–1992). The Elo system has been adapted to professional team sports in the United States and is the basis of Jeff Sagarin’s rankings in *USA Today*.

In ranking sports teams, we can go beyond binary win/loss data to consider the margin of victory or game point differentials. Again we must account for variability in team schedules. When looking at points scored in games, we must also adjust for games involving extra innings in baseball or overtime periods in football or basketball.

Another consideration when working with scores relates to very high scoring games. These may result from games involving teams that are unevenly matched (“blowouts”) or games that have little bearing on team possibilities for moving into the playoffs. In games such as these, teams may be using second string or bench players in place of their usual lineups. Some adjustment of game scores would be appropriate in such instances.

Other approaches to team rankings involve developing separate measures or ranks for offensive and defensive performance, which are later combined into a single measure or rank. Sometimes it makes sense to employ a multivariate approach, accounting for team performance on many measures and/or across many player positions. Strength profiles for teams may be represented in parallel coordinate plots.

Teams want to build future winners. General managers wonder about the characteristics that make winning more likely.

Teams can improve through player drafts, free agent acquisitions, promotion of players from minor leagues, and player trades. And player performance measures evaluated in the context of current team performance measures can point to the player additions that are most beneficial to a team. A general manager can assess a team’s ranking with and without the addition of new players.

Player selection problems have multiple constraints. There are constraints on team size and composition. There is often a salary cap constraint. The decision variables are binary—each available player is either selected for the team or not. Problems such as this are known as multidimensional knapsack problems (MKP) and have been studied extensively in

operations research (Fréville 2004; Kellerer, Pferschy, and Pisinger 2004). Guttag (2013) provides an introduction to knapsack problems using Python examples. See the overview of mathematical programming presented in appendix A (page 200).

Having determined player rosters, team managers and coaches must decide on player starting lineups for each game. Again, we can provide suggestions using methods of constrained optimization such as mathematical programming. With the extensive data that are available today, it is possible to build credible models for improving the performance of teams.

Sports team rankings and models for predicting sporting event outcomes have garnered considerable attention from mathematicians as well as sports enthusiasts. Langville and Meyer (2012) provide a comprehensive review of rating and ranking algorithms and methods. Ranking methods, the use of rankings in the design of sporting tournaments, and the seeding of teams in tournaments have been discussed extensively (Thompson 1975; Groeneveld 1990; Appleton 1995; Carlin 1996; West 2006). There has been special interest in National Football League games (Thompson 1975; Stern 1991; Glickman and Stern 1998). And much has been written about rating and ranking strategies for picking winning brackets in the men's Division One NCAA basketball tournament, known as *March Madness* (Schwertman, McCready, and Howard 1991; Schwertman, Schenk, and Holbrook 1996; Carlin 1996; Kaplan and Garstka 2001; West 2006).

Unidimensional scaling and the method of paired comparisons have long histories in measurement theory and statistics.

Paired comparisons have been used in taste-testing experiments, psychophysical investigations, preference scaling, and studies of consistency across judges (inter-rater reliability), as well as in ranking players and teams. The extensive literature in this field, documented in Davidson and Farquhar (1976), dates back to early work by Thurstone (1927) and Guilford (1936), who used normal distribution theory to obtain scale scores for underlying psychological dimensions. In the statistical literature, Bradley and Terry (1952) introduced traditional logistic or logit models for analyzing paired comparisons. Reviews of paired comparison methods have been provided by Torgerson (1958), David (1963), and Bradley (1976).

Exhibit 3.1 shows an R program for unidimensional scaling of box scores from the 2014–2015 NBA season. These provided the basis for constructing matrices of wins and proportions of wins across the season. The matrix of proportions is used to compute average schedule-adjusted proportions, which are then referred to a standard normal distribution. As a final step in the process, we set the desired mean and standard deviation for univariate scale scores. The program employs standard R graphics to plot results from this unidimensional scaling. Included in the program are matrices for performing other types of team rankings, including those that take account of runs scored by winning and losing teams.

Exhibit 3.1. Assessing Team Strength by Unidimensional Scaling (R)

[Click here to view code image](#)

```
# Assessing Team Strength by Unidimensional
Scaling (R)

# NBA Regular Season 2014-2015 data from
Basketball-Reference.com

# read comma-delimited text file to create
data frame
nba_input <-
read.csv("basketball_2014_2015_season.csv",
stringsAsFactors = FALSE)

# add record number to records
record_number <- seq(nrow(nba_input))
nba_scores <- cbind(data.frame(record_number),
nba_input)

# explore the data
with(nba_scores, plot(home_points,
visitor_points))
with(nba_scores, table(overtime))

# define minutes played using overtime
information
nba_scores$minutes <- rep(48, length =
nrow(nba_scores)) # standard game
for (i in seq(along = nba_scores$minutes)) {
  if(nba_scores$overtime[i] == "OT")
    nba_scores$minutes[i] <-
nba_scores$minutes[i] + 5 # 1 OT period
  if(nba_scores$overtime[i] == "2OT")
    nba_scores$minutes[i] <-
nba_scores$minutes[i] + 10 # 2 OT periods
  if(nba_scores$overtime[i] == "3OT")
    nba_scores$minutes[i] <-
nba_scores$minutes[i] + 15 # 3 OT periods
}
with(nba_scores, table(overtime, minutes)) # check minute calculations
```

```

# compute points per minute for each team
nba_scores$visitor_ppm <-
  nba_scores$visitor_points / nba_scores$minutes
nba_scores$home_ppm <- nba_scores$home_points
/ nba_scores$minutes

# check calculations
print(head(nba_scores))
print(tail(nba_scores))

# explore points-per-minute data
with(nba_scores, plot(home_ppm, visitor_ppm))

# read in team names and abbreviations
team_info <-
  read.csv("nba_team_names_abbreviations.csv",
    stringsAsFactors = FALSE)

# append team information to nba_scores data
# frame
list_of_team_names <- team_info$team_name
# first do the visitor team
nba_scores$visitor_conference <- rep("", length =
  nrow(nba_scores))
nba_scores$visitor_division <- rep("", length =
  nrow(nba_scores))
nba_scores$visitor_code <- rep("", length =
  nrow(nba_scores))
for (i in seq(along = list_of_team_names)) {
  this_team_info <-
    team_info[(team_info$team_name ==
      list_of_team_names[i]),]
  indices_for_visitor_team <-
    which (nba_scores$visitor_team ==
      list_of_team_names[i])
  for (j in seq(along =
    indices_for_visitor_team)) {
    nba_scores$visitor_conference[indices_for_v
    <-
      this_team_info$conference[1]
}

```

```

nba_scores$visitor_division[indices_for_vis
<-
  this_team_info$division[1]
nba_scores$visitor_code[indices_for_visitor
<-
  this_team_info$abbreviation[1]
}
}

# next do the home team
nba_scores$home_conference <- rep("", length =
nrow(nba_scores))
nba_scores$home_division <- rep("", length =
nrow(nba_scores))
nba_scores$home_code <- rep("", length =
nrow(nba_scores))
for (i in seq(along = list_of_team_names)) {
  this_team_info <-
    team_info[(team_info$team_name ==
list_of_team_names[i]),]
  indices_for_home_team <-
    which (nba_scores$home_team ==
list_of_team_names[i])
  for (j in seq(along =
indices_for_home_team)) {
    nba_scores$home_conference[indices_for_home
<-
      this_team_info$conference[1]
    nba_scores$home_division[indices_for_home_t
<-
      this_team_info$division[1]
    nba_scores$home_code[indices_for_home_team[
<-
      this_team_info$abbreviation[1]
    }
  }

# define winning team (Visitor or Home)
nba_scores$win_visitor_home <- rep("Home",
length = nrow(nba_scores))
for (i in seq(along =

```

```
nba_scores$record_number))
  if (nba_scores$visitor_ppm[i] >
nba_scores$home_ppm[i])
    nba_scores$win_visitor_home[i] <-
"Visitor"

# who wins more games.. visitor or home team
with(nba_scores, table(win_visitor_home))
with(nba_scores, plot(home_ppm, visitor_ppm))

# define winning and losing teams by
three-character abbreviation/code
nba_scores$win_team_code <- rep("", length =
nrow(nba_scores))
nba_scores$lose_team_code <- rep("", length =
nrow(nba_scores))
for (i in seq(along =
nba_scores$record_number)) {
  if (nba_scores$visitor_ppm[i] >
nba_scores$home_ppm[i]) {
    nba_scores$win_team_code[i] <-
nba_scores$visitor_code[i]
    nba_scores$lose_team_code[i] <-
nba_scores$home_code[i]
  }
  if (nba_scores$visitor_ppm[i] <
nba_scores$home_ppm[i]) {
    nba_scores$win_team_code[i] <-
nba_scores$home_code[i]
    nba_scores$lose_team_code[i] <-
nba_scores$visitor_code[i]
  }
}

# check the nba_scores data frame
print(head(nba_scores))
print(tail(nba_scores))

# create OKC data frame for testing and
visualization example
```

```

OKC_visitor <-
nba_scores[(nba_scores$visitor_code ==
"OKC"),]
OKC_home <- nba_scores[(nba_scores$home_code
== "OKC"),]
OKC_data <- rbind(OKC_visitor, OKC_home)
OKC_data <-
OKC_data[sort.list(OKC_data$record_number),]
# save file for Oklahoma City Thunder for data
visualization work
write.csv(OKC_data, file =
"okc_data_2014_2015.csv", row.names = FALSE)

# write win team score and lose team score in
points per minute
nba_scores$win_team_ppm <- rep(0, length =
nrow(nba_scores))
nba_scores$lose_team_ppm <- rep(0, length =
nrow(nba_scores))
for (i in seq(along =
nba_scores$record_number)) {
  if (nba_scores$visitor_ppm[i] >
nba_scores$home_ppm[i]) {
    nba_scores$win_team_ppm[i] <-
nba_scores$visitor_ppm[i]
    nba_scores$lose_team_ppm[i] <-
nba_scores$home_ppm[i]
  }
  if (nba_scores$visitor_ppm[i] <
nba_scores$home_ppm[i]) {
    nba_scores$win_team_ppm[i] <-
nba_scores$home_ppm[i]
    nba_scores$lose_team_ppm[i] <-
nba_scores$visitor_ppm[i]
  }
}

# compute margin of victory in points per
minute
nba_scores$margin_ppm <-

```

```

nba_scores$win_team_ppm -
nba_scores$lose_team_ppm

# initialize matrices for entire set of 30
# teams
ordered_team_codes <-
sort(team_info$abbreviation)
# total number of wins
wins_mat <- matrix(0, nrow = 30, ncol = 30,
dimnames = list(ordered_team_codes,
ordered_team_codes))
prop_mat <- wins_mat # proportion wins
ppm_mat <- wins_mat # sum of ppm scores

# build matrices for entire set of 30 teams
for (i in seq(along =
nba_scores$record_number)) {
    # tally the number of times team k beats
    # team j
    wins_mat[nba_scores$lose_team_code[i],
nba_scores$win_team_code[i]] <-
    wins_mat[nba_scores$lose_team_code[i],
nba_scores$win_team_code[i]] + 1
    # sum the ppm scores for teams and enter in
    ppm_mat
    # two entries needed for each record in
    nba_scores
    ppm_mat[nba_scores$lose_team_code[i],
nba_scores$win_team_code[i]] <-
    ppm_mat[nba_scores$lose_team_code[i],
nba_scores$win_team_code[i]] +
    nba_scores$win_team_ppm[i]
    ppm_mat[nba_scores$win_team_code[i],
nba_scores$lose_team_code[i]] <-
    ppm_mat[nba_scores$win_team_code[i],
nba_scores$lose_team_code[i]] +
    nba_scores$lose_team_ppm[i]
}

# check wins_mat entries... should be 82 games

```

```

for every team
for (j in 1:length(ordered_team_codes))
  print(sum(wins_mat[,j]) +
sum(wins_mat[j,]))

# compute matrix of proportions prop_mat
# proportion of times team j beats team k
for(j in 1:length(ordered_team_codes)) { # begin outer for-loop
  for(k in 1:length(ordered_team_codes)) { # begin inner for-loop
    if (j == k) prop_mat[j,k] <- NA      # set diagonal entries missing
    if (j > k) { # begin outer if-block between_team_games <-
      wins_mat[ordered_team_codes[j],ord +
      wins_mat[ordered_team_codes[k],ord

      # if teams never play others within the season
      # we set the entry in the cell
      # to be the mean of the team
      if (between_team_games == 0) { # begin first inner if-block
        prop_mat[j,k] <-
sum(wins_mat[j,]) /
          (sum(wins_mat[,j]) +
sum(wins_mat[j,]))
        prop_mat[k,j] <-
sum(wins_mat[k,]) /
          (sum(wins_mat[,k]) +
sum(wins_mat[k,]))
      } # end first inner if-block

      # when teams play other teams at least once
      # we compute the proportion of times they beat one another
      if (between_team_games > 0) { #

```

```

begin second inner if-block
    prop_mat[j,k] <-
        wins_mat[ordered_team_codes[j]
                  between_team_games
    prop_mat[k,j] <-
        wins_mat[ordered_team_codes[k]
                  between_team_games
    } # end second inner if-block
} # end outer if-block
} # end inner for-loop
} # end outer for-loop

# check prop_mat against known results for the
season
# results should be close but not exact
# because prop_mat adjusts for strength of
schedule
for(i in seq(along = ordered_team_codes))
    cat("\n", ordered_team_codes[i], ":" ,
mean(prop_mat[,i], na.rm = TRUE))
# unidimensional scaling work begins here
# begin by defining the paired_comparisons
scaling matrix
pc_mat <- prop_mat
for(j in 1:length(ordered_team_codes))
    for(k in 1:length(ordered_team_codes))
        if (j == k) pc_mat[j,k] <- 0.50    #
set diagonal entries missing

nobjects <- length(ordered_team_codes)

mean_pc_mat <- numeric(nobjects)
object_quantile <- numeric(nobjects)
for(j in 1:nobjects)
{
    mean_pc_mat[j] <- mean(pc_mat[,j])
    object_quantile[j] <- qnorm(mean_pc_mat[j])
}

# user-defined function provides mean 500

```

```

standard deviation 100
z.score.converter <- function(z)
{round(((100*z) + 500),digits=0)}

scale_score <- numeric(nobjects)
for(j in 1:nobjects)
  scale_score[j] <-
z.score.converter(object_quantile[j])

sorted_team_info <-
team_info[sort.list(team_info$abbreviation),]
sorted_team_info$scale_score <- scale_score

sorted_team_info$name_with_score <- rep("", 
length = nrow(sorted_team_info))
for(i in seq(along = ordered_team_codes))
sorted_team_info$name_with_score[i] <-
  paste(sorted_team_info$team_name[i],
        " (",
        sorted_team_info$scale_score[i], ")",
        sep =
      "")

scaling_frame <-
  sorted_team_info[,c("abbreviation",
"team_name",
        "name_with_score", "scale_score")]
names(scaling_frame) <-
  c("object.name", "long.object.name",
    "long.object.name.with.score",
  "scale_score")

ordered_scaling_frame <-
  scaling_frame[sort.list(scale_score,
decreasing=FALSE),]

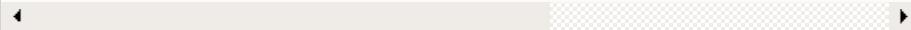
# plotting to external pdf file using standard
R graphics
pdf(file =
"fig_unidimensional_scaling_analysis.pdf",
width = 11, height = 8.5)

```

```
text.angle = 45
par(mfrow=c(1,1), xpd=NA, cex=1, yaxt= "n"
, lwd=3, bty="n",
srt=text.angle, mar=c(5, 0, 4, 0) + 0.1)
plot(ordered_scaling_frame$scale_score,
rep(0.1,length(ordered_scaling_frame$scale_sco
type="h",
xlab=paste("Team Scale Scores Adjusted for
Strength of Schedule"),
ylab="",
ylim=c(0,2),
xlim=c(min(ordered_scaling_frame$scale_score)
- 50,
max(ordered_scaling_frame$scale_score) +
50))

text(ordered_scaling_frame$scale_score + 40,
rep(0.65,length(ordered_scaling_frame$scale_sc
ordered_scaling_frame$long.object.name.with.sc
pos=2, cex=1)

dev.off()
```



4. Predicting Scores

“You know what the difference between hitting .250 and .300 is? It’s 25 hits. Twenty-five hits in 500 at-bats is 50 points. OK? There’s six months in a season. That’s about 25 weeks. That means if you get one extra flare a week. Just one. A gork. You get a ground ball with eyes. You get a dying quail. Just one more dying quail a week, and you go to Yankee Stadium.”

—KEVIN COSTNER AS CRASH DAVIS IN *Bull Durham* (1988)

Sporting events take place in a public arena. We know the managers and players. We know their salaries, and we have extensive information about their performance. We know who wins, and we know who loses. The data are available and easily accessible. These data, both text and numbers, come to us in time and place. And the analysis of these data may be carried out using methods well known to data scientists.

Competitive advantage—the extra inches, hits, and free throws—can be gained through the use of predictive models. Just as we can predict which customers will buy products or which stocks will outperform the market, we can predict which players and teams will succeed on the fields and courts of play. Just as easily as we can guide the decisions of manufacturers and retailers, we can guide the decisions of coaches and general managers in sports.

By analyzing sports data, we may learn that bunting to advance a runner in baseball or punting on fourth down in football are not good ideas. We may discover that the most valuable basketball player is not the one who scores the most

points or makes the highest percentage of three-point shots, but the one who helps his teammates score. Through analytics, we discover many small findings of interest or curiosity in sports. But more importantly, we gain insight into sports business management and strategy.

Just as we can view sports in a larger context as entertainment, we can view sports analytics within the larger context of data science. Information technology professionals deliver data as data. Statisticians and machine learning experts focus on models. Data scientists, building on a foundation of data and models, tell a story that others can understand. They are concerned with prediction, testing, and interpreting model results, and if the data and models warrant, recommending management action. The work of data science, illustrated in figure 4.1, involves learning from data and models and helping managers make informed decisions. We begin with a question. We end with a story, a report to management.

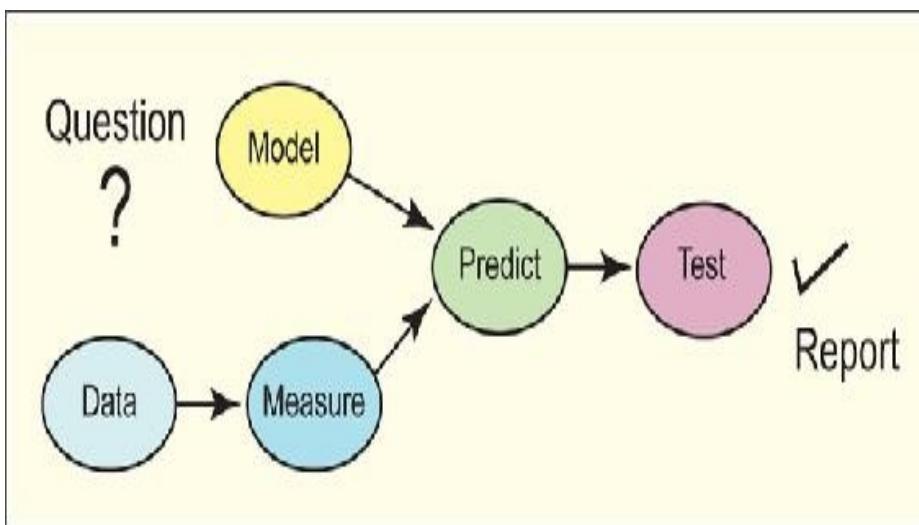


Figure 4.1. Work of Data Science

Predictive models come in two general forms, referred to as classification and regression. Classification concerns

predicting a future class or category. Will the consumer of sports buy a ticket to the game? If she buys a ticket to the game, will it be a ticket for a standard, preferred, or a box seat. Perhaps she becomes a loyal fan of the team. If so, will she buy a season ticket for next year?

With classification, we predict consumer choices or discrete responses. With regression, on the other hand, we predict a response with meaningful magnitude. How many dollars will be spent on tickets? How much will be spent on concessions? How many tickets will be sold to the game? What will be in the number of runs or points scored by the home team? What will be the right fielder's batting average next year? How many yards will the fullback run next week? What kind of salary will a player receive when he becomes a free agent?

A model is a representation of things, a rendering or description of reality. A typical model in data science is an attempt to relate one set of variables to another. Limited, imprecise, but useful, a model helps us to make sense of the world. A model is more than just talk because it is based on data.

We can think of data science as the new statistics, a blending of modeling techniques, information technology, and business savvy. Data science apprehends a data-intensive world, requiring a multidisciplinary skill set essential for success in business, nonprofit organizations, and government. Data science is predictive analytics. Whether forecasting sales or market shares, finding a good retail site or investment opportunity, identifying consumer segments and target markets, or assessing the potential of new products or risks for existing products, modeling methods in predictive analytics

provide the key.

Data scientists speak the language of business—accounting, finance, marketing, and management. They know about information technology, including data structures, algorithms, and object-oriented programming. They understand statistical modeling, machine learning, and mathematical programming. Data scientists are methodological eclectics, drawing from many scientific disciplines and translating the results of empirical research into words and pictures that management can understand.

Data science, as with much of statistics, involves searching for meaningful relationships among variables and representing those relationships in models. There are response variables—things we are trying to predict. There are explanatory variables or predictors—things that we observe, manipulate, or control and might relate to the response.

Prediction problems are defined by their width or number of potential predictors and by their depth or number of observations in the data set. It is the number of potential predictors in business, marketing, and investment analysis that causes the most difficulty. There can be thousands of potential predictors with weak relationships to the response. With the aid of computers, hundreds or thousands of models can be fit to subsets of the data and tested on other subsets of the data, providing an evaluation of each predictor. Predictive modeling involves finding good subsets of predictors. Models that fit the data well are better than models that fit the data poorly. Simple models are better than complex models.

Consider three general approaches to research and modeling: traditional, data-adaptive, and model-dependent. Figure 4.2

illustrates the approaches. With a traditional approach, statistical inference and modeling begin with the specification of a theory or model. We use methods such as linear regression and logistic regression to estimate parameters for linear predictors. Classical or Bayesian methods of statistical inference may be employed. We fit models to data and check models with diagnostics. We validate models before using them to make predictions.

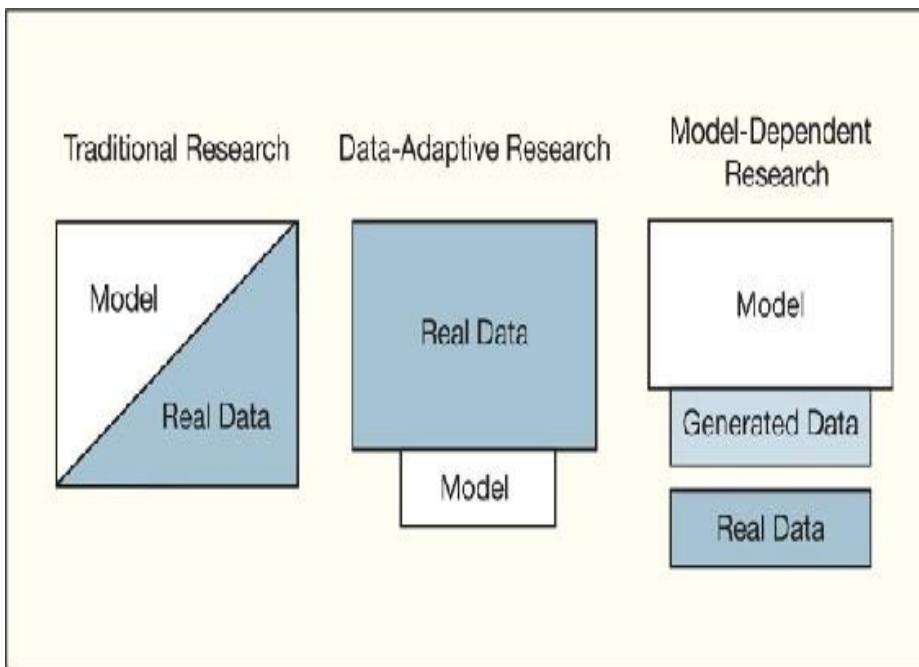


Figure 4.2. Data and Models for Research

When we employ a data-adaptive approach, we begin with data and search through those data to find useful predictors. We give little thought to theories or hypotheses prior to running the analysis. This is the world of machine learning, sometimes called statistical learning or data mining. Data-adaptive methods adapt to the available data, representing nonlinear relationships and interactions among variables. The data determine the model. Data-adaptive methods are data-driven. As with traditional models, we validate data-adaptive

models before using them to make predictions.

Model-dependent research is the third approach. It begins with the specification of a model and uses that model to generate data, predictions, or recommendations. Simulations and mathematical programming methods, primary tools of operations research, are examples of model-dependent research. When employing a model-dependent or simulation approach, models are improved by comparing generated data with real data. We ask whether simulated consumers, firms, and markets behave like real consumers, firms, and markets. The comparison with real data serves as a form of validation.

Data may be organized by observational unit, time, and space. The observational or cross-sectional unit could be an individual consumer or business or any other basis for collecting and grouping data. Data are organized in time by seconds, minutes, hours, days, and so on. Space or location is often defined by longitude and latitude. As we consider sports analytics problems in this book, we touch on many types of models, including cross-sectional, time series, and spatial data models.

Whatever the structure of the data and associated models, prediction is the unifying theme. We use the data we have to predict data we do not yet have, recognizing that prediction is a precarious enterprise. It is the process of extrapolating and forecasting. And model validation is essential to the process.

How do we test the accuracy of a predictive model? We employ a training-and-test regimen. As in economic and financial research, we see how well we can predict the past before we attempt to predict the future. That is, we use more distant past observations to predict less distant past

observations. If we are working with a sixteen-week football season, for example, we use the first ten weeks' data to predict what will happen in the eleventh week.

Then we use the first eleven weeks' data to predict what will happen in the twelfth week, and so on. Measures of uncertainty in predicting the past help us to assess uncertainty about predicting the future.

To make predictions, we may employ classical or Bayesian methods. Or we may dispense with traditional statistics entirely and rely on machine learning algorithms. We do what works. Our approach to data science is based on a simple premise:

The value of a model lies in the quality of its predictions.

Indices such as the Akaike information criterion (AIC) or the Bayes information criterion (BIC) help us to judge one model against another, providing a balance between goodness-of-fit and parsimony. Central to our approach is a *training-and-test regimen*. We partition sample data into training and test sets. We build our model on the training set and evaluate it on the test set. Simple two-and three-way data partitioning are shown in figure 4.3.

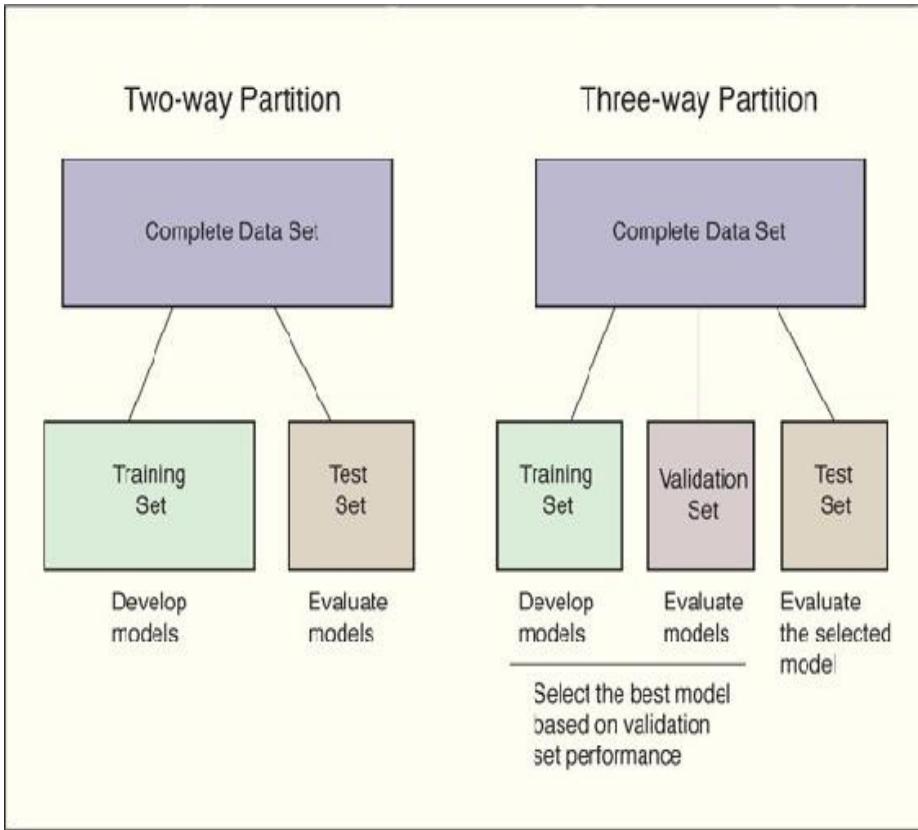


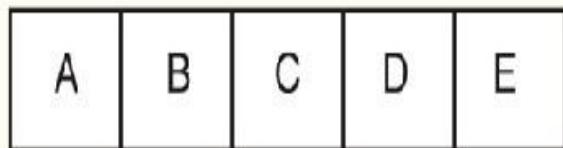
Figure 4.3. Training-and-Test Regimen for Model Evaluation

A random splitting of a sample into training and test sets could be fortuitous, especially when working with small data sets, so we sometimes conduct statistical experiments by executing a number of random splits and averaging performance indices from the resulting test sets. There are extensions to and variations on the training-and-test theme.

One variation on the training-and-test theme is multi-fold cross-validation, illustrated in figure 4.4. We partition the sample data into M folds of approximately equal size and conduct a series of tests. For the five-fold cross-validation shown in the figure, we would first train on sets B through E and test on set A . Then we would train on sets A and C through E , and test on B . We continue until each of the five folds has

been utilized as a test set. We assess performance by averaging across the test sets. In leave-one-out cross-validation, the logical extreme of multi-fold cross-validation, there are as many test sets as there are observations in the sample.

Randomly divide the sample into folds of approximately equal size:



Each fold serves once as a test fold:

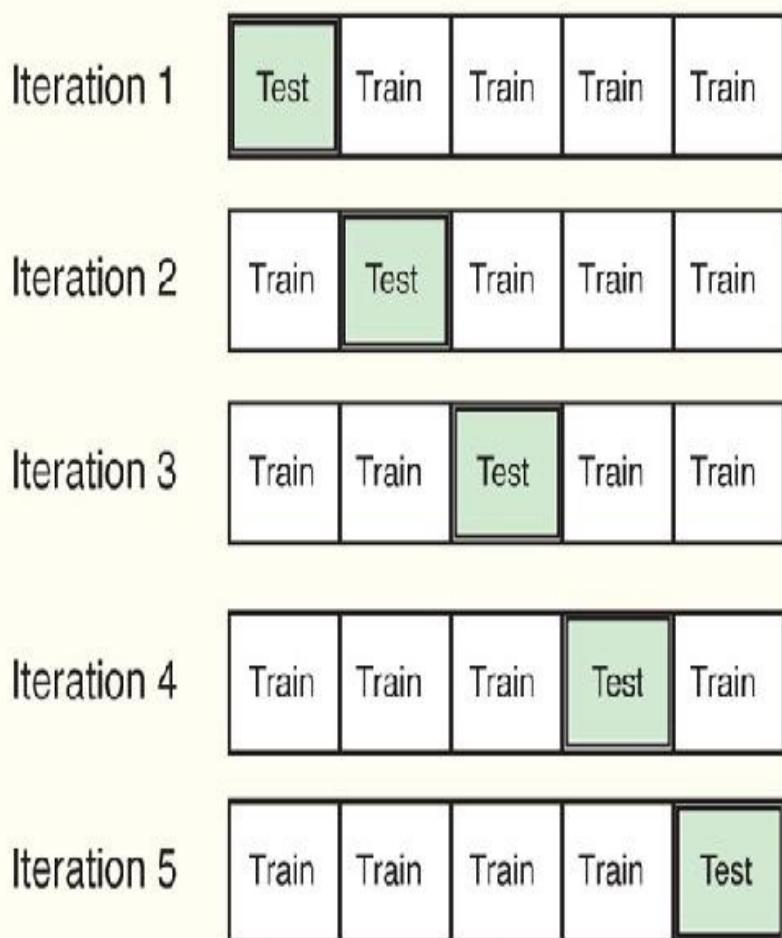


Figure 4.4. Training-and-Test Using Multi-fold Cross-validation

Another variation on the training-and-test regimen is the class of bootstrap methods. If a sample approximates the population from which it was drawn, then a sample from the sample (what is known as a resample) also approximates the population. A bootstrap procedure, as illustrated in figure 4.5, involves repeated resampling with replacement. We take many random samples with replacement from the sample, and for each of these resamples, we compute a statistic of interest. The bootstrap distribution of the statistic approximates the sampling distribution of that statistic.

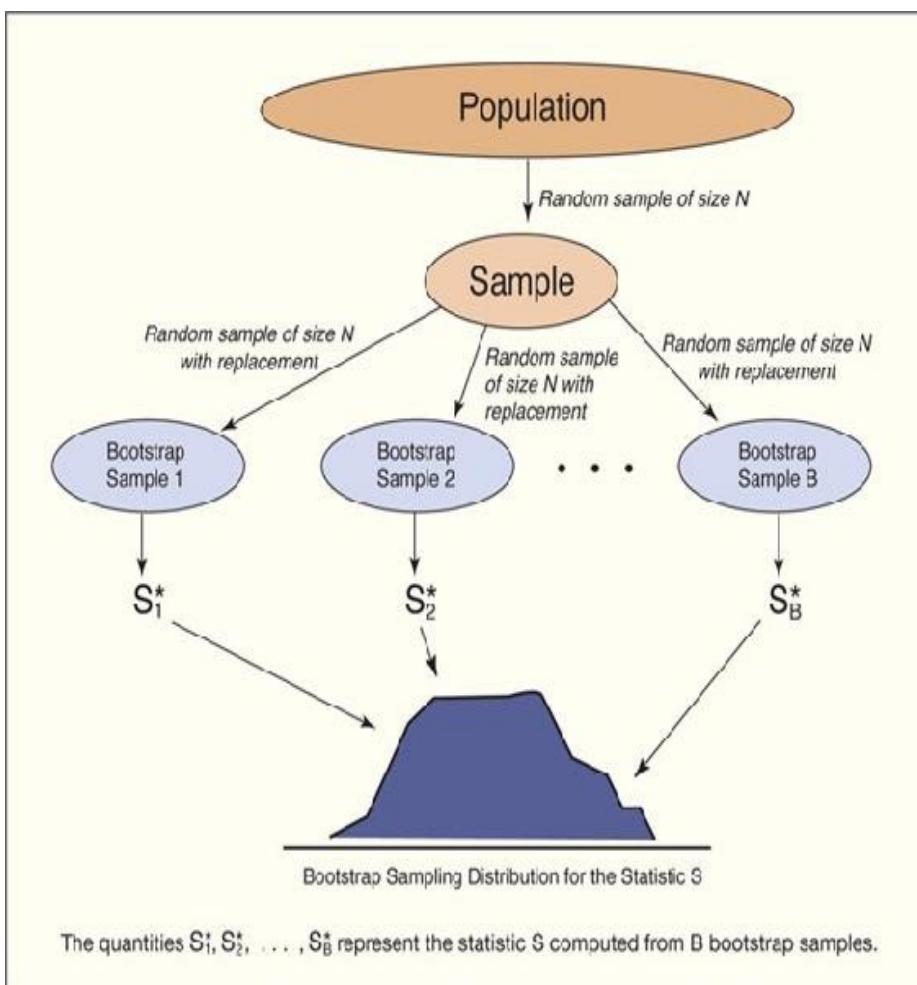


Figure 4.5. Training-and-Test with Bootstrap Resampling

What is the value of the bootstrap? It frees us from having to

make assumptions about the population distribution. We can estimate standard errors and make probability statements working from the sample data alone. The bootstrap may also be employed to improve estimates of prediction error within a leave-one-out cross-validation process. Cross-validation and bootstrap methods are reviewed in Davison and Hinkley (1997), Efron and Tibshirani (1993), and Hastie, Tibshirani, and Friedman (2009).

Much of what passes as sports analytics is performance measurement alone. In baseball, as we have discussed, there is debate about which is the best measure of hitting prowess: batting average, on-base percentage, slugging percentage, or some combination of these. But when we need to make a prediction, we can dispense with arguments about measurement validity and use the combination of measures that provides the best prediction.

Building a predictive model is a matter of finding the right combination of measures to provide trustworthy predictions. We select measures or explanatory variables. We select the type of modeling method to be employed. And with traditional statistical methods, we specify the mathematical form of models and estimate the weights to apply to measures.

The simplest of models are those that employ a linear predictor, adding one variable effect to another with regression coefficients as weights. More complicated models consider interactions among explanatory variables. These may take a linear form as well, but they are more likely to employ a data-adaptive method, using a machine learning algorithm to extract the model specification from the data themselves.

With hundreds or thousands of measures or explanatory

variables from which to choose, there will be millions of possible model specifications. Selecting the right method and the right model specification—this is the art and science of predictive modeling. There are many modeling techniques from which to choose, including traditional linear regression, tree-structured regression, and neural networks, among others.

Often a researcher will test a number of techniques to see which technique or combination of techniques works best. An approach that works well in many contexts is an ensemble technique that combines many small, simple models into a large, more complex model. For example, we could use tree-structured regression for each of a number of performance measures in baseball, and then average their forecasts to get a summary prediction.

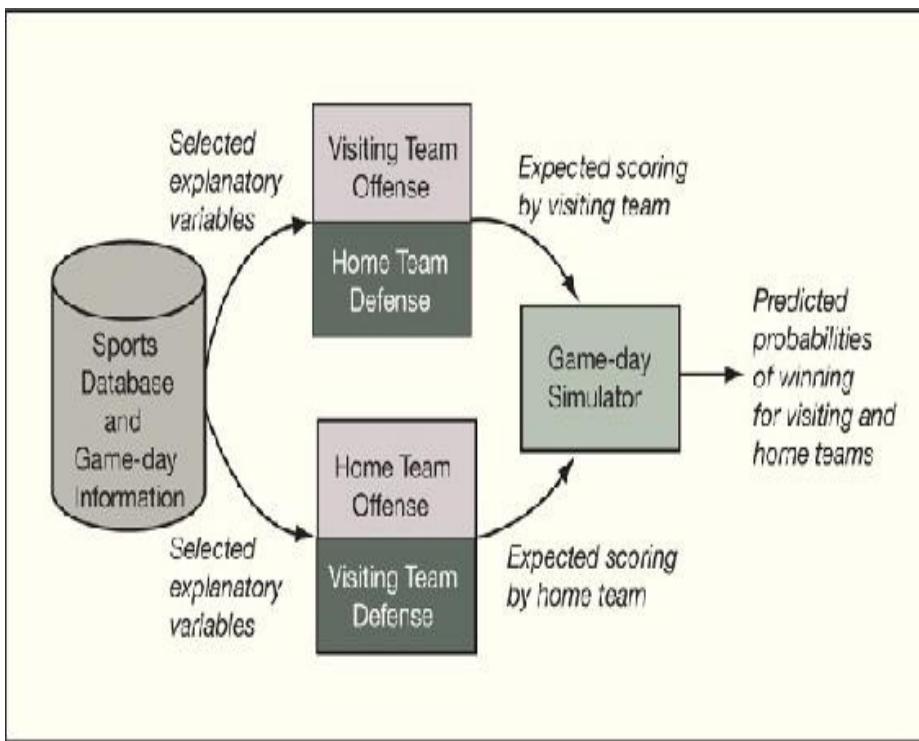
To determine the accuracy, goodness, or utility of a model, we choose a measure of accuracy. We could use the percentage error in prediction, the mean-square error in prediction, or the correlation between observed scores and the predicted scores.

A good choice for measuring the accuracy of a regression model is the squared correlation of observed scores with predicted scores, also known as the coefficient of determination. The coefficient of determination can be interpreted as a proportion. It is the proportion of response variance accounted for by the predictive model. As a proportion, it varies between zero and one in all settings.

When the coefficient of determination is zero, the model is of no value in predicting the future. Perfect predictive power would yield a coefficient of determination equal to one.

Miller (2008) presents a generic, data-driven approach to picking a winning team, as shown in figure 4.6. The modeling

framework follows the principles of predictive inference. Explanatory variables relate to past player and team performance. These are used to predict runs or points scored by opposing teams. We note where and when games are played and other conditions that may affect game outcomes. Starting pitchers, for example, are especially important in predicting outcomes in baseball.



Source. Miller (2008).

Figure 4.6. Predictive Modeling Framework for Team Sports

The framework allows distinct sets of explanatory variables to be used for visiting and home teams. The drivers of success for the Yankees may be different from the drivers of success for the Red Sox, the drivers for the Red Sox different from the Angels, the Angels different from the Dodgers, and so on. For each team, the things that matter when playing at home may be different from the things that matter when playing away.

Various measures of win/loss strength and offensive and defensive performance may be used as explanatory variables. An important part of the modeling framework is game-day simulation, which we illustrate in chapter 10 with an example from Major League Baseball.

One approach to game-day modeling is to predict runs or points scored and runs or points allowed for each team and then use simulation to convert these into binary win/loss predictions. A second approach is to treat the problem as a logistic regression, predicting the win/loss response directly. A third approach is to use multivariate regression to predict visiting and home team runs concurrently.

Think of a game as a multivariate or, more precisely, a bivariate regression problem—we predict two scores simultaneously, scores of home and visiting teams. Then we translate each pair of predicted scores into a binary classification—either the home team wins or the visiting team wins. With predictions of winners in hand, it is a simple matter to see what happens in actual games and compute the proportion of correct predictions, an index of predictive accuracy well understood by managers and sports enthusiasts.

Predictive models can be traditional or data-adaptive, or a combination of the two. As always, a training-and-test regimen is employed in the evaluation of models.

The data of sports and the predictive models that draw on these data can provide competitive advantage. We may see a time when the smartest, best informed, and most technologically advanced teams are the ones that win.

5. Making Game-Day Decisions

Jimmy: "Taking a little day trip?"

Dottie: "No, Bob and I are driving back to Oregon."

Jimmy: "You know, I really thought you were a ball player."

Dottie: "Well, you were wrong."

Jimmy: "Was I?"

Dottie: "Yeah, it is only a game, Jimmy. It's only a game. And I have Bob. I don't need this. I don't."

Jimmy: "I gave away five years at the end of my career to drinking. Five years. And now there isn't anything I wouldn't give to get back one day of it."

Dottie: "Well, we're different."

Jimmy: "Chicken shit, Dottie. If you want to go back to Oregon and make a hundred babies, great. I'm in no position to tell anyone how to live. But sneaking out like this, quitting—you'll regret it the rest of your life. Baseball is what gets inside of you. It's what lights you up. You can't deny that."

Dottie: "It just got too hard."

Jimmy: "It's supposed to be hard. If it wasn't hard, everyone would do it. The hard is what makes it great."

—TOM HANKS AS JIMMY DUGAN AND GEENA DAVIS AS DOTTIE HINSON IN *A League of Their Own* (1992)

To show how data can inform game-day decisions, it is convenient to begin with baseball because baseball may be described as a sequence of discrete events. Play stops and play starts in baseball—there is no continuous clock or time limit.

Among the sports, analytics in baseball is the most advanced, with extensive documentation in the public domain.

Table 5.1 shows the twenty-five states of a baseball half-inning, along with the expected runs to be scored by the time the inning ends. The expected runs in this table are computed from play-by-play records across all regular season games in 2014 ([FanGraphs.com](#) 2015).

State Code	Outs	Runners on Base			Expected Runs
		First	Second	Third	
[0000]	0	0	0	0	0.461
[1000]	1	0	0	0	0.234
[2000]	2	0	0	0	0.095
[0100]	0	1	0	0	0.831
[1100]	1	1	0	0	0.489
[2100]	2	1	0	0	0.214
[0010]	0	0	1	0	1.068
[1010]	1	0	1	0	0.644
[2010]	2	0	1	0	0.305
[0001]	0	0	0	1	1.426
[1001]	1	0	0	1	0.865
[2001]	2	0	0	1	0.413
[0110]	0	1	1	0	1.373
[1110]	1	1	1	0	0.908
[2110]	2	1	1	0	0.343
[0101]	0	1	0	1	1.798
[1101]	1	1	0	1	1.140
[2101]	2	1	0	1	0.471
[0011]	0	0	1	1	1.920
[1011]	1	0	1	1	1.354
[2011]	2	0	1	1	0.570
[0111]	0	1	1	1	2.282
[1111]	1	1	1	1	1.520
[2111]	2	1	1	1	0.736
[END]					0.000

Expected runs estimates from FanGraphs.com (2015).

Table 5.1. Twenty-five States of a Baseball Half-Inning

Every baseball half-inning is a sequence of events or transitions from one state to another. It begins with no outs and no one on base (state code [0000]). It ends with three outs, referred to as an absorbing state or sink (state code [END]). There are also probabilities associated with going from one state to another—these are called transition probabilities in the language of **Markov chains**, and they can be estimated from baseball play-by-play data.

A pitcher's perfect inning is represented by the sequence {[0000], [1000], [2000], [END]}. If a pitcher were to allow three runners on base through walks, say, and then strike out the side, the sequence of baseball states would be represented as {[0000], [0100], [0110], [0111], [1111], [2111], [END]}.

A no-out walk with no runners on first base would take a team from [0000] to [0100], which has the same effect as a single with no runners on base. In fact, a walk is the same as a single with one or two outs, moving from state [1000] to [1100] or from [2000] to [2100]. Quite often, however, a walk is not as good as a hit, because a hit potentially moves base runners further toward home. With a runner on first base, for example, a single could move that runner to third. With no outs, we often see a transition from [0100] to [0101], whereas a walk with no outs and a runner on first is represented as a transition from [0100] to [0110].

We can use baseball states and their associated expected runs data to provide advice to managers and coaches making game-day decisions. Suppose we begin with a runner on first with no outs, represented as [0100]. We note its associated expected runs 0.831. A batter coming to home plate could attempt a sacrifice bunt. If successful, that sacrifice bunt would move

the team to the state [1010] with its associated expected runs 0.644.

It is not hard to understand why baseball analysts eschew the sacrifice bunt—giving up an out to move a runner to second base lowers the expected runs for the team at bat. To make matters worse, not all sacrifice bunts are successful. If a bunt fails to move the runner to second base (due to being popped up, having the runner thrown out at second, or having the bunt fouled off on a third strike), then the result of the failed bunt moves the team from state [0100] to [1100] with its associated expected runs 0.489. A bunt with no outs sometimes results in a double play, leaving the team in the precarious position of two outs and no runners on base—state [2000] with expected runs 0.095. To provide a complete strategic analysis, we would consider all possibilities, including a bunt going for a hit, which moves the team from a no-outs state [0100] to [0110] with expected runs 1.373. Each transition from state to state has an associated probability. Strategic in-game analysis is a matter of computing expected runs across mutually exclusive outcomes or events.

To illustrate probability analysis across mutually exclusive events, consider another in-game decision in baseball, a player's attempt to steal second base with no outs. Across Major League Baseball, the success rate of stealing second base is about 70 percent. With no outs, a successful steal takes a team from [0100] to [0010], moving expected runs from 0.831 to 1.068. An unsuccessful steal, however, moves the team to [1000] with expected runs 0.243. Computing the expected runs of stealing with no outs, we have

$$(0.70 \times 1.068) + (0.30 \times 0.243) = 0.748 + 0.073 = 0.821$$

So stealing second with no outs is not a good idea when the probability of being called “safe” at second is 0.70, the league average. Should the manager signal a steal? That depends on the baserunner’s speed on the base paths, the pitcher’s speed in delivering the ball to home plate, the catcher’s ability to throw runners out, and the skill of middle infielders in tagging runners out—all of which vary from one set of players to the next.

There are many factors to consider in making in-game decisions in baseball, and for each factor there are relevant data to evaluate given the play-by-play record of past games. Because baseball is clearly represented as discrete states and player responsibilities are clearly identified, baseball analysts can evaluate alternative strategies by drawing on the history of the game and player-specific data. The analysis is a straightforward application of mathematics and probability theory.

In evaluating alternative strategies, baseball analysts utilize expected runs estimates relevant to the teams and players in each game and the field of play. Weather conditions and the time of the game (day or night) can also affect expected runs. Some ball parks, such as Coors Field in Denver and Fenway Park in Boston, are known to be favorable to hitters, yielding higher expected runs across all in-game states. Other parks, such as U.S. Cellular Field in Chicago and AT&T Park in San Francisco, are more favorable to pitchers, yielding lower expected runs ([ESPN.com 2015](#)). Umpires at home plate can also affect expected runs due to differences in how balls and strikes are called. An umpire with a wide strike zone favors pitchers, resulting in lower expected runs across all in-game

states. See Weinstock (2012) for evidence of umpire individual differences in strike zone areas.

The baseball manager determines the starting line-up, when to pull the starting pitcher, when to pinch hit, when to bunt or hit away, when to steal second, third, or home, and when to request replay review of an umpire's call. There are those "hidden rules of baseball" that are understood by the best managers and coaches in the game (Thorn and Palmer 1985; Keri 2006; Tango, Lichtman, and Dolphin 2007). The mantra of Sabermetrics is to use objective evidence from baseball to evaluate past player performance and inform in-game decisions. And baseball lovers have long had a fascination with data (Schwarz 2004; James 2010).

Baseball provides a model for strategic analysis—we identify alternative strategies for each play and compute expected results for each strategy. To the extent that other sports can emulate baseball's discrete state-to-state model, they too provide opportunities for strategic analysis with math and probability theory. This is the challenge—to convert continuous action in sports such as basketball, soccer, and hockey into discrete events and to convert offensive and defensive player positions in these sports and football into a finite set of states.

Key in basketball is the decision about which five players to put in the game at any given time, player matchups in guarding opposing players, and the type of defense to employ. A notable case comes from the 2015 NBA Finals. After the Cleveland Cavaliers and Golden State Warriors had split two initial overtime games, there was a dramatic turn of events between games three and four of the series.

In game three of the 2015 NBA Finals, with Cavs guard Kyrie Irving out for the rest of the season with a fractured left knee, LeBron James took charge, playing 46 of the 48 minutes and scoring 40 points. The Cavs won game three 96 to 91, despite scoring only 24 points in the fourth period, compared to the Warriors' 36.

Steve Kerr, manager of the Golden State Warriors, had not started Andre Iguodala all season. But because of Iguodala's prowess in guarding James, Kerr put Andre in the starting lineup in game four. Iguodala and his teammates managed to limit James to 20 points, and the Warriors won game four 103 to 82, with Iguodala and Steph Curry each scoring 22 points.

Iguodala remained in the Warriors' starting lineup for the last two games of the series. LeBron James continued to have an amazing series in terms of minutes played and points scored, but his productivity was kept in check by Iguodala and the Warriors. On the offensive side, Andre Iguodala and Steph Curry scored 14 and 37 points, respectfully, in the Warriors' 104-91 victory in game five, and they each scored 25 points in the Warriors' 105-97 victory in the game six series finale.¹

¹ Regarding the 2015 NBA Finals, see the National Basketball Association (2015) for Finals box scores and see Jenkins (2015a, 2015b) for additional commentary.

Basketball is a team sport, and, except for performance at the free throw line, it is difficult to assess the performance of any one player. Nonetheless, data relating to player performance under alternative matchups can guide coaching decisions.

Individual matchups in basketball, as in many sports, can spell the difference between winning and losing.

Mathematical programming (integer programming, in particular) can help coaches in defining lineup and playing-

time game plans. The objective is to score the most points over four twelve-minute periods. Only five of thirteen players may be on court at once, and each player may have a playing time constraint, defined by his age, physical condition, and susceptibility to injury. A coach can impose player position or player type constraints—two guards (one point guard and one shooting guard), two forwards (one power forward), and one center. Despite the many uncertainties involved in assessing individual player contribution in sports such as basketball, we can take each player’s expected contribution to team points as a starting point in models.

The NBA provides member teams with standard data sets for all games, including box scores and play-by-play data. The NBA provides extensive basketball data from 1946 to present—these historical box scores and play-by-play data are in the public domain.

The “hot hand” in basketball, like hitting streaks and slumps in baseball, has generated considerable interest in the statistical community. Sports enthusiasts and coaches may be surprised to learn that there is little convincing evidence that these are anything but runs of good fortune or bad luck. Bar-Eli, Avugos, and Raab (2006) provide a comprehensive review of research in this area.

Commercial data sources complement NBA-distributed data, offering more detailed or more granular records of each game. There are videos and annotated videos, computer-vision-aided observations and in-person observations. NBA teams can purchase these data and make use of them to obtain competitive advantage. Current spatial data sources, for example, show the position of every player and the ball

twenty-five times a second. As sports analysts and data scientists work with these data sources, we can expect new measures and models to emerge, and many of these measures and models will inform in-game decisions.

For football in-game decisions, coaches have offensive and defensive formations to consider, whether to run or pass, and whether to punt or go for it on fourth and one (or two, or more yards to go). Much has been written about football strategy and coaches' thinking ([American Football Coaches Association 1999b](#), [1999a](#)). The challenge is to convert coaching wisdom and practice into discrete sets of offensive and defensive actions and to trace the success of those actions. Data, measures, and models sometimes contradict conventional approaches to coaching. One area of special note is punting on fourth down, which many sports analysts view as a poor decision ([Romer 2006](#); [Berri and Schmidt 2010](#); [Moskowitz and Wertheim 2011](#)).

Among sports writers, there is widespread use of the word "strategy" to mean what are actually "tactics." We have continued with this practice in the present chapter. Business managers, on the other hand, make a clear distinction between the terms, with "strategy" referring to an overall approach to competition and "tactics" referring to the operational details.

Teams in all sports can utilize mathematical programming to define player lineups and in-game strategies to maximize expected scoring, subject to constraints. Review of strategic alternatives with mathematical programming are illustrated in the edited volume by Ladany and E. Machol ([1977](#)). And contributors to the *Journal of Quantitative Analysis in Sports* have been most influential in developing predictive models to

guide strategy.

Lindsey (1963) provided strategic analysis in baseball, setting the stage for subsequent work with expected runs across possible game situations or states. Books by Kemeny and Snell (1976), Ross (2014), Puterman (2005), and Privault (2013) review the mathematics of Markov chains. Bukiet, Elliotte, and Palacios (1997) and Albert (2003) illustrate Markov chain analysis of baseball. Marchi and Albert (2014) provide R programs for computing expected runs and for simulating baseball games with Markov chains.

Baumer and Zimbalist (2014) provide a comprehensive review of Sabermetrics and its relevance to baseball strategy. They critique *Moneyball*, both the Lewis' (2003) book and the movie, showing how these popular works present an inaccurate and incomplete rendering of Sabermetric thought.

For further discussion of basketball analytics, see Oliver (2004), Winston (2009), Shea and Baker (2013), Shea (2014), and Martin (2016). For football analytics and strategy see Carroll, Palmer, and Thorn (1988) and Carrol *et al.* (1998). Brown (2015) and Dresow (2015) document football coaching strategies, but present little analytics. Those interested in understanding the philosophy of one coach in particular, Bill Belichick, can refer to Halberstam (2005) and Belichick (2008). Soccer strategy is reviewed in books by Wells (2008) and Wilson (2008).

6. Crafting a Message

Dan: "Kalb."

Eugene: "Thanks for the Lakers Tickets."

Dan: "You bet."

Eugene: "The seats were terrific. But I'm still not going to advertise in the magazine. My son-in-law tells me that people don't read much any more. Too much effort moving their eyes back and forth. So we're going to put most of our budget into television, radio, Internet."

Dan: "OK."

Eugene: "OK? What does that mean?"

Dan: "I'm not going to try to sell you."

Eugene: "Why the hell not? You're a salesman."

Dan: "Yeah, just not a very good one. That's all."

Eugene: "I'll say."

Dan: "But I am going to ask you one favor. I'm going to leave you an issue of the magazine. And I'm personally going to send you a new one every week. Now I'll call you in a few weeks. And if you want to, we'll talk. There's a great article in there comparing today's quarterbacks with Johnny Unitas."

Eugene: "Unitas would kick their butts. So this is your sales pitch?"

Dan: "I've been with the magazine for twenty years. I believe in it."

—DENNIS QUAID AS DAN FOREMAN AND PHILIP BAKER HALL AS
EUGENE KALB IN *In Good Company* (2004)

Crafting a message, marketing communications, brand development, sales, and marketing are important to professional sports teams, just as they are to any business. There are ticket pricing, concessions, and team-branded merchandise options to consider. There are advertisers, sponsors, and business partnerships to pursue. Teams need to get people's attention, to attract interest, and ultimately to build loyalty. Teams benefit when they turn sports consumers into fans.

Data science methods and models have wide applicability on the business side of professional sports. Sports teams benefit by having an understanding of sports and entertainment markets.

To show how sport fits into the entertainment space, we begin with a simple example. Consider seven event entertainment or activity options available to Washington D.C. consumers in the fall:

- Comedy at Warner Theater (Comedy)
- National Symphony Orchestra (Symphony)
- National Zoological Park (Zoo)
- Popular Music at EagleBank Arena (Pop Music)
- Smithsonian Museum (Museum)
- Washington Capitals Game (Hockey)
- Washington Redskins Game (Football)

Noting similarities among activities, we can see how they relate to one another in the entertainment space. When studying small sets of activities, we can seek direct ratings or rankings across all pairs of activities. With k activities there will be $k(k - 1)/2$ pairs or combinations of two activities. With

seven activities, there are $\frac{7(7-1)}{2} = 21$ pairs of activities. So we can use twenty-one similarity ratings to construct a distance matrix showing how far each activity is from every other activity.

We can represent similarity ranks in a matrix. Table 6.1 shows the lower triangle of a data matrix for the seven activities in the Washington D.C. area. Because larger numbers correspond to larger differences between activities, we call this a dissimilarity or distance matrix. We can use this matrix to construct a product positioning map.

	Comedy	Symphony	Zoo	Pop Music	Museum	Hockey	Football
Comedy	-						
Symphony	6	-					
Zoo	11	10	-				
Pop Music	5	3	9	-			
Museum	8	2	4	7	-		
Hockey	15	19	17	13	21	-	
Football	14	18	16	12	20	1	-

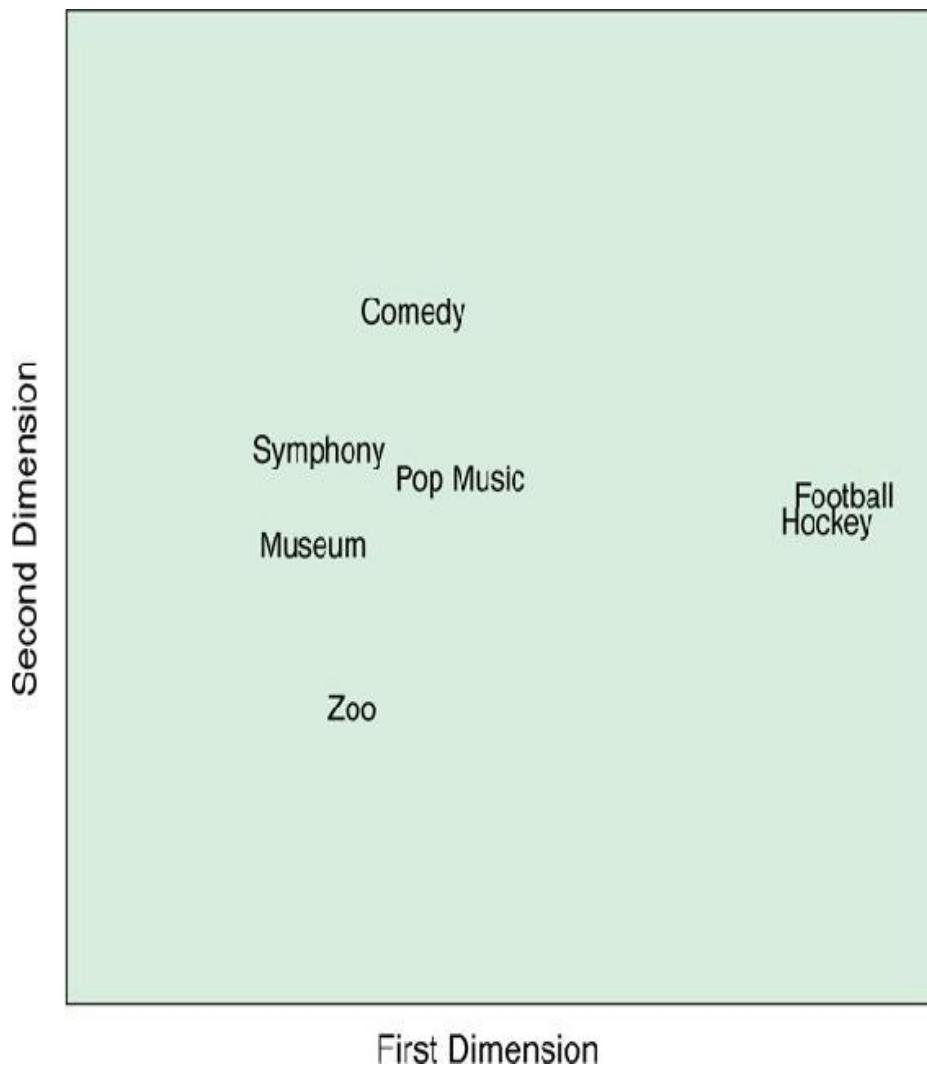
Table 6.1. Dissimilarity Matrix for Entertainment Events and Activities

Similarity judgments are especially useful in product or service categories for which attributes are difficult to identify or describe, such as categories defined by style, look, odor, or flavor. The resulting distance or dissimilarity matrix serves as input to multidimensional scaling algorithms, which produce product positioning maps. The distance or dissimilarity matrix can also serve as input to cluster analysis algorithms, which organize products or services into groups.

When studying a large group of products and services, we might use data about product attributes or consumer ratings of

product attributes as input to multidimensional scaling or cluster analysis. For this work we would use multivariate distance measures to construct the distance matrix.

Figure 6.1 shows the product positioning or perceptual map of the activities in Washington D.C. area. In many research contexts, products or services close together on maps represent potential substitutes for one another. In the eyes of this consumer, sports are seen as decidedly different from other entertainment events and activities. Seeing a football game is not a substitute for going to a museum or listening to a symphony orchestra, whereas seeing a football game may well be a substitute for seeing a hockey game.



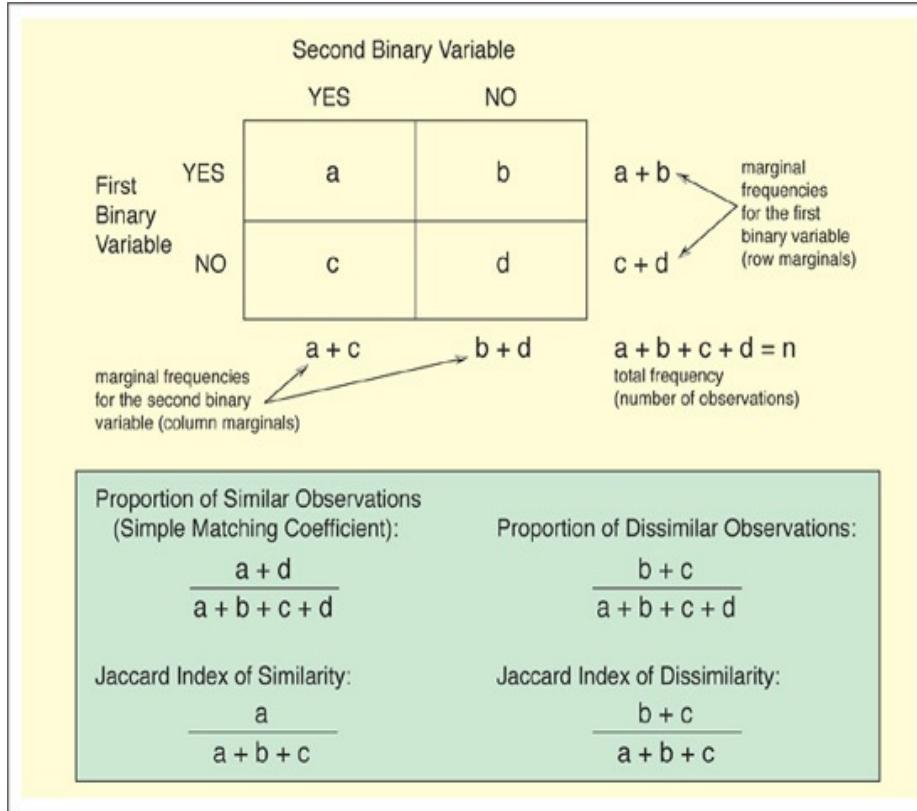
**Figure 6.1. How Sports Fit into the Entertainment Space
(Or Not)**

We call a product positioning map a *perceptual map* because it is a rendering of consumer perceptions of activities—in this case one consumer’s perceptions. And it is not too hard to imagine how making a map of products in space for one consumer could translate into making a map for many consumers. We would ask a group of consumers to rank pairs of activities on similarity and then place average ranks in the dissimilarity matrix.

Multidimensional scaling uses dissimilarities or distances as

inputs, providing a map as output. The orientation of the map is arbitrary. We can rotate the map without changing the solution. We can develop a reflection of the map without changing the solution. We can exchange the top for the bottom or the left for the right without changing the solution.

Maps based on different similarity metrics will also differ. To assess similarity between activities, products, or brands, we often use the Jaccard dissimilarity index (see figure 6.2). When different multivariate methods yield decidedly different results, as they often do, we use our best judgment to choose a solution that will be easiest to communicate to management.



The Jaccard index of dissimilarity is obtained from a two-by-two table for binary variables. The table above shows how to compute the Jaccard indices of similarity and dissimilarity from the frequencies a , b , c , and d in the two-bytwo table. In many research problems the Jaccard index of dissimilarity makes more sense as an index of dissimilarity than the proportion of dissimilar observations.

For pairs of low-incidence binary variables, the proportion of dissimilar

observations is low because most observations are in the *NO/NO* cell of the two-by-two table. These *NO/NO* observations are more a reflection of low-incidence than similarity between the binary variables. In computing the Jaccard dissimilarity index, we drop the *NO/NO* cell frequency, thus avoiding problems with low-incidence binary variables. Similarity and dissimilarity indices play an important role in such methods as multidimensional scaling, cluster analysis, and nearest-neighbor analysis. Market basket analysis often begins with consumers or individual market baskets defining the rows of a binary data matrix and products or services defining the columns. Indices of similarity show how likely it is that two products will be included in a shopper's market basket. Further discussion of indices of similarity and dissimilarity may be found in Kaufman and Rousseeuw (1990).

Figure 6.2. *Indices of Dissimilarity between Pairs of Binary Variables*

Whether concerned with new or current products and services, product positioning is an important part of business and marketing strategy. There are basic questions to address: What constitutes the category of products? How do products within a category compare with one another? What can a organization do to distinguish its products and services from those offered by other organizations?

Product positioning maps are especially useful in product planning and competitive analysis. Products close to one another in space may be thought of as substitute products or close competitors. Open areas in the product space may represent opportunities for new, differentiated products. These same technologies may be applied to brands to obtain information about brand positioning and to guide branding strategy.

Product or brand positioning may be studied in concert with product and brand preferences, yielding a joint perception/preference mapping of products or brands in space.

Critical to strategic product positioning are areas of the product space most desirable to consumers. Product managers like to find areas of the product space where there are many potential customers and few competitive brands or products.

Multivariate methods study relationships among many variables. The language of principal components, factor analysis, multidimensional scaling, and other multivariate methods suggests that we are trying to identify underlying or latent dimensions in the data. We want to reduce the number of dimensions being considered, so we can focus on the most critical or salient dimensions.

Cluster analysis, a class of multivariate methods, represents an alternative to multidimensional scaling. Quite often, products or activities within the same cluster are potential substitutes for one another. Cluster analysis is also appropriate for classification problems in which classes are not known in advance. Products that are similar to one another in terms of their values on measured attributes in the data set get placed in the same cluster. Products within a cluster should be more similar to one another than they are to objects in other clusters.

Going to a museum or zoo involves active participation, walking while observing. Listening to a symphony or watching a movie, on the other hand, are passive activities, sitting while observing. Just as entertainment events and activities differ from one another, consumers differ from one another. We often use cluster analysis to identify market segments or groups of consumers that differ from one another. Sports fans are not a homogeneous group. They come from all economic levels and educational backgrounds. There is wide variability as well in active participants in sports, an additional

factor for defining segments.

We choose methods of analysis appropriate for the data and the business problem at hand. We choose methods for presenting results that are both appropriate for the data and meaningful to management. A plot of points in two dimensions is easier to understand than a large table of numbers. It is hard to see relationships in large matrices of correlations or dissimilarities, but it is not so hard to see relationships in maps.

Conjoint analysis is a modeling method particularly useful in sports marketing. Managers often ask about what drives buyer choice. They want to know what is important to choice or which factors determine choice. To the extent that buyer behavior is affected by product features, brand, and price, managers are able to influence buyer behavior, increasing demand, revenue, and profitability.

Ask buyers what they want, and they may say, *the best of everything*. Ask them what they would like to spend, and they may say, *as little as possible*. There are limitations to assessing buyer willingness to pay and product preferences with direct-response rating scales, or what are sometimes called self-explicative scales. Simple rating scale items arranged as they often are, with separate questions about product attributes, brands, and prices, fail to capture tradeoffs that are fundamental to consumer choice. To learn more from buyer surveys, we provide a context for responding and then gather as much information as we can. This is what conjoint and choice studies do, and many of them do it quite well.

Conjoint analysis is really *conjoint measurement*. Marketing analysts present product profiles to consumers. Product

profiles are defined by their attributes. By ranking, rating, or choosing products, consumers reveal their preferences for products and the corresponding attributes that define products. The computed attribute importance values and part-worths associated with levels of attributes represent measurements that are obtained as a group or jointly—thus the name conjoint analysis. The task—ranking, rating, or choosing—can take many forms.

When doing traditional conjoint analysis, we utilize regression analysis and *sum contrasts*, so that the sum of fitted coefficients across the levels of each attribute is zero. The fitted regression coefficients represent conjoint measures of utility called *part-worths*. Part-worths reflect the strength of individual consumer preferences for each level of each attribute in the study.

Positive part-worths add to a product's value in the mind of the consumer. Negative part-worths subtract from that value. When we sum across the part-worths of a product, we obtain a measure of the utility or benefit to the consumer. A linear model fit to preference rankings is an example of *traditional conjoint analysis*, a modeling technique designed to show how product attributes affect purchasing decisions.

To demonstrate conjoint analysis, we consider seating and ticket prices at Dodger Stadium. There are four large seating areas for fans: field, loge, reserved, and top deck. A “loge,” according to *Webster’s Collegiate Dictionary* (eleventh ed.), is “a raised section or level of seats in a sports stadium.” At Dodger Stadium, field level seats are closest to the field of play. Next is the loge level, followed by reserved seating. The top level is far removed from the field of play. Ticket prices

vary widely across and within these areas, where sections within the areas might be described as “Front Row,” “Box,” or “VIP,” for example. One other factor included in this demonstration study is promotions or give-aways. Some games might have a ball, bobblehead doll, or cap promotion available to the first fifty thousand fans. Table 6.2 shows ranking data for one fan across sixteen product profiles for Dodger Stadium seating. To display the results of the conjoint analysis, we use a special type of dot plot called the *spine chart*, shown in figure 6.3.

question	price	seating	boxvip	frontrow	promotion	ranking
1	Price \$45	Reserved	Box/VIP NO	Front Row NO	None	10
2	Price \$20	Field	Box/VIP NO	Front Row NO	Ball	16
3	Price \$20	Reserved	Box/VIP YES	Front Row YES	Bobblehead	13
4	Price \$45	Loge	Box/VIP NO	Front Row YES	Bobblehead	11
5	Price \$20	Loge	Box/VIP YES	Front Row NO	None	14
6	Price \$95	Loge	Box/VIP NO	Front Row NO	Cap	6
7	Price \$70	Field	Box/VIP NO	Front Row YES	None	12
8	Price \$70	Reserved	Box/VIP YES	Front Row NO	Cap	7
9	Price \$45	Top Deck	Box/VIP YES	Front Row NO	Ball	3
10	Price \$95	Reserved	Box/VIP NO	Front Row YES	Ball	5
11	Price \$70	Top Deck	Box/VIP NO	Front Row NO	Bobblehead	2
12	Price \$70	Loge	Box/VIP YES	Front Row YES	Ball	9
13	Price \$95	Field	Box/VIP YES	Front Row NO	Bobblehead	8
14	Price \$20	Top Deck	Box/VIP NO	Front Row YES	Cap	4
15	Price \$45	Field	Box/VIP YES	Front Row YES	Cap	15
16	Price \$95	Top Deck	Box/VIP YES	Front Row YES	None	1

Table 6.2. Consumer Preference Data for Dodger Stadium Seating

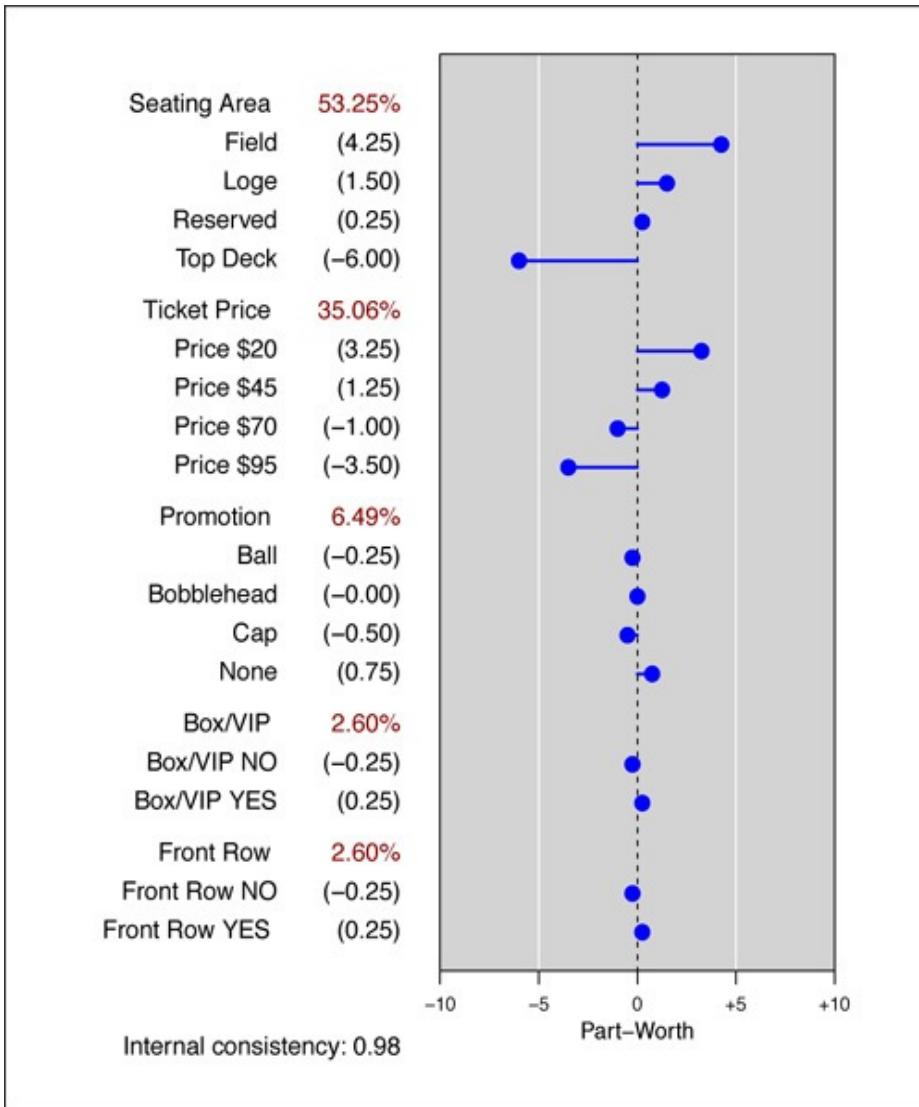


Figure 6.3. Consumer Preferences for Dodger Stadium Seating

In the spine chart, part-worths can be displayed on a common part-worths scale across attributes. The vertical line in the center, the spine, is anchored at zero. The part-worth of each level of each attribute is displayed as a dot with a connecting horizontal line, extending from the spine. Preferred product or service characteristics have positive part-worths and fall to the right of the spine. Less preferred product or service characteristics fall to the left of the spine. The relative

importance of attributes in a conjoint analysis is defined using the ranges of part-worths within attributes. These importance values are scaled so the sum across all attributes is 100 percent.

What does the spine chart say about this consumer's seating preferences? It shows that the seating area is of primary importance. Next in order of importance is price. Location within the general seating area, defined by "Front Row," "Box," or "VIP" designations, is considerably less important than general seating area itself. Promotions appear to have little import, as this consumer may prefer no promotions to receiving a ball, cap, or bobblehead doll.

This simple study measures consumer preferences for seating. It should come as no surprise that seats closer to the field are preferred or that lower prices are preferred. But the real value of a study like this comes in what it shows us about consumer willingness to pay.

To assess willingness to pay within the context of conjoint measures, we note the part-worth differential across the range of ticket prices. From figure 6.3, we see that a part-worth of 3.25 corresponds to a ticket price of \$20. Similarly, a part worth of -3.50 corresponds to a ticket price of \$95. A few simple calculations provide a dollar-metric for part-worths:

$$|3.25 - (-3.50)| = 6.75 \text{ part-worth differential (utils)}$$

$$|20 - 95| = 75 \text{ ticket price differential (dollars)}$$

$$6.75 \text{ utils} = 75 \text{ dollars}$$

$$1 \text{ util} = 11.11 \text{ dollars}$$

Each unit of part-worth (call it a *util* for want of a better term)

is worth \$11.11 to this consumer. And for this consumer the part-worth difference between a field level seat and a loge seat is $|4.25 - 1.50| = 2.75$ *utils*. So this consumer would be willing to spend $2.75 \times 11.11 = 30.55$ dollars more for a field seat than a loge seat.

Traditional conjoint analysis fits a linear model to each individual's ratings or rankings, thus measuring the utility or part-worth of each level of each attribute, as well as the relative importance of attributes. Traditional conjoint analysis is one of many types of conjoint analysis (Orme 2013).

To assess willingness to pay more fully, we would employ a choice-based conjoint approach including a *none* alternative. Figure 6.4 shows what a choice item would look like in such a study. We present numerous choice items to each consumer respondent and use methods of discrete choice analysis to estimate part-worths and attribute importance. Miller (2015a) shows how to use Bayesian hierarchical models to analyze choice data. Talluri and van Ryzin (2004) describe discrete choice methods in revenue management, as appropriate for assessing willingness to pay in capacity-limited, variable-pricing problems, such as airline and stadium seating.

Select the ticket you would most likely purchase. Or if you would be unlikely to purchase any of these tickets, select None.

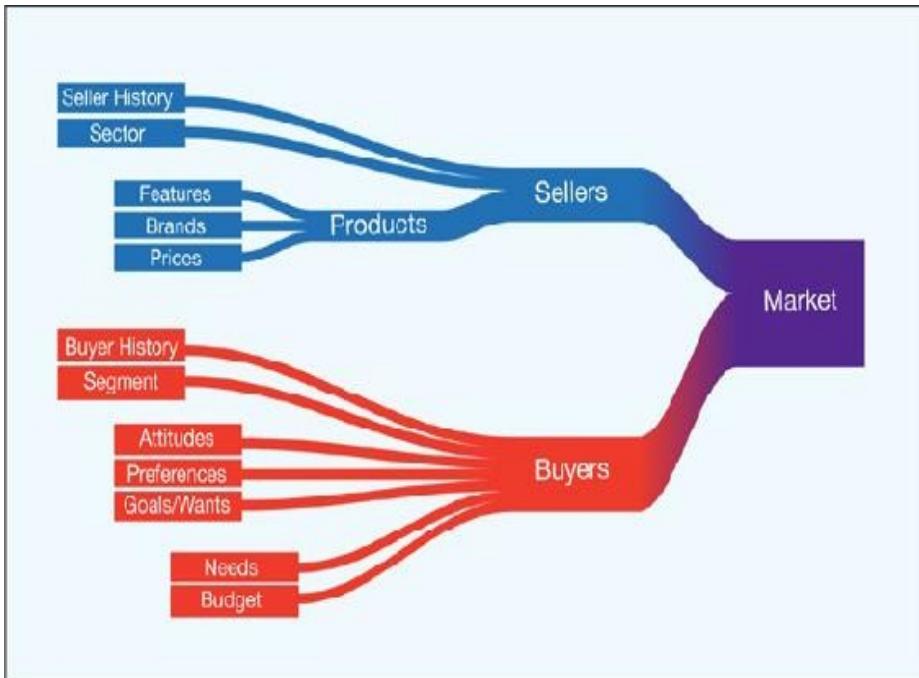
Loge Level Box Seat Front Row Bobblehead Doll	Reserved Level VIP Seating Middle Row Baseball Cap	Top Level General Seating Middle Row No Promotion	None
\$70	\$45	\$20	

Figure 6.4. Choice Item for Assessing Willingness to Pay for Tickets

The measures we obtain from conjoint studies may be analyzed to identify consumer segments. We can use conjoint measures to predict consumer choices in the marketplace. We can draw on the results of conjoint studies to perform marketplace simulations, exploring alternative product designs and pricing policies. Consumers reveal their preferences in responses to surveys and ultimately in choices they make in the marketplace.

Figure 6.5 provides a framework for understanding buyers' and sellers' choices in a market. A market is a place where buyers and sellers get together. Buyers represent the demand side, and sellers the supply side. We use methods and models to predict what will happen in a market, how much of a product will be sold and what will happen with prices.

Studying the behavior of markets in the aggregate is the work of economics. Marketing research and data science study market segments and the behavior of buyers within segments.



Source: Miller (2015a).

Figure 6.5. The Market: A Meeting Place for Buyers and Sellers

Professional sports teams can ask basic questions about buyers. Consumers and fans have varied histories, preferences, and attitudes. Some consumers are fans of a particular team, others are fans of a particular sport, and still others are fans of sports in general. A team's market should not be seen as limited to the city where that team resides. There are opportunities for revenue growth through national branding and media contracts. Location-bound sports stadia have limited capacity. Sporting events distributed through media have no such limits.

What is the market for a team? Many Cub fans across the nation have little connection with Chicagoland. The Dallas Cowboys can be described as "America's team" by fans who have little intention to visit Texas. Small-market teams such as the Green Bay Packers of the NFL or the Oklahoma City

Thunder of the NBA can build fan bases well beyond their city limits. To use a sports analogy, media “level the playing field” between small-and major-market teams.

Data science methods and models are relevant to all markets, business-to-consumer and business-to-business markets alike, and they are certainly relevant to professional sports marketing. Blattberg, Kim, and Neslin (2008) and Miller (2015a) review methods and models relevant to the general domain of marketing data science. Rein, Kotler, and Shields (2006) show how general marketing principles apply to professional sports team marketing.

Preference scaling has a long history in psychometrics (Thurstone 1927; Young and Householder 1938; Richardson 1938). Guilford (1954, first published in 1936) and Torgerson (1958) provided in-depth treatments. Traditional unidimensional scaling methods, building as they do on a paired comparison preference matrix, may be used for data arising from actual paired comparisons, rank orders, and multiple rank orders, as well as from best-worst scaling items, choice studies, pick lists, and elimination pick lists. Miller (2015a) provides examples.

There are many good sources for learning about multidimensional scaling (Davison 1992; Cox and Cox 1994; Carroll and Green 1997; Borg and Groenen 2010). Lilien and Rangaswamy (2003) discuss methods for the joint mapping of perception and preference. R software for multidimensional scaling is described in Venables and Ripley (2002).

Multivariate methods are reviewed by Seber (2000), Manly (1994), Sharma (1996), Gnanadesikan (1997), Johnson and Wichern (1998), and Izenman (2008). Principal component

bipLOTS represent an alternative to multidimensional scaling plots (Gabriel 1971; Gower and Hand 1996). Biplots allow us to plot consumers and products/brands in the same space.

Conjoint measurement, a critical tool of marketing data science, focuses on buyers or the demand side of markets. The method was originally developed by Luce and Tukey (1964). Orme (2013) presents a comprehensive review of conjoint methods.

Exhibit 6.1 shows the R program for product positioning using the entertainment event and activities example. The R program draws on the work of Ripley *et al.* (2015). We show the corresponding Python program in exhibit 6.2.

Exhibits 6.3 and 6.4 show R and Python programs for analyzing ranking or rating data for consumer preferences. The programs perform traditional conjoint analysis. The R program shows how to construct a spine chart, which is a customized data visualization for conjoint and choice studies. Using standard R graphics, the program builds this chart one point, line, and text string at a time. The precise placement of points, lines, and text is under program control.

Exhibit 6.1. Mapping Entertainment Events and Activities (R)

Click here to view code image

```
# Product Positioning of Entertainment Events  
and Activities (R)  
  
library(MASS) # includes functions for  
multidimensional scaling  
library(wordcloud) # textplot utility to  
avoid overlapping text
```

```
USE_METRIC_MDS <- FALSE # metric versus
nonmetric toggle

# utility function for converting a distance
structure
# to a distance matrix as required for some
routines and
# for printing of the complete matrix for
visual inspection.
make.distance.matrix <-
function(distance_structure)
{
  n <- attr(distance_structure, "Size")
  full <- matrix(0,n,n)
  full[lower.tri(full)] <-
distance_structure
  full+t(full)
}

# enter data into a distance structure as
required for various
# distance-based routines. That is, we enter
the upper triangle
# of the distance matrix as a single vector of
distances
distance_structure <-
as.single(c(6,11,5,8,15,14,10,3,2,19,18,9,4,17

# provide a character vector of entertainment
event or activity names
activity_names <-
c("Comedy", "Symphony", "Zoo", "Pop Music",
"Museum", "Hockey", "Football")

attr(distance_structure, "Size") <-
length(activity_names) # set size attribute

# check to see that the distance structure has
been entered correctly
# by converting the distance structure to a
distance matrix
```

```

# using the utility function
make.distance.matrix, which we had defined
distance_matrix <-
unlist(make.distance.matrix(distance_structure))
cat("\n","Distance Matrix of Seven
Activities","\n")
print(distance_matrix)

if (USE_METRIC_MDS)
{
  # apply the metric multidimensional
  scaling algorithm and plot the map
  mds_solution <-
  cmdscale(distance_structure, k=2, eig=T)
}

# apply the nonmetric multidimensional scaling
algorithm
# this is more appropriate for rank-order data
# and provides a more satisfactory solution
here

if (!USE_METRIC_MDS)
{
  mds_solution <- isoMDS(distance_matrix, k
= 2, trace = FALSE)
}
pdf(file =
"plot_nonmetric_mds_seven_activities.pdf",
width=8.5, height=8.5) # opens pdf
plotting device

# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))

# original solution
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2]

```

```

# set up the plot but do not plot points...
use names for points
plot(First_Dimension, Second_Dimension, type =
  "n", cex = 1.5,
  xlim = c(-15, 15), ylim = c(-15, 15)) # 
first page of pdf plots
# We plot the sport names in the locations
where points normally go.
text(First_Dimension, Second_Dimension, labels
= activity_names,
  offset = 0.0, cex = 1.5)
title("Seven Activities (initial solution)")

# reflect the horizontal dimension
# multiply the first dimension by -1 to get
reflected image
First_Dimension <- mds_solution$points[,1] *
-1
Second_Dimension <- mds_solution$points[,2]
plot(First_Dimension, Second_Dimension, type =
  "n", cex = 1.5,
  xlim = c(-15, 15), ylim = c(-15, 15)) # 
second page of pdf plots
text(First_Dimension, Second_Dimension, labels
= activity_names,
  offset = 0.0, cex = 1.5)
title("Seven Activities (horizontal
reflection)")

# reflect the vertical dimension
# multiply the section dimension by -1 to get
reflected image
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2] *
-1
plot(First_Dimension, Second_Dimension, type =
  "n", cex = 1.5,
  xlim = c(-15, 15), ylim = c(-15, 15)) # 
third page of pdf plots
text(First_Dimension, Second_Dimension, labels

```

```

= activity_names,
    offset = 0.0, cex = 1.5)
title("Seven Activities (vertical
reflection)")

# multiply the first and second dimensions by
-1
# for reflection in both horizontal and
vertical directions
First_Dimension <- mds_solution$points[,1] *
-1
Second_Dimension <- mds_solution$points[,2] *
-1
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # fourth page of pdf plots
text(First_Dimension, Second_Dimension, labels
= activity_names,
      offset = 0.0, cex = 1.5)
title("Seven Activities (horizontal and
vertical reflection)")
dev.off() # closes the pdf plotting device

pdf(file =
"plot_pretty_original_mds_seven_activities.pdf",
      width=8.5, height=8.5) # opens pdf
plotting device
# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] # no reflection
Second_Dimension <-
mds_solution$points[,2] # no reflection
# wordcloud utility for plotting with no
overlapping text
textplot(x = First_Dimension,
      y = Second_Dimension,
      words = activity_names,

```

```
show.lines = FALSE,
xlim = c(-15, 15), # extent of horizontal
axis range
ylim = c(-15, 15), # extent of vertical
axis range
xaxt = "n", # suppress tick marks
yaxt = "n", # suppress tick marks
cex = 1.15, # size of text points
mgp = c(0.85, 1, 0.85), # position of
axis labels
cex.lab = 1.5, # magnification of axis
label text
xlab = "First Dimension",
ylab = "Second Dimension")
dev.off() # closes the pdf plotting device
```

**Exhibit 6.2. Mapping Entertainment Events and Activities
(Python)**

[Click here to view code image](#)

```
# Product Positioning of Entertainment Events
and Activities (Python)

# prepare for Python version 3x features and
functions
from __future__ import division,
print_function

# import packages for multivariate analysis
import numpy as np # arrays and numerical
processing
import scipy
import matplotlib.pyplot as plt # 2D plotting

# alternative distance metrics for
multidimensional scaling
from sklearn.metrics import
```

```

euclidean_distances
from sklearn.metrics.pairwise import
linear_kernel as cosine_distances
from sklearn.metrics.pairwise import
manhattan_distances as manhattan_distances

from sklearn import manifold # 
multidimensional scaling

# These are the original data from one
respondent
# Pairs of activities are judged on their
similarity
# Smaller numbers are more similar to one
another
# Zero on the diagonal means no difference
#    0      6     11      5      8     15     14   Comedy
#    6      0     10      3      2     19     18   Symphony
#   11     10      0      9      4     17     16    Zoo
#    5      3      9      0      7     13     12    Pop
Music
#    8      2      4      7      0     21     20   Museum
#   15     19     17     13     21      0      1   Hockey
#   14     18     16     12     20      1      0   Football

# define a numpy array for these data
distance_matrix = np.array([
[0,      6,     11,      5,      8,     15,     14], \
[6,      0,     10,      3,      2,     19,     18], \
[11,     10,      0,      9,      4,     17,     16], \
[5,      3,      9,      0,      7,     13,     12], \
[8,      2,      4,      7,      0,     21,     20], \
[15,     19,     17,     13,     21,      0,      1], \
[14,     18,     16,     12,     20,      1,      0]]))

# check to see that the distance structure has
been entered correctly
print(distance_matrix)
print(type(distance_matrix))

```

```

# apply the multidimensional scaling algorithm
and plot the map
mds_method = manifold.MDS(n_components = 2,
random_state = 9999,\
dissimilarity = 'precomputed')
mds_fit = mds_method.fit(distance_matrix)
mds_coordinates =
mds_method.fit_transform(distance_matrix)

activity_label = ["Comedy", "Symphony", "Zoo",
\
"Pop Music", "Museum", "Hockey",
"Football"]

# plot mds solution in two dimensions using
activity labels
# defined by multidimensional scaling
plt.figure()
plt.scatter(mds_coordinates[:,0],mds_coordinates[:,1],
facecolors = 'none', edgecolors =
'none') # points in white (invisible)
labels = activity_label
for label, x, y in zip(labels,
mds_coordinates[:,0], mds_coordinates[:,1]):
    plt.annotate(label, (x,y), xycoords =
'data')
plt.xlabel('First Dimension')
plt.ylabel('Second Dimension')
plt.show()
plt.savefig('fig_positioning_products_mds_activiti
bbox_inches = 'tight', dpi=None,
facecolor='w', edgecolor='b',
orientation='landscape', papertype=None,
format=None,
transparent=True, pad_inches=0.25,
frameon=None)

```

Exhibit 6.3. Preferences for Sporting Events—Conjoint Analysis (R)

[Click here to view code image](#)

```
# Consumer Preferences for Sporting  
Events---Conjoint Analysis (R)  
  
library(support.CEs) # package for survey  
construction  
  
# generate a balanced set of product profiles  
for survey  
# varies with each call to the function  
# so each individual gets a different set of  
sixteen items  
provider.survey <- Lma.design(attribute.names  
=  
  list(price = c("PRICE $20", "PRICE  
$45", "PRICE $70", "PRICE $95"),  
    seating = c("Field", "Loge", "Reserved", "Top  
Deck"),  
    boxvip = c("Box/VIP NO", "Box/VIP YES"),  
    frontrow = c("Front Row NO", "Front Row  
YES"),  
    promotion =  
    c("Ball", "Bubblehead", "Cap", "None")),  
    nalternatives = 1, nblocks=1, seed =  
7777)  
print(questionnaire(provider.survey)) # print  
survey design for review  
  
sink("questions_for_survey.txt") # send  
survey to external text file  
questionnaire(provider.survey)  
sink() # send output back to the screen  
print.digits <- 2 # set number of digits on  
print and spine chart  
  
# user-defined function for printing conjoint  
measures  
if (print.digits == 2)  
  pretty.print <- function(x)
```

```
{sprintf("%1.2f", round(x, digits = 2))}  
if (print.digits == 3)  
  pretty.print <- function(x)  
{sprintf("%1.3f", round(x, digits = 3))}  
  
#  
-----  
# user-defined function for printing conjoint  
measures  
# spine chart accommodates up to 45  
part-worths on one page  
# |part-worth| <= 40 can be plotted directly  
on the spine chart  
# |part-worths| > 40 can be accommodated  
through  
# standardization# user-defined function  
for spine chart  
#  
-----  
spine.chart <- function(conjoint.results,  
  color.for.part.worth.point = "blue",  
  color.for.part.worth.line = "blue",  
  left.side.symbol.to.print.around.part.worths  
= "(",  
  right.side.symbol.to.print.around.part.worths  
= ")",  
  left.side.symbol.to.print.around.importance  
= "",  
  right.side.symbol.to.print.around.importance  
= "",  
  color.for.printing.importance.text = "dark  
red",  
  color.for.printing.part.worth.text =  
"black",  
  draw.gray.background = TRUE,  
  draw.optional.grid.lines = TRUE,  
  print.internal.consistency = TRUE,  
  fix.max.to.4 = FALSE,  
  put.title.on.spine.chart = FALSE,  
  title.on.spine.chart = paste("TITLE GOES
```

```

HERE IF WE ASK FOR ONE", sep=""),
plot.framing.box = TRUE,
do.ordered.attributes = TRUE) {
# fix.max.to.4 option to override the range
for part-worth plotting

if(!do.ordered.attributes) effect.names <-
conjoint.results$attributes
if(do.ordered.attributes) effect.names <-
conjoint.results$ordered.attributes

number.of.levels.of.attribute <- NULL
for(index.for.factor in
seq(along=effect.names))
number.of.levels.of.attribute <-
c(number.of.levels.of.attribute,
length(conjoint.results$xlevels[[effect.name

# total number of levels needed for vertical
length of spine the spine plot
total.number.of.levels <-
sum(number.of.levels.of.attribute)

# define size of spaces based upon the
number of part-worth levels to plot
if(total.number.of.levels <= 20) {
smaller.space <- 0.01
small.space <- 0.02
medium.space <- 0.03
large.space <- 0.04
}

if(total.number.of.levels > 20) {
smaller.space <- 0.01 * 0.9
small.space <- 0.02 * 0.9
medium.space <- 0.03 * 0.9
large.space <- 0.04 * 0.9
}

if(total.number.of.levels > 22) {

```

```

smaller.space <- 0.01 * 0.85
small.space <- 0.02 * 0.85
medium.space <- 0.03 * 0.825
large.space <- 0.04 * 0.8
}

if(total.number.of.levels > 25) {
  smaller.space <- 0.01 * 0.8
  small.space <- 0.02 * 0.8
  medium.space <- 0.03 * 0.75
  large.space <- 0.04 * 0.75
}

if(total.number.of.levels > 35) {
  smaller.space <- 0.01 * 0.65
  small.space <- 0.02 * 0.65
  medium.space <- 0.03 * 0.6
  large.space <- 0.04 * 0.6
}

# of course there is a limit to how much we
can plot on one page
if (total.number.of.levels > 45)
  stop("\n\nTERMINATED: More than 45
part-worths on spine chart\n")
  part.worth.plotting.list <-
conjoint.results$part.worths

# check the range of part-worths to see
which path to go down for plotting
# initialize these toggles to start
max.is.less.than.40 <- FALSE
max.is.less.than.20 <- FALSE
max.is.less.than.10 <- FALSE
max.is.less.than.4 <- FALSE
max.is.less.than.2 <- FALSE
max.is.less.than.1 <- FALSE

if
(max(abs(min(unlist(part.worth.plotting.list)),na.r

```

```
    max(unlist(part.worth.plotting.list),na.rm=TRUE
<= 40) {
  max.is.less.than.40 <- TRUE
  max.is.less.than.20 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

if
(max(abs(min(unlist(part.worth.plotting.list),na.r
  max(unlist(part.worth.plotting.list),na.rm=TRUE
<= 20) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- TRUE
  max.is.less.than.10 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

if(max(abs(min(unlist(part.worth.plotting.list),
  max(unlist(part.worth.plotting.list),na.rm=TRUE
<= 10) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.10 <- TRUE
  max.is.less.than.4 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

if
(max(abs(min(unlist(part.worth.plotting.list),na.r
  max(unlist(part.worth.plotting.list),na.rm=TRUE
<= 4) {
```

```

max.is.less.than.40 <- FALSE
max.is.less.than.20 <- FALSE
max.is.less.than.4 <- TRUE
max.is.less.than.10 <- FALSE
max.is.less.than.2 <- FALSE
max.is.less.than.1 <- FALSE
}
if(max(abs(min(unlist(part.worth.plotting.list),
  max(unlist(part.worth.plotting.list),na.rm=TRUE
<= 2) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.2 <- TRUE
  max.is.less.than.1 <- FALSE
}

if(max(abs(min(unlist(part.worth.plotting.list),
  max(unlist(part.worth.plotting.list),na.rm=TRUE
<= 1) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- TRUE
}

# sometimes we override the range for
part-worth plotting
# this is not usually done... but it is an
option
if (fix.max.to.4) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.4 <- TRUE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

```

```

        }

if (!max.is.less.than.1 &
!max.is.less.than.2 & !max.is.less.than.4 &
!max.is.less.than.10 &
!max.is.less.than.20 & !max.is.less.than.40)
  stop("\n\nTERMINATED: Spine chart cannot
plot |part-worth| > 40")

# determine point positions for plotting
part-worths on spine chart
if (max.is.less.than.1 | max.is.less.than.2
| max.is.less.than.4 | 
  max.is.less.than.10 | max.is.less.than.20
| max.is.less.than.40) {
  # begin if-block plotting when all
  part-worths in absolute value
  # are less than one of the tested range
  values
  # part-worth positions for plotting
  # end if-block plotting when all part-worths
  in absolute value
  # are less than one of the tested range
  values
  # offsets for plotting vary with the
  max.is.less.than setting
  if(max.is.less.than.1) {
    list.scaling <- function(x) {0.75 + x/5}
    part.worth.point.position <-
      lapply(part.worth.plotting.list,list.scali
    }
  if(max.is.less.than.2) {
    list.scaling <- function(x) {0.75 +
x/10}
    part.worth.point.position <-
      lapply(part.worth.plotting.list,list.scali
    }
  if(max.is.less.than.4) {
    list.scaling <- function(x) {0.75 +
x/20}
    part.worth.point.position <-

```

```

    lapply(part.worth.plotting.list,list.scali
  }

  if(max.is.less.than.10) {
    list.scaling <- function(x) {0.75 +
x/50}
    part.worth.point.position <-
      lapply(part.worth.plotting.list,list.scali
  }

  if(max.is.less.than.20) {
    list.scaling <- function(x) {0.75 +
x/100}
    part.worth.point.position <-
      lapply(part.worth.plotting.list,list.scali
  }

  if(max.is.less.than.40) {
    list.scaling <- function(x) {0.75 +
x/200}
    part.worth.point.position <-
      lapply(part.worth.plotting.list,list.scali
  }

  part.worth.point.position <-
lapply(part.worth.plotting.list,list.scaling)
}

if (plot.framing.box)
plot(c(0,0,1,1),c(0,1,0,1),xlab="",ylab"",
  type="n",xaxt="n",yaxt="n")

if (!plot.framing.box)
plot(c(0,0,1,1),c(0,1,0,1),xlab="",ylab"",
  type="n",xaxt="n",yaxt="n", bty="n")

if (put.title.on.spine.chart) {
  text(c(0.50),c(0.975),pos=3,labels=title.on.sp
  y.location <- 0.925 # starting position
with title

```

```

}

if (!put.title.on.spine.chart) y.location <-
0.975 # no-title start

# store top of vertical line for later
plotting needs
y.top.of.vertical.line <- y.location

x.center.position <- 0.75 # horizontal
position of spine

# begin primary plotting loop
# think of a plot as a collection of text
and symbols on screen or paper
# we are going to construct a plot one text
string and symbol at a time
# (note that we may have to repeat this
process at the end of the program)
for(k in seq(along=effect.names)) {
  y.location <- y.location - large.space
  text(c(0.4),c(y.location),pos=2,
    labels=paste(effect.name.map(effect.names[k]
",sep=""),cex=01.0)
  text(c(0.525),c(y.location),pos=2,col=color.fo
  labels=paste(
",left.side.symbol.to.print.around.importance,
  pretty.print(
    unlist(conjoint.results$attribute.importance
    right.side.symbol.to.print.around.importance

# begin loop for printing part-worths
  for(m in
seq(1:number.of.levels.of.attribute[k])) {
    y.location <- y.location - medium.space
    text(c(0.4),c(y.location),pos=2,
      conjoint.results$xlevel[[effect.names[k]]][m
    # part.worth.label.data.frame[k,m],cex=01.0)

    text(c(0.525),c(y.location),pos=2,

```

```

    col=color.for.printing.part.worth.text,
    labels= paste(
",left.side.symbol.to.print.around.part.worths,
pretty.print(part.worth.plotting.list[[effec
right.side.symbol.to.print.around.part.worth

points(part.worth.point.position[[effect.nam
type = "p", pch = 20, col =
color.for.part.worth.point, cex = 2)
segments(x.center.position, y.location,
part.worth.point.position[[effect.names[k]]]
y.location,
col = color.for.part.worth.line, lty
= 1, lwd = 2)
}
}

y.location <- y.location - medium.space

# begin center axis and bottom plotting
y.bottom.of.vertical.line <- y.location # 
store top of vertical line

below.y.bottom.of.vertical.line <-
y.bottom.of.vertical.line - small.space/2

if (!draw.gray.background) {
# four optional grid lines may be drawn on
the plot parallel to the spine
if (draw.optional.grid.lines) {
segments(0.55, y.top.of.vertical.line,
0.55,
y.bottom.of.vertical.line, col =
"black", lty = "solid", lwd = 1)

segments(0.65, y.top.of.vertical.line,
0.65,
y.bottom.of.vertical.line, col =
"gray", lty = "solid", lwd = 1)
}
}

```

```
    segments(0.85, y.top.of.vertical.line,
0.85,
            y.bottom.of.vertical.line, col =
"gray", lty = "solid", lwd = 1)

    segments(0.95, y.top.of.vertical.line,
0.95,
            y.bottom.of.vertical.line, col =
"black", lty = "solid", lwd = 1)
        }
    }

# gray background for plotting area of the
points
if (draw.gray.background) {
    rect(xleft = 0.55, ybottom =
y.bottom.of.vertical.line,
        xright = 0.95, ytop =
y.top.of.vertical.line, density = -1, angle =
45,
        col = "light gray", border = NULL, lty =
"solid", lwd = 1)

# four optional grid lines may be drawn on
the plot parallel to the spine
if (draw.optional.grid.lines) {
    segments(0.55, y.top.of.vertical.line,
0.55,
            y.bottom.of.vertical.line, col =
"black", lty = "solid", lwd = 1)

    segments(0.65, y.top.of.vertical.line,
0.65,
            y.bottom.of.vertical.line, col =
"white", lty = "solid", lwd = 1)

    segments(0.85, y.top.of.vertical.line,
0.85,
            y.bottom.of.vertical.line, col =
"white", lty = "solid", lwd = 1)
```

```
    segments(0.95, y.top.of.vertical.line,
0.95,
        y.bottom.of.vertical.line, col =
"black", lty = "solid", lwd = 1)
    }
}
# draw the all-important spine on the plot
segments(x.center.position,
y.top.of.vertical.line, x.center.position,
y.bottom.of.vertical.line, col = "black",
lty = "dashed", lwd = 1)

# horizontal line at top
segments(0.55, y.top.of.vertical.line, 0.95,
y.top.of.vertical.line,
col = "black", lty = 1, lwd = 1)

# horizontal line at bottom
segments(0.55, y.bottom.of.vertical.line,
0.95, y.bottom.of.vertical.line,
col = "black", lty = 1, lwd = 1)

# plot for ticks and labels
segments(0.55, y.bottom.of.vertical.line,
0.55, below.y.bottom.of.vertical.line,
col = "black", lty = 1, lwd = 1) # tick
line at bottom

segments(0.65, y.bottom.of.vertical.line,
0.65, below.y.bottom.of.vertical.line,
col = "black", lty = 1, lwd = 1) # tick
line at bottom

segments(0.75, y.bottom.of.vertical.line,
0.75, below.y.bottom.of.vertical.line,
col = "black", lty = 1, lwd = 1) # tick
line at bottom

segments(0.85, y.bottom.of.vertical.line,
0.85, below.y.bottom.of.vertical.line,
```

```
    col = "black", lty = 1, lwd = 1) # tick
line at bottom

segments(0.95, y.bottom.of.vertical.line,
0.95, below.y.bottom.of.vertical.line,
col = "black", lty = 1, lwd = 1) # tick
line at bottom

# axis labels vary with the max.is.less.than
range being used
if (max.is.less.than.1)
text(c(0.55,0.65,0.75,0.85,0.95),
rep(below.y.bottom.of.vertical.line,times=5),
pos=1,labels=c("-1","-0.5","0","+0.5","+1"),cex=0)

if (max.is.less.than.2)
text(c(0.55,0.65,0.75,0.85,0.95),
rep(below.y.bottom.of.vertical.line,times=5),
pos=1,labels=c("-2","-1","0","+1","+2"),cex=0.

if (max.is.less.than.4)
text(c(0.55,0.65,0.75,0.85,0.95),
rep(below.y.bottom.of.vertical.line,times=5),
pos=1,labels=c("-4","-2","0","+2","+4"),cex=0.

if (max.is.less.than.10)
text(c(0.55,0.65,0.75,0.85,0.95),
rep(below.y.bottom.of.vertical.line,times=5),
pos=1,labels=c("-10","-5","0","+5","+10"),cex=0)

if (max.is.less.than.20)
text(c(0.55,0.65,0.75,0.85,0.95),
rep(below.y.bottom.of.vertical.line,times=5),
pos=1,labels=c("-20","-10","0","+10","+20"),cex=0)

if (max.is.less.than.40)
text(c(0.55,0.65,0.75,0.85,0.95),
rep(below.y.bottom.of.vertical.line,times=5),
pos=1,labels=c("-40","-20","0","+20","+40"),cex=0)
```

```

y.location <-
below.y.bottom.of.vertical.line - small.space

text(.75,y.location,pos=1,labels=c("Part-Worth")
cex=0.95)

y.location <-
below.y.bottom.of.vertical.line - small.space

text(0.75,y.location,pos=1,labels=c("Part-Worth"
cex=0.95)

if(print.internal.consistency) {
  y.location <- y.location - medium.space
  text(c(0.525),c(y.location),pos=2,labels=paste
consistency: ",
  pretty.print(conjoint.results$internal.consist
sep=""))
}

# if we have grid lines we may have plotted
over part-worth points
# if we have a gray background then we have
plotted over part-worth points
# so let us plot those all-important
part-worth points and lines once again
if(draw.gray.background ||
draw.optional.grid.lines) {
  y.location <- y.top.of.vertical.line # 
retreive the starting value

# repeat the primary plotting loop
for(k in seq(along=effect.names)) {
  y.location <- y.location - large.space
  text(c(0.4),c(y.location),pos=2,
  labels=paste(effect.name.map(effect.names[k]
",sep=""),cex=01.0)
  text(c(0.525),c(y.location),pos=2,col=color.fo
  labels=paste(
",left.side.symbol.to.print.around.importance,

```

```

pretty.print(
  unlist(conjoint.results$attribute.importance
    right.side.symbol.to.print.around.importance

# begin loop for printing part-worths
  for(m in
    seq(1:number.of.levels.of.attribute[k])) {
      y.location <- y.location -
medium.space
      text(c(0.4),c(y.location),pos=2,
        conjoint.results$xlevel[[effect.names[k]]]

      text(c(0.525),c(y.location),
        pos=2,col=color.for.printing.part.worth
        labels=paste(
",left.side.symbol.to.print.around.part.worths,
        pretty.print(part.worth.plotting.list[[
          right.side.symbol.to.print.around.part.

        points(part.worth.point.position[[effect.nam
          type = "p", pch = 20, col =
color.for.part.worth.point, cex = 2)
          segments(x.center.position, y.location,
            part.worth.point.position[[effect.names[k]]]
y.location,
          col = color.for.part.worth.line, lty
= 1, lwd = 2)
        }
      }
    }
  }

# user-defined function for plotting
descriptive attribute names
effect.name.map <- function(effect.name) {
  if(effect.name=="price") return("Ticket
Price")
  if(effect.name=="seating") return("Seating
Area")
  if(effect.name=="boxvip") return("Box/VIP")
}

```

```
if(effect.name=="frontrow") return("Front
Row")
if(effect.name=="promotion")
return("Promotion")
}
# read in conjoint survey profiles with
respondent ranks
conjoint.data.frame <-
read.csv("sporting_event_ranking.csv")

# set up sum contrasts for effects coding as
needed for conjoint analysis
options(contrasts=c("contr.sum","contr.poly"))
# main effects model specification
main.effects.model <-
{ranking ~ price + seating + boxvip +
frontrow + promotion}

# fit linear regression model using main
effects only (no interaction terms)
main.effects.model.fit <-
lm(main.effects.model,
data=conjoint.data.frame)
print(summary(main.effects.model.fit))

# save key list elements of the fitted model
as needed for conjoint measures
conjoint.results <-
main.effects.model.fit[c("contrasts","xlevels",""
conjoint.results$attributes <-
names(conjoint.results$contrasts)

# compute and store part-worths in the
conjoint.results list structure
part.worths <- conjoint.results$xlevels # 
list of same structure as xlevels
end.index.for.coefficient <- 1 # intitialize
skipping the intercept
part.worth.vector <- NULL # used for
accumulation of part worths
```

```

for(index.for.attribute in
  seq(along=conjoint.results$contrasts)) {
  nlevels <-
  length(unlist(conjoint.results$xlevels[index.for.a
    begin.index.for.coefficient <-
    end.index.for.coefficient + 1
    end.index.for.coefficient <-
    begin.index.for.coefficient + nlevels -2
    last.part.worth <-
    -sum(conjoint.results$coefficients[
      begin.index.for.coefficient:end.index.for.coef
      part.worths[index.for.attribute] <-
      list(as.numeric(c(conjoint.results$coefficient
        begin.index.for.coefficient:end.index.for.coef
        last.part.worth)))
      part.worth.vector <-
      c(part.worth.vector,unlist(part.worths[index.f
      }
      conjoint.results$part.worths <- part.worths

      # compute and store part-worth ranges for each
      attribute
      part.worth.ranges <-
      conjoint.results$contrasts
      for(index.for.attribute in
        seq(along=conjoint.results$contrasts))
        part.worth.ranges[index.for.attribute] <-
        dist(range(conjoint.results$part.worths[index.f
        conjoint.results$part.worth.ranges <-
        part.worth.ranges

        sum.part.worth.ranges <-
        sum(as.numeric(conjoint.results$part.worth.ranges))

        # compute and store importance values for each
        attribute
        attribute.importance <-
        conjoint.results$contrasts
        for(index.for.attribute in

```

```
seq(along=conjoint.results$contrasts))
attribute.importance[index.for.attribute] <-
(dist(range(conjoint.results$part.worths[index.f
sum.part.worth.ranges) * 100
conjoint.results$attribute.importance <-
attribute.importance

# data frame for ordering attribute names
attribute.name <-
names(conjoint.results$contrasts)
attribute.importance <-
as.numeric(attribute.importance)
temp.frame <-
data.frame(attribute.name,attribute.importance)
conjoint.results$ordered.attributes <-
as.character(temp.frame[sort.list(
temp.frame$attribute.importance,decreasing =
TRUE),"attribute.name"])

# respondent internal consistency added to
list structure
conjoint.results$internal.consistency <-
summary(main.effects.model.fit)$r.squared

# user-defined function for printing conjoint
measures
if (print.digits == 2)
pretty.print <- function(x)
{sprintf("%1.2f",round(x,digits = 2))}
if (print.digits == 3)
pretty.print <- function(x)
{sprintf("%1.3f",round(x,digits = 3))}

# report conjoint measures to console
# use pretty.print to provide nicely formated
output
for(k in
seq(along=conjoint.results$ordered.attributes))
{
cat("\n", "\n")
```

```

cat(conjoint.results$ordered.attributes[k],"Leve
",
  unlist(conjoint.results$xlevels[conjoint.results

  cat("\n"," Part-Worths:   ")
  cat(pretty.print(unlist(conjoint.results$part.wo
    [conjoint.results$ordered.attributes[k]])))

  cat("\n"," Attribute Importance:   ")
  cat(pretty.print(unlist(conjoint.results$attribu
    [conjoint.results$ordered.attributes[k]])))
}

# plotting of spine chart begins here
# all graphical output is routed to external
pdf file
pdf(file =
"fig_zsports_willingness_to_pay.pdf",
width=8.5, height=11)
spine.chart(conjoint.results)
dev.off() # close the graphics output device

```

Exhibit 6.4. Preferences for Sporting Events—Conjoint Analysis (Python)

[Click here to view code image](#)

```

# Consumer Preferences for Sporting
Events---Conjoint Analysis (Python)

# prepare for Python version 3x features and
functions
from __future__ import division,
print_function

# import packages for analysis and modeling
import pandas as pd # data frame operations
import numpy as np # arrays and math

```

```
functions
import statsmodels.api as sm # statistical
models (including regression)
import statsmodels.formula.api as smf # 
R-like model specification
from patsy.contrasts import Sum

# read in conjoint survey profiles with
respondent ranks
conjoint_data_frame =
pd.read_csv('sporting_event_ranking.csv')

# set up sum contrasts for effects coding as
needed for conjoint analysis
# using C(effect, Sum) notation within main
effects model specification
main_effects_model = 'ranking ~ C(price, Sum)
+ C(seating, Sum) + \
C(boxvip, Sum) + C(frontrow, Sum) +
C(promotion, Sum)'

# fit linear regression model using main
effects only (no interaction terms)
main_effects_model_fit = \
smf.ols(main_effects_model, data =
conjoint_data_frame).fit()
print(main_effects_model_fit.summary())
conjoint_attributes = ['price', 'seating',
'boxvip', 'frontrow', 'promotion']

# build part-worth information one attribute
at a time
level_name = []
part_worth = []
part_worth_range = []
end = 1 # initialize index for coefficient in
params
for item in conjoint_attributes:
    level_set = set(conjoint_data_frame[item])
    nlevels = len(level_set)
```

```

    level_name.append(list(sorted(list(level_set)))
begin = end
end = begin + nlevels - 1
new_part_worth =
list(main_effects_model_fit.params[begin:end])
new_part_worth.append((-1) *
sum(new_part_worth))
part_worth_range.append(max(new_part_worth)
- min(new_part_worth))
part_worth.append(new_part_worth)
# end set to begin next iteration

# compute attribute relative importance values
from ranges
attribute_importance = []
for item in part_worth_range:
    attribute_importance.append(round(100 *
(item / sum(part_worth_range)),2))
# user-defined dictionary for printing
descriptive attribute names
effect_name_dict = {'price' : 'Ticket Price',
\
'seating' : 'Seating Area','boxvip' :
'Box/VIP', \
'frontrow' : 'Front Row', 'promotion' :
'Promotion'}
# report conjoint measures to console
index = 0 # initialize for use in for-loop
for item in conjoint_attributes:
    print('\nAttribute:',
effect_name_dict[item])
    print('    Importance:',
attribute_importance[index])
    print('    Level Part-Worths')
    for level in
range(len(level_name[index])):
        print('        ',level_name[index][level],
part_worth[index][level])
    index = index + 1

```

7. Promoting Brands and Products

Hobbs: "Still dogging me, huh, Max?"

Max: "Yeah. End of the road, Hobbs."

Hobbs: "Want to hear what I think our chances are?"

Max: "So, you read my mind."

Hobbs: "That takes all of three seconds."

Max: "They come and they go, Hobbs. They come and they go. I'm guessing I'm going to be around here a lot longer than you or anyone else around here. I'm here to protect the game."

Hobbs: "Who's game?"

Max: "I do it by making or breaking the likes of you."

Hobbs: "Did you ever play ball, Max?"

Max: "No, never have. But I make it a little more fun to watch. You see, it happens today whether you're a goat or a hero. You're gonna make me a quick story. Hey, see you around."

—ROBERT REDFORD AS ROY HOBBS, AND ROBERT DUVAL AS
MAX MERCY IN *The Natural* (1984)

It was a Thursday night in July. I was thinking about going to the ballpark. The Los Angeles Dodgers were playing the Colorado Rockies, and I was supposed to get an Adrian Gonzalez bobblehead with my ticket. Although I was not excited about the bobblehead, seeing a ball game at Dodger Stadium sounded like great fun. In April and May the Dodgers' record had not been the best, but things were looking better by July. I wondered if bobbleheads would bring

additional fans to the park. Dodgers management may have been wondering the same thing, or perhaps making plans for Yasiel Puig bobbleheads.

Suppose we were working for the Dodgers and wanted to learn about promotions and their effect on attendance. We call this example *Bobbleheads and Dodger Dogs or Shaking Our Bobbleheads Yes and No*. The example draws on Major League Baseball data from the 2012 season.

Relevant data for Dodgers' home games are shown in [table 7.1](#). Dodger Stadium, with a capacity of 56,000, is the largest ballpark in the world. We can see that Dodger Stadium was filled to capacity only twice in 2012. There were two cap promotions and three shirt promotions in 2012, not enough to draw meaningful inferences. Fireworks were used thirteen times on Friday nights, and once on the Fourth of July. The eleven bobblehead promotions occurred on night games, six of those being Tuesday nights.

month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bobblehead
APR	10	56003	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	NO
APR	11	29729	Wednesday	Pirates	58	Cloudy	Night	NO	NO	NO	NO
APR	12	28325	Thursday	Pirates	57	Cloudy	Night	NO	NO	NO	NO
APR	13	31601	Friday	Padres	54	Cloudy	Night	NO	NO	YES	NO
APR	14	46549	Saturday	Padres	57	Cloudy	Night	NO	NO	NO	NO
APR	15	38359	Sunday	Padres	65	Clear	Day	NO	NO	NO	NO
APR	23	26375	Monday	Braves	60	Cloudy	Night	NO	NO	NO	NO
APR	24	44014	Tuesday	Braves	63	Cloudy	Night	NO	NO	NO	NO
APR	25	26345	Wednesday	Braves	61	Cloudy	Night	NO	NO	NO	NO
APR	27	44807	Friday	Nationals	66	Clear	Night	NO	NO	YES	NO
APR	28	54242	Saturday	Nationals	71	Clear	Night	NO	NO	NO	YES
APR	29	48753	Sunday	Nationals	74	Clear	Day	NO	YES	NO	NO
MAY	7	43713	Monday	Giants	67	Clear	Night	NO	NO	NO	NO
MAY	8	32799	Tuesday	Giants	75	Clear	Night	NO	NO	NO	NO
MAY	9	33993	Wednesday	Giants	71	Clear	Night	NO	NO	NO	NO
MAY	11	35591	Friday	Rockies	65	Clear	Night	NO	NO	YES	NO
MAY	12	33735	Saturday	Rockies	65	Clear	Night	NO	NO	NO	NO
MAY	13	49124	Sunday	Rockies	70	Clear	Day	NO	NO	NO	NO
MAY	14	24312	Monday	Snakes	67	Clear	Night	NO	NO	NO	NO
MAY	15	47077	Tuesday	Snakes	70	Clear	Night	NO	NO	NO	YES
MAY	18	40905	Friday	Cardinals	64	Clear	Night	NO	NO	YES	NO
MAY	19	39383	Saturday	Cardinals	67	Clear	Night	NO	NO	NO	NO
MAY	20	44005	Sunday	Cardinals	77	Clear	Night	NO	NO	NO	NO
MAY	25	36283	Friday	Astros	59	Cloudy	Night	NO	NO	YES	NO
MAY	26	3E561	Saturday	Astros	61	Cloudy	Night	NO	NO	NO	NO
MAY	27	33305	Sunday	Astros	70	Clear	Day	NO	NO	NO	NO
MAY	28	38015	Monday	Brewers	73	Clear	Night	NO	NO	NO	NO
MAY	29	51137	Tuesday	Brewers	74	Clear	Night	NO	NO	NO	YES
MAY	30	25509	Wednesday	Brewers	69	Clear	Night	NO	NO	NO	NO
MAY	31	26773	Thursday	Brewers	70	Clear	Night	NO	NO	NO	NO
JUN	11	50550	Monday	Angels	68	Clear	Night	NO	YES	NO	NO
JUN	12	55279	Tuesday	Angels	66	Cloudy	Night	NO	NO	NO	YES
JUN	13	43494	Wednesday	Angels	67	Clear	Night	NO	NO	NO	NO
JUN	15	40432	Friday	White Sox	67	Clear	Night	NO	NO	YES	NO
JUN	16	45210	Saturday	White Sox	68	Clear	Night	NO	NO	NO	NO
JUN	17	53504	Sunday	White Sox	74	Clear	Day	NO	NO	NO	NO
JUN	28	45005	Thursday	Mets	75	Clear	Night	NO	NO	NO	YES
JUN	29	49763	Friday	Mets	72	Clear	Night	NO	NO	YES	NO
JUN	30	44217	Saturday	Mets	78	Clear	Day	NO	NO	NO	NO

JUL	1	55359	Sunday	Mets	75	Clear	Night	NO	NO	NO	YES
JUL	2	34493	Monday	Reds	70	Clear	Night	NO	NO	NO	NO
JUL	3	33884	Tuesday	Reds	70	Cloudy	Night	YES	NO	NO	NO
JUL	4	53570	Wednesday	Reds	70	Clear	Night	NO	NO	YES	NO
JUL	13	43873	Friday	Padres	76	Cloudy	Night	NO	NO	YES	NO
JUL	14	54014	Saturday	Padres	75	Clear	Night	NO	NO	NO	YES
JUL	15	39715	Sunday	Padres	77	Clear	Day	NO	NO	NO	NO
JUL	16	32238	Monday	Phillies	67	Clear	Night	NO	NO	NO	NO
JUL	17	53495	Tuesday	Phillies	70	Clear	Night	NO	NO	NO	NO
JUL	18	33955	Wednesday	Phillies	80	Cloudy	Day	NO	NO	NO	NO
JUL	30	33180	Monday	Snakes	73	Clear	Night	NO	NO	NO	NO
JUL	31	52832	Tuesday	Snakes	75	Cloudy	Night	NO	NO	NO	YES
AUG	1	36595	Wednesday	Snakes	79	Clear	Day	NO	NO	NO	NO
AUG	3	43537	Friday	Cubs	73	Clear	Night	NO	NO	YES	NO
AUG	4	46588	Saturday	Cubs	73	Cloudy	Night	NO	NO	NO	NO
AUG	5	42495	Sunday	Cubs	83	Clear	Day	YES	NO	NO	NO
AUG	6	32659	Monday	Rockies	79	Clear	Night	NO	NO	NO	NO
AUG	7	55024	Tuesday	Rockies	80	Clear	Night	NO	NO	NO	YES
AUG	8	37081	Wednesday	Rockies	81	Clear	Night	NO	NO	NO	NO
AUG	20	30879	Monday	Giants	80	Clear	Night	NO	NO	NO	NO
AUG	21	56000	Tuesday	Giants	75	Clear	Night	NO	NO	NO	YES
AUG	22	40173	Wednesday	Giants	75	Clear	Night	NO	NO	NO	NO
AUG	24	39005	Friday	Marlins	71	Clear	Night	NO	NO	YES	NO
AUG	25	40284	Saturday	Marlins	70	Clear	Night	NO	NO	NO	NO
AUG	26	41907	Sunday	Marlins	81	Clear	Day	NO	NO	NO	NO
AUG	30	54621	Thursday	Snakes	80	Clear	Night	NO	NO	NO	YES
AUG	31	37622	Friday	Snakes	77	Clear	Night	NO	NO	YES	NO
SEP	1	35992	Saturday	Snakes	81	Clear	Night	NO	NO	NO	NO
SEP	2	31607	Sunday	Snakes	89	Clear	Day	NO	NO	NO	NO
SEP	3	33540	Monday	Padres	84	Cloudy	Night	NO	NO	NO	NO
SEP	4	40619	Tuesday	Padres	78	Clear	Night	NO	YES	NO	NO
SEP	5	50560	Wednesday	Padres	77	Cloudy	Night	NO	NO	NO	NO
SEP	13	43309	Thursday	Cardinals	80	Clear	Night	NO	NO	NO	NO
SEP	14	40167	Friday	Cardinals	85	Clear	Night	NO	NO	YES	NO
SEP	15	42449	Saturday	Cardinals	95	Clear	Night	NO	NO	NO	NO
SEP	16	35754	Sunday	Cardinals	86	Clear	Day	NO	NO	NO	NO
SEP	28	37133	Friday	Rockies	77	Clear	Night	NO	NO	YES	NO
SEP	29	40724	Saturday	Rockies	84	Cloudy	Night	NO	NO	NO	NO
SEP	30	35607	Sunday	Rockies	95	Clear	Day	NO	NO	NO	NO
OCT	1	33624	Monday	Giants	86	Clear	Night	NO	NO	NO	NO
OCT	2	42473	Tuesday	Giants	83	Clear	Night	NO	NO	NO	NO
OCT	3	34014	Wednesday	Giants	82	Cloudy	Night	NO	NO	NO	NO

Table 7.1. Bobbleheads and Dodger Dogs

Exploratory graphics help us find models that might work for predicting attendance and evaluating the effect of promotions on attendance. Figure 7.1 shows distributions of attendance across days of the week, and figure 7.2 shows attendance by month. Box plots like these reveal the overall values of the data, with the boxes covering the middle fifty percent or so of the distribution and with the center line representing the

median. The dotted lines or whiskers extend to more extreme values in the distribution.¹ By looking at the box plots, we can make comparisons by day and by month across the distributions of attendance.

¹ To determine the length of the whiskers, we first compute the interquartile range, which is the distance between the 25th percentile and the 75th percentile. The end-points of the whiskers are defined by what are called *adjacent values*. The upper whisker extends to the upper adjacent value, a point one-and-a-half times the interquartile range above the upper end of the box. Or, if the maximum value in the distribution is less than that, the upper whisker extends to that maximum value. We often think of outliers as being points outside the whiskers; these outlier points are plotted as open circles. Box plots were the invention of John Tukey (1977).

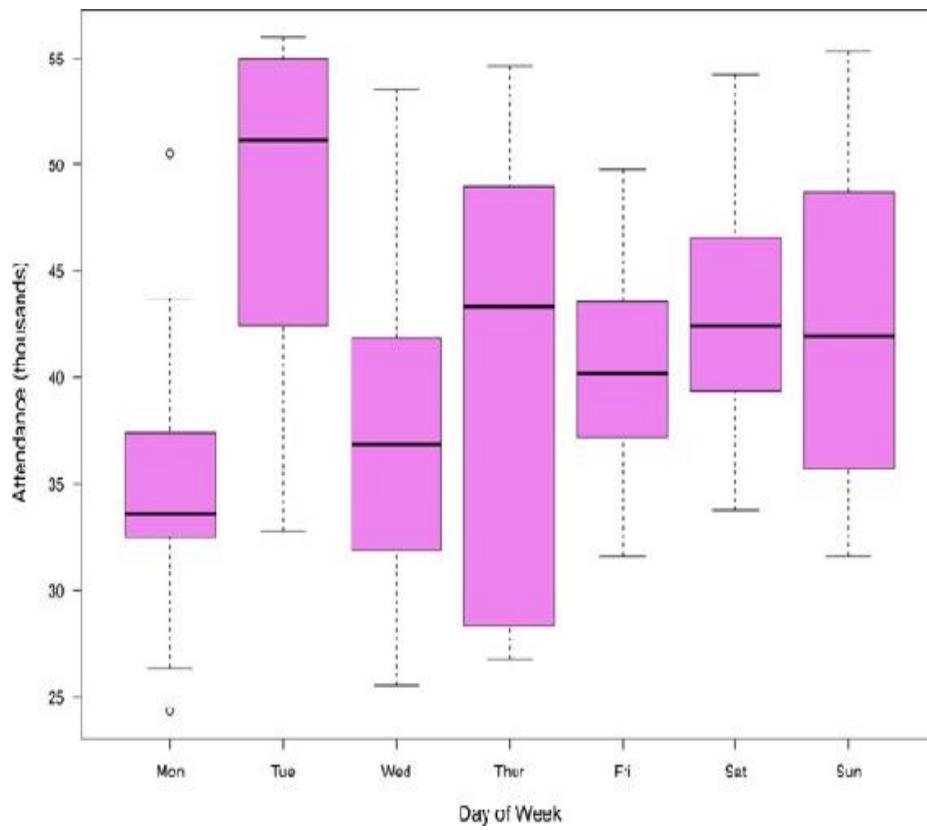


Figure 7.1. Dodgers Attendance by Day of Week

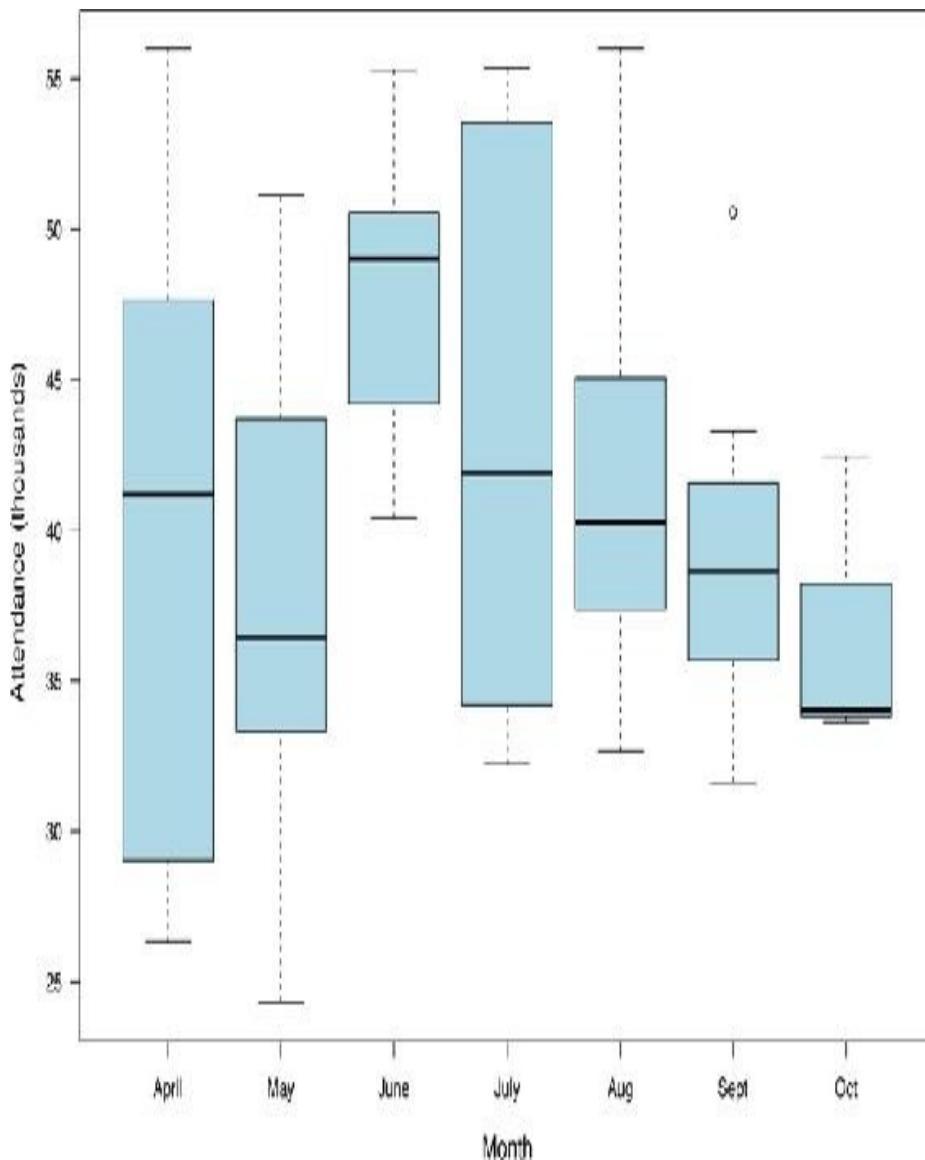


Figure 7.2. Dodgers Attendance by Month

We can explore these data further using a lattice of scatter plots. In figure 7.3 we map the relationship between temperature and attendance, controlling for time of game (day or night) and clear or cloudy skies. On day games with clear skies, we see what appears to be a moderate inverse relationship between temperature and attendance. Day games are usually on Sunday, and in 2012 all but one of those games was played under clear skies—a benefit of being in Los

Angeles.

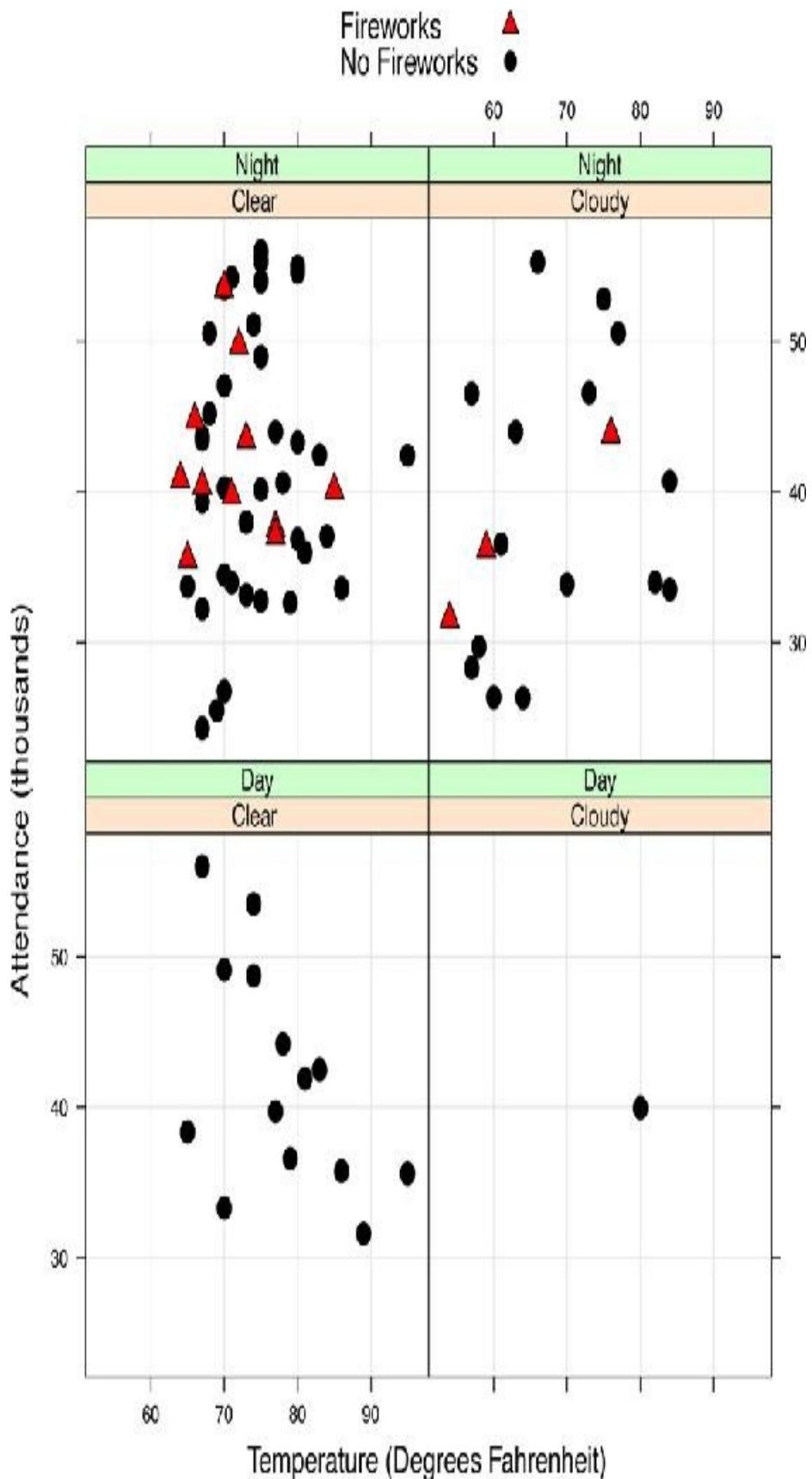


Figure 7.3. *Dodgers Weather, Fireworks, and Attendance*

More telling perhaps are strip plots of attendance by opponent or visiting team; these are the univariate scatter plots in figure 7.4. Opponents from the large metropolitan areas (the New York Mets, Chicago Cubs and White Sox, Los Angeles Angels, and Washington D.C. Nationals) are consistently associated with higher attendance. But there are seventeen visiting teams in this study, and only eighty-one games or observations. Accordingly, utilizing the visiting team as a categorical predictor presents problems.

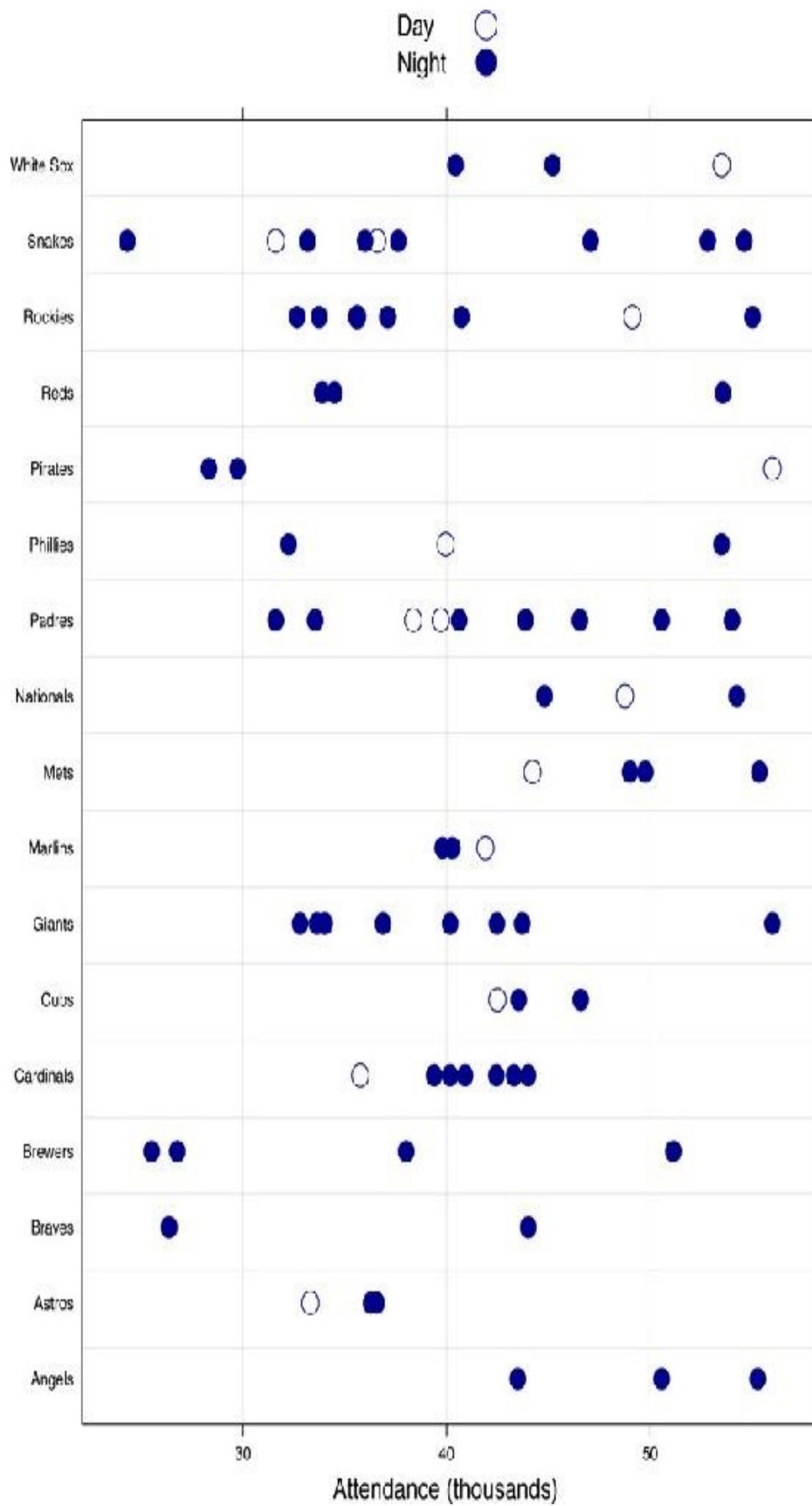


Figure 7.4. *Dodgers Attendance by Visiting Team*

To advise management regarding promotions, we would like to know if promotions have a positive effect on attendance, and if they do have a positive effect, how much might that effect be. So we build a linear model for predicting attendance using month, day of the week, and an indicator variable for the bobblehead promotion, and then we see how well it works. We enter these explanatory variables in a particular order so we can answer the basic question: “Do bobblehead promotions increase attendance, controlling for the date of the game (month and day of the week)?” Being data scientists, we employ a training-and-test regimen to provide an honest evaluation of the model’s predictive performance.

For the Los Angeles Dodgers bobblehead promotion, the fitted model does a good job of predicting higher attendance at games when bobbleheads are distributed. Our computer programs can provide indices of goodness of fit, but, more importantly, they can provide predictions of attendance that we can display on scatter plots.

How does a training-and-test regimen play out for the model we have developed for the Dodgers? Figure 7.5 provides a picture of model performance that data scientists and business managers can understand. The model fit to the training set holds up when used with the test set.

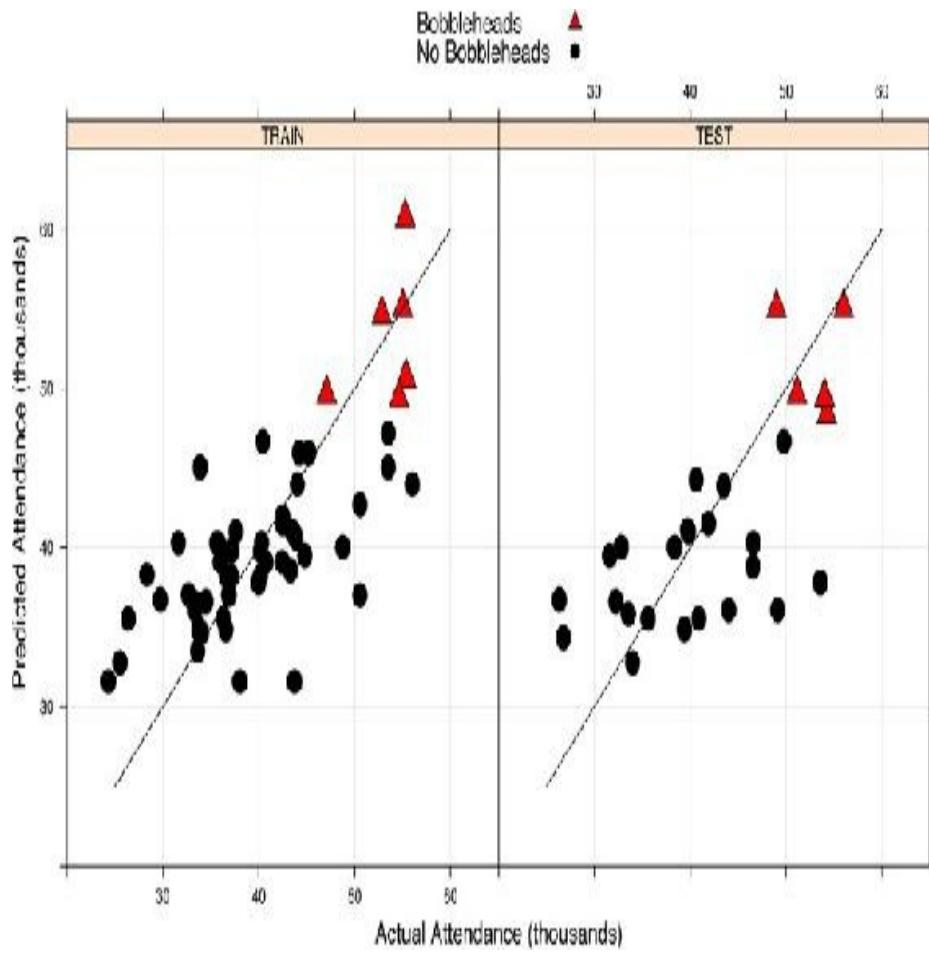


Figure 7.5. Regression Model Performance: Bobbleheads and Attendance

In figure 7.5, TRAIN refers to the training set, the data on which we fit the model and TEST refers to the hold-out-data on which we test our model. Running the code for this example, we would see that, in the test set, more than 45 percent of the variance in attendance is accounted for by the linear model—this is the square of the correlation of observed and predicted attendance. To explain a model to management, however, it is better to show a performance graph than to talk about squared correlation coefficients, mean squared errors of prediction, or other model summary statistics. This is one graph among many possible graphs that we could have

produced for the Dodgers. It shows the results of one particular random splitting of the data into training and test. Running the code for the study on the complete set of home game data for the Los Angeles Dodgers in 2012 would yield a set of regression coefficients, estimates of the parameters in the linear model, as shown in [table 7.2](#). A sequential analysis of variance shows a statistically significant effect for the bobblehead promotion, controlling for month and day of the week. A test of residuals from the model would identify any statistically significant outliers—there were none for this problem. Most importantly, the model can provide an assessment of the effect of the bobblehead promotion. In particular, we can see that bobblehead promotions have the potential of increasing attendance by 10,715 fans per game, all other things being equal.

Response: Attendance	
Month (May)	-2,385.625
Month (June)	7,163.234**
Month (July)	2,849.828
Month (August)	2,377.924
Month (September)	29.030
Month (October)	-662.668
Day of Week (Tuesday)	7,911.494***
Day of Week (Wednesday)	2,460.023
Day of Week (Thursday)	775.364
Day of Week (Friday)	4,883.818*
Day of Week (Saturday)	6,372.056**
Day of Week (Sunday)	6,724.003***
Bobblehead Promotion (YES)	10,714.900***
Constant	33,909.160***
Observations	81
R ²	0.544
Adjusted R ²	0.456
Residual Std. Error	6,120.158(<i>df</i> = 67)
F statistic	6.158***(<i>df</i> = 13; 67)

Notes:

***Significant at the 1 percent level.
 **Significant at the 5 percent level.
 *Significant at the 10 percent level.

Table 7.2. Regression of Attendance on Month, Day of Week, and Bobblehead Promotion

The baseball promotions example was chosen to be simple in structure, so that ordinary least squares regression could be

employed. This is a cross-sectional study with the baseball game serving as the unit of analysis.

More complicated models are possible, and diagnostic plots can provide additional information for the data scientist seeking to improve the model we specified. Nonetheless, it is interesting to note how much information a linear regression model can provide. Predictive models like the one used in this small example, with results presented in graphical summaries, can help guide management decisions.

One of the things that distinguishes data science from statistics is its focus on business requirements. In evaluating the utility of a model, the data scientist considers financial criteria as well as statistical criteria. And, in presenting predictions to management, she provides financial analysis as well as a description of the statistical model itself.

Using the fitted predictive model for the Dodgers bobblehead promotion, we can predict the attendance for each game in the forthcoming season, and we can predict this attendance with and without a bobblehead promotion. Knowing fixed and variable costs associated with a bobblehead promotion, as well as expected revenues from ticket sales and concessions, we can help the Los Angeles Dodgers assess the financial contribution of bobblehead promotions.

Considering costs for the forthcoming season in this example, the unit cost of a bobblehead doll is expected to be no more than \$3 when ordered in quantities of at least 20,000.

Bobbleheads are provided to the first 50,000 fans entering Dodger Stadium. To complete our work, then, we would use cost/volume/profit analysis to assess profit contribution.²

² Cost/volume/profit analysis is a common technique in management accounting.

It is sometimes called break even analysis or cost/benefit analysis. There are challenges in carrying out a financial analysis for Dodgers' promotions because ticket prices vary by the type of game and seating location. Ticket prices for (four-star) bobblehead games in 2013 varied from \$20 for a top deck seat to \$120 for a VIP field box seat. These prices were raised in 2014 to \$25 for a top deck seat to \$140 for a VIP field box seat. A portion of ticket revenues goes to support concessions and additional staff needed to distribute bobbleheads. We would obtain these cost estimates from Dodger management.

Among Major League Baseball teams, the Dodgers had the third highest attendance in 2012 and the highest attendance in 2013 and 2014 ([Knight 2015](#)). The Dodgers are one of many MLB teams using promotions to increase attendance. Reports suggest that bobblehead promotions in particular are on the rise, with 2.27 million dolls distributed in 2012 ([Broughton 2012](#)) and an estimated 2.7 million dolls in 2013 ([Foster 2013](#)).

Promotions such as the bobblehead promotion do more than simply drive up attendance. They also reinforce the name of the brand in the minds of consumers. Advertising and promotion are the “promotion” part of the marketing mix or *the four Ps*: product, price, promotion, and place. Here “product” relates to product or service, and “price” is simply price. The word “place” refers to channels of distribution (face-to-face selling, wholesale, retail, brick-and-mortar, mail-order, online, or mobile). Advertising and promotion are thought of as distinct fields of study by marketing academics. Advertising refers to the message, the marketing communication, while promotion is what firms do in addition to the message.

As we have shown, traditional regression models are especially relevant to these areas of inquiry. Useful sources for regression modeling include Kutner, Nachtsheim, Neter, and

Li (2004), Ryan (2008), and Chatterjee and Hadi (2012). For guidance in R programming for regression, see Venables and Ripley (2002), Fox and Weisberg (2011), and Fox (2014).

Moving beyond traditional linear models, we can consider modern, data-adaptive regression methods, as reviewed by Izenman (2008) and Hastie, Tibshirani, and Friedman (2009). We provide additional discussion of traditional and data-adaptive (machine learning) methods in [appendix A](#).

For an overview of advertising and promotion, marketing management textbooks may be consulted (Dickson 1997; Kotler and Keller 2012). Market response models attempt to predict sales and market shares across products within categories. These build upon econometric and time series methods. Hanssens, Parsons, and Schultz (2001) discuss market response modeling. Lilien and Rangaswamy (2003) suggest applications of market response modeling in sales force and channel management.

For those wishing to explore models across all Major League Baseball teams, complete promotion and attendance data for all teams for the 2012 season are provided on the publisher’s website <http://www.ftpress.com/miller/> and on GitHub at <https://github.com/mtpa/>.³ These data have a format similar to the Dodgers data in [table 7.1](#), except that there are extra columns for the year and home team.

Having data for all teams allows us to explore alternative modeling approaches, such as building a model for each team, aggregate models for groups of teams, or hierarchical models for game-day observations within teams. When predicting attendance at Major League Baseball parks, we would need to consider the fact that ballparks are often filled to capacity.

Special models may be required to accommodate this high-end censoring (Lemke, Leonard, and Tlhokwane 2010).

³ Major League Baseball data for promotions and attendance during the 2012 season were collected by Erica Costello. She graciously contributed these data so others could learn from them.

The R program *Shaking Our Bobbleheads Yes and No* is shown in [exhibit 7.1](#) and draws on packages for regression and graphics by Fox (2014) and Sarkar (2014), respectively. A similar Python program is shown in [exhibit 7.2](#).

Exhibit 7.1. Shaking Our Bobbleheads Yes and No (R)

[Click here to view code image](#)

```
# Predictive Model for Los Angeles Dodgers  
# Promotion and Attendance (R)  
  
library(car) # special functions for linear  
# regression  
library(lattice) # graphics package  
  
# read in data and create a data frame called  
# dodgers  
dodgers <- read.csv("dodgers.csv")  
print(str(dodgers)) # check the structure of  
# the data frame  
  
# define an ordered day-of-week variable  
# for plots and data summaries  
dodgers$ordered_day_of_week <-  
with(data=dodgers,  
  ifelse ((day_of_week == "Monday"),1,  
  ifelse ((day_of_week == "Tuesday"),2,  
  ifelse ((day_of_week == "Wednesday"),3,  
  ifelse ((day_of_week == "Thursday"),4,  
  ifelse ((day_of_week == "Friday"),5,  
  ifelse ((day_of_week ==  
  "Saturday"),6,7))))))
```

```
dodgers$ordered_day_of_week <-
  factor(dodgers$ordered_day_of_week,
  levels=1:7,
  labels=c("Mon", "Tue", "Wed", "Thur", "Fri",
  "Sat", "Sun"))

# exploratory data analysis with standard
graphics: attendance by day of week
with(data=dodgers,plot(ordered_day_of_week,
attend/1000,
xlab = "Day of Week", ylab = "Attendance
(thousands)",
col = "violet", las = 1))

# when do the Dodgers use bobblehead
promotions
with(dodgers,
table(bobblehead,ordered_day_of_week)) #  
bobbleheads on Tuesday

# define an ordered month variable
# for plots and data summaries
dodgers$ordered_month <- with(data=dodgers,
  ifelse ((month == "APR"),4,
  ifelse ((month == "MAY"),5,
  ifelse ((month == "JUN"),6,
  ifelse ((month == "JUL"),7,
  ifelse ((month == "AUG"),8,
  ifelse ((month == "SEP"),9,10)))))))

dodgers$ordered_month <-
  factor(dodgers$ordered_month, levels=4:10,
  labels = c("April", "May", "June", "July",
  "Aug", "Sept", "Oct"))

# exploratory data analysis with standard R
graphics: attendance by month
with(data=dodgers,plot(ordered_month,attend/1000,
xlab = "Month",
ylab = "Attendance (thousands)", col = "light
blue", las = 1))
```

```

# exploratory data analysis displaying many
variables
# looking at attendance and conditioning on
day/night
# the skies and whether or not fireworks are
displayed
library(lattice) # used for plotting
# let us prepare a graphical summary of the
dodgers data
group.labels <- c("No Fireworks","Fireworks")
group.symbols <- c(21,24)
group.colors <- c("black","black")
group.fill <- c("black","red")
xyplot(attend/1000 ~ temp | skies + day_night,
       data = dodgers, groups = fireworks, pch =
group.symbols,
       aspect = 1, cex = 1.5, col = group.colors,
fill = group.fill,
       layout = c(2, 2), type = c("p","g"),
       strip=strip.custom(strip.levels=TRUE,strip.nam
style=1),
       xlab = "Temperature (Degrees Fahrenheit)",
       ylab = "Attendance (thousands)",
       key = list(space = "top",
                  text = list(rev(group.labels),col =
rev(group.colors)),
       points = list(pch =
rev(group.symbols), col = rev(group.colors),
           fill = rev(group.fill))))
# attendance by opponent and day/night game
group.labels <- c("Day","Night")
group.symbols <- c(1,20)
group.symbols.size <- c(2,2.75)
bwplot(opponent ~ attend/1000, data = dodgers,
groups = day_night,
       xlab = "Attendance (thousands)",
       panel = function(x, y, groups, subscripts,
... )
       {panel.grid(h =

```

```
(length(levels(dodgers$opponent)) - 1), v =
-1)
    panel.stripplot(x, y, groups = groups,
subscripts = subscripts,
        cex = group.symbols.size, pch =
group.symbols, col = "darkblue")
    },
key = list(space = "top",
text = list(group.labels,col = "black"),
points = list(pch = group.symbols, cex =
group.symbols.size,
col = "darkblue")))

# employ training-and-test regimen for model
validation
set.seed(1234) # set seed for repeatability of
training-and-test split
training_test <-
c(rep(1,length=trunc((2/3)*nrow(dodgers))),
rep(2,length=(nrow(dodgers) -
trunc((2/3)*nrow(dodgers)))))

dodgers$training_test <- sample(training_test)
# random permutation
dodgers$training_test <-
factor(dodgers$training_test,
      levels=c(1,2), labels=c("TRAIN","TEST"))
dodgers.train <- subset(dodgers, training_test
== "TRAIN")
print(str(dodgers.train)) # check training
data frame
dodgers.test <- subset(dodgers, training_test
== "TEST")
print(str(dodgers.test)) # check test data
frame

# specify a simple model with bobblehead
entered last
my.model <- {attend ~ ordered_month +
ordered_day_of_week + bobblehead}
# fit the model to the training set
```

```
train.model.fit <- lm(my.model, data =
dodgers.train)
# summary of model fit to the training set
print(summary(train.model.fit))
# training set predictions from the model fit
# to the training set
dodgers.train$predict_attend <-
predict(train.model.fit)
# test set predictions from the model fit to
# the training set
dodgers.test$predict_attend <-
predict(train.model.fit,
newdata = dodgers.test)
# compute the proportion of response variance
# accounted for when predicting out-of-sample
cat("\n","Proportion of Test Set Variance
Accounted for: ",
round((with(dodgers.test,cor(attend,predict_attend
digits=3),"\n",sep=""))
# merge the training and test sets for
plotting
dodgers.plotting.frame <-
rbind(dodgers.train,dodgers.test)

# generate predictive modeling visual for
management
group.labels <- c("No
Bubbleheads","Bubbleheads")
group.symbols <- c(21,24)
group.colors <- c("black","black")
group.fill <- c("black","red")
xyplot(predict_attend/1000 ~ attend/1000 |
training_test,
      data = dodgers.plotting.frame, groups =
bubblehead, cex = 2,
      pch = group.symbols, col =
group.colors, fill = group.fill,
      layout = c(2, 1), xlim = c(20,65), ylim
= c(20,65),
      aspect=1, type = c("p","g"),
```

```
panel=function(x,y, ...)
  {panel.xyplot(x,y,...)
   panel.segments(25,25,60,60,col="black"
   },
  strip=function(...) strip.default(...,
style=1),
  xlab = "Actual Attendance (thousands)",
  ylab = "Predicted Attendance
(thousands)",
  key = list(space = "top",
  text =
list(rev(group.labels),col =
rev(group.colors)),
  points = list(pch =
rev(group.symbols),
  col = rev(group.colors),
  fill = rev(group.fill))))
# use the full data set to obtain an estimate
of the increase in
# attendance due to bobbleheads, controlling
for other factors
my.model.fit <- lm(my.model, data =
dodgers) # use all available data
print(summary(my.model.fit))
# tests statistical significance of the
bobblehead promotion
# type I anova computes sums of squares for
sequential tests
print(anova(my.model.fit))
cat("\n","Estimated Effect of Bobblehead
Promotion on Attendance: ",
round(my.model.fit$coefficients[length(my.model.fi
dges = 0),"\n",sep=""])
# standard graphics provide diagnostic plots
plot(my.model.fit)
# additional model diagnostics drawn from the
car package
library(car)
residualPlots(my.model.fit)
marginalModelPlots(my.model.fit)
```

```
print(outlierTest(my.model.fit))
```

Exhibit 7.2. Shaking Our Bobbleheads Yes and No (Python)

[Click here to view code image](#)

```
# Predictive Model for Los Angeles Dodgers  
Promotion and Attendance (Python)  
  
# prepare for Python version 3x features and  
functions  
from __future__ import division,  
print_function  
from future_builtins import ascii, filter,  
hex, map, oct, zip  
  
# import packages for analysis and modeling  
import pandas as pd # data frame operations  
from pandas.tools.rplot import RPlot,  
TrellisGrid, GeomPoint,\n    ScaleRandomColour # trellis/lattice  
plotting  
import numpy as np # arrays and math  
functions  
from scipy.stats import uniform # for  
training-and-test split  
import statsmodels.api as sm # statistical  
models (including regression)  
import statsmodels.formula.api as smf #  
R-like model specification  
import matplotlib.pyplot as plt # 2D plotting  
  
# read in Dodgers bobbleheads data and create  
data frame  
dodgers = pd.read_csv("dodgers.csv")  
# examine the structure of the data frame  
print("\nContents of dodgers data frame  
-----")
```

```
# attendance in thousands for plotting
dodgers['attend_000'] = dodgers['attend']/1000
# print the first five rows of the data frame
print(pd.DataFrame.head(dodgers))

mondays = dodgers[dodgers['day_of_week'] == 'Monday']
tuesdays = dodgers[dodgers['day_of_week'] == 'Tuesday']
wednesdays = dodgers[dodgers['day_of_week'] == 'Wednesday']
thursdays = dodgers[dodgers['day_of_week'] == 'Thursday']
fridays = dodgers[dodgers['day_of_week'] == 'Friday']
saturdays = dodgers[dodgers['day_of_week'] == 'Saturday']
sundays = dodgers[dodgers['day_of_week'] == 'Sunday']

# convert days' attendance into list of
# vectors for box plot
data = [mondays['attend_000'],
        tuesdays['attend_000'],
        wednesdays['attend_000'],
        thursdays['attend_000'],
        fridays['attend_000'],
        saturdays['attend_000'],
        sundays['attend_000']]
ordered_day_names = ['Mon', 'Tue', 'Wed',
                     'Thur', 'Fri', 'Sat', 'Sun']

# exploratory data analysis: box plot for day
# of the week
fig, axis = plt.subplots()
axis.set_xlabel('Day of Week')
axis.set_ylabel('Attendance (thousands)')
day_plot = plt.boxplot(data, sym='o', vert=1,
                       whiskerprops={ 'whis': 1.5 })
plt.setp(day_plot['boxes'], color = 'black')
```

```
plt.setp(day_plot['whiskers'], color =
'black')
plt.setp(day_plot['fliers'], color = 'black',
marker = 'o')
axis.set_xticklabels(ordered_day_names)
plt.show()
plt.savefig('fig_advert_promo_dodgers_eda_day_of_w
bbox_inches = 'tight', dpi=None,
facecolor='w', edgecolor='b',
orientation='portrait', papertype=None,
format=None,
transparent=True, pad_inches=0.25,
frameon=None)
april = dodgers[dodgers['month'] == 'APR']
may = dodgers[dodgers['month'] == 'MAY']
june = dodgers[dodgers['month'] == 'JUN']
july = dodgers[dodgers['month'] == 'JUL']
august = dodgers[dodgers['month'] == 'AUG']
september = dodgers[dodgers['month'] == 'SEP']
october = dodgers[dodgers['month'] == 'OCT']
data = [april['attend_000'],
may['attend_000'],
june['attend_000'], july['attend_000'],
august['attend_000'],
september['attend_000'],
october['attend_000']]
ordered_month_names = ['April', 'May', 'June',
'July', 'Aug', 'Sept', 'Oct']

fig, axis = plt.subplots()
axis.set_xlabel('Month')
axis.set_ylabel('Attendance (thousands)')
day_plot = plt.boxplot(data, sym='o', vert=1,
whis=1.5)
plt.setp(day_plot['boxes'], color = 'black')
plt.setp(day_plot['whiskers'], color =
'black')
plt.setp(day_plot['fliers'], color = 'black',
marker = 'o')
axis.set_xticklabels(ordered_month_names)
```

```
plt.show()
plt.savefig('fig_advert_promo_dodgers_eda_month_Py
    bbox_inches = 'tight', dpi=None,
facecolor='w', edgecolor='b',
    orientation='portrait', papertype=None,
format=None,
    transparent=True, pad_inches=0.25,
frameon=None)

# trellis/lattice plot attendance by temp,
conditioning on skies
# and day_night with bobblehead NO/YES shown
in distinct colors
plt.figure()
plot = RPlot(dodgers,    x = 'temp', y =
'attend_000')
plot.add(TrellisGrid(['day_night', 'skies']))
plot.add(GeomPoint(colour =
ScaleRandomColour('bobblehead')))
plot.render(plt.gcf())
plt.show()
plt.savefig('fig_advert_promo_dodgers_eda_many.pdf
    bbox_inches = 'tight', dpi=None,
facecolor='w', edgecolor='b',
    orientation='portrait', papertype=None,
format=None,
    transparent=True, pad_inches=0.25,
frameon=None)

# map day_of_week to ordered_day_of_week
day_to_ordered_day = {'Monday' : '1Monday',
'Tuesday' : '2Tuesday',
'Wednesday' : '3Wednesday',
'Thursday' : '4Thursday',
'Friday' : '5Friday',
'Saturday' : '6Saturday',
'Sunday' : '7Sunday'}
dodgers['ordered_day_of_week'] =
dodgers['day_of_week'].map(day_to_ordered_day)
```

```
# map month to ordered_month
month_to_ordered_month = {'APR' : '1April',
                           'MAY' : '2May',
                           'JUN' : '3June',
                           'JUL' : '4July',
                           'AUG' : '5Aug',
                           'SEP' : '6Sept',
                           'OCT' : '7Oct'}
dodgers['ordered_month'] =
dodgers['month'].map(month_to_ordered_month)

# employ training-and-test regimen for model
validation
np.random.seed(1234)
dodgers['runiform'] = uniform.rvs(loc = 0,
scale = 1, size = len(dodgers))
dodgers_train = dodgers[dodgers['runiform'] >=
0.33]
dodgers_test = dodgers[dodgers['runiform'] <
0.33]
# check training data frame
print('\ndodgers_train data frame (rows,
columns): ',dodgers_train.shape)
print(dodgers_train.head())
# check test data frame
print('\ndodgers_test data frame (rows,
columns): ',dodgers_test.shape)
print(dodgers_test.head())

# specify a simple model with bobblehead
entered last
my_model = str('attend ~ ordered_month +
ordered_day_of_week + bobblehead')

# fit the model to the training set
train_model_fit = smf.ols(my_model, data =
dodgers_train).fit()
# summary of model fit to the training set
print(train_model_fit.summary())
# training set predictions from the model fit
```

```
to the training set
dodgers_train['predict_attend'] =
train_model_fit.fittedvalues

# test set predictions from the model fit to
# the training set
dodgers_test['predict_attend'] =
train_model_fit.predict(dodgers_test)

# compute the proportion of response variance
# accounted for when predicting out-of-sample
print('\nProportion of Test Set Variance
Accounted for: ',\
      round(np.power(dodgers_test['attend'].\
      corr(dodgers_test['predict_attend']),2),3)))
# use the full data set to obtain an estimate
# of the increase in
# attendance due to bobbleheads, controlling
# for other factors
my_model_fit = smf.ols(my_model, data =
dodgers).fit()
print(my_model_fit.summary())
print('\nEstimated Effect of Bobblehead
Promotion on Attendance: ',\
      round(my_model_fit.params[13],0))
```

8. Growing Revenues

Roy: "I think of the golf swing as a poem. The critical opening phrase of this poem will always be the grip. The hands unite to form a single unit by the simple overlap of the . . . little finger."

Molly: "Right."

Roy: "Lowly and slowly the club head is led back, pulled into position not by the hands, but by the body, which turns away from the targets, shifting weight to the right side without shifting balance. Tempo is everything, perfection unattainable. As the body coils down at the top of the swing, there is a slight hesitation, a nod to the gods . . . It's alive, this swing, and a sculpture. And down through contact, striking the ball crisply with character. . . . And the unfinished business of Roy McAvoy."

Molly: "Why unfinished?"

Roy: "I have a short follow-through. It has an unfinished look."

Molly: "Why?"

Roy: "Well some say that's the easiest way to play in the winds of west Texas, and some say it's because I never finished anything in my life. You can decide. But the point is every finishing position is unique. . . . That's what the golf swing is all about. It's about gaining control of your life and letting go at the same time. . . . There's only one other acceptable theory about how to hit a golf ball."

Molly: "Oh, boy. I'm afraid to ask. Well, what's the other

theory?”

Roy: “Grip it and rip it.”

—KEVIN COSTNER AS ROY McAVOY AND RENE RUSSO AS DR.
MOLLY GRISWOLD IN *Tin Cup* (1996)

Professional team sports is called “the winning business” because there is often a relationship between winning and revenues. Sports consumers like winners. Winning means higher ticket sales. Winning means going into the playoffs and collecting revenue from additional games. Winning means it is easier to garner governmental support for facilities, recruit players, and find business partners, advertisers, and corporate sponsors. Winning is part of the revenue growth story, but not the whole story.

Rein, Shields, and Grossman (2015) lament “the winning fallacy” of sports organizations—the idea that winning is essential to sports business success. It is not. There are notable examples of teams that have been perennial losers, while beloved by their fans and successful as businesses. Witness the Boston Red Sox and Chicago Cubs throughout much of their history (Goodwin 1997; Will 2014). It is possible for teams to have brand equity, loyal fans, and full stadia without winning championships. Leagues also have expanded playoff schedules so more teams are winners.

Equally or perhaps more important than winning are marketing strategy and tactics. As in any business, there is work to do finding new customers and retaining current customers, identifying and targeting market segments, building brand loyalty, developing products, and defining effective pricing plans. Data science can contribute to all of

these endeavors.

No business can long survive without finding new customers, and the process of finding new customers begins by learning as much as we can about current customers and groups of customers. We identify the types of customers who are most likely to buy as well as variables that may be used to find those customers.

A market segment is a group of consumers that is different from other groups of consumers in ways that matter to marketing managers. We can identify segments by geographic, demographic, psychographic, and behavioral characteristics. Most useful for target marketing are characteristics that are accessible and easily measured.

When properly executed, market segmentation can guide marketing strategy and tactics. What consumers like, what they buy, where they buy, and how much they buy may differ across segments. Some consumers are responsive to advertisements and promotions. Some are more price-conscious than others. By using knowledge of a consumer's market segment, marketing managers can make informed decisions about marketing actions. They can use market segments in product development, advertising, promotion, pricing, and target marketing.

To be useful to marketing management, market segments must be discoverable and reachable. We must be able to identify a consumer's segment from available data, find our way to that consumer, and take appropriate marketing action. This is the core of sales and marketing—converting a consumer with little knowledge of a product into a buyer of the product.

Variables that go into the segmentation should be easily

available or accessible. We avoid variables that are difficult to measure. And we prefer publicly available data. Common segmentation variables reflect geographical location, age, income, and behavioral or lifestyle variables. We look for data that we can gather without survey sampling.

Data for market segmentation are many and varied.

Geographic data refer to where people live, such as the region of the country, state, city, street location, or census area.

Behavioral data include occupational information, how consumers travel, where they shop, what they read, and to which community groups they belong. Demographic data include variables such as gender, age, education, occupation, and level of income. Psychographic data include psychological factors, attitudes, interests, personality characteristics, and lifestyle factors. Of special importance in sports consumer segmentation is information about sports participation.

Age is a common segmentation variable. Many marketing managers think in terms of generations by date of birth. It is common to distinguish among members of the silent generation (birth range 1925–1945), baby boomers (1946–1964), Generation Xers (1965–1978), and Millennials (also called Generation Y, 1979–1994)). Consumers in generational segments are understood to have diverse needs, attitudes, and behaviors. Generational differences can affect consumer response to marketing messages, products, promotions, and pricing. Millennials, for example, can be expected to be less price-sensitive and more technologically savvy (Kotler and Keller 2012).

Marketing managers often design products and tailor marketing messages for individual segments or combinations

of segments. Knowing that Millennials are technology-savvy, for example, may prompt managers to advertise products on computers and mobile devices to reach that segment.

Or they may emphasize online sales promotion as opposed to brick-and-mortar promotion.

Traditional methods of cluster analysis are widely used in market segmentation. They represent multivariate techniques for grouping consumers based on their similarity to one another. Distance metrics or measures of agreement between consumers guide the segmentation process. Cluster analysis involves finding groups in data. Cluster analysis takes many consumer-related variables and uses them to represent differences between consumers. When two consumers have similar values for such variables as age, marital status, job type, and education, they are seen as being similar. When two consumers have very different values on these variables, they are seen as dissimilar. Cluster analysis looks at the differences among consumers—their distances from one another—to identify groups of consumers or market segments.

In most cases, we would perform many cluster analyses, providing a set of clustering solutions. We can begin by seeing what happens when we divide the entire set of consumers into just two groups or clusters. Then we can try three clusters, then four, and so on. The number of clusters is an important feature of a clustering solution. If there are too many clusters, the clustering solution may be difficult to utilize in product marketing.

It is common to look for solutions involving ten or fewer clusters. We also seek solutions for which all clusters or segments are of sufficient size to warrant marketing attention.

We note the proportion of consumers in each cluster when evaluating solutions. Clusters that are very small may be of little interest to management. The fewer consumers in a cluster, the lower the potential sales revenue.

Selecting a clustering solution among the many possible solutions we obtain from cluster analysis algorithms is as much art as science. There are numerous factors to consider in the context of market segmentation. We look for segments that are easy to interpret in terms of descriptive statistics. After segments have been identified, we can return to the full database of customer or client information to see if there are relationships between segment membership and sales order history and/or responsiveness to sales and marketing activities.

Mass marketing treats all customers as one group. One-to-one marketing focuses on one customer at a time. Target marketing to selected groups of customers or market segments lies between mass marketing and one-to-one marketing. Target marketing involves directing marketing activities to those customers who are most likely to buy.

Market segmentation and target marketing often go hand-in-hand. But there are people in marketing who are philosophically opposed to using segments for targeting. Instead of targeting market segments, these researchers promote a one-to-one marketing approach that targets each individual as an individual.

Targeting involves identifying customers who are most likely to buy products and directing marketing efforts toward those customers. The customers are given more attention in terms of pre-sales activities and post-sales support. Targeting implies selection. Some customers are identified as more valuable than

others and these more highly valued customers are given special attention. By becoming skilled at targeting, a company can improve its profitability, increasing revenues and decreasing costs.

Targeting is best executed by organizations that keep detailed records for individuals. These are companies that offer loyalty programs or use a customer relationship management system. Sales transactions need to be associated with specific customers and stored in a database. Where revenues (cash inflows) and expenses (cash outflows) are recorded, we can carry out discounted cash-flow analysis and compute the return on investment for each customer.

A target is a customer who is worth pursuing. A target is a profitable customer—sales revenues from the target exceed costs of sales and support. Another way to say this is that a target is a customer with positive lifetime value. Over the course of a company's relationship with the customer, more money comes into the business than goes out of the business.

In target marketing, we need to identify factors that are useful and determine how to use those factors in modeling techniques. A response variable is something we want to predict, such as sales dollars, volume, or whether a consumer will buy a product. Customer lifetime value is a composite response variable, computed from many transactions with each customer, and these transactions include observations of sales and costs.

Explanatory variables are used to predict response variables. Explanatory variables can be continuous (having meaningful magnitude) or categorical (without meaningful magnitude). Statistical models show the relationship between explanatory

variables and response variables.

Regression and classification can both be used in target marketing. When the response variable (the variable to be predicted) is continuous or has meaningful magnitude, we use regression to make the prediction. Examples of response variables with meaningful magnitude are sales dollars, sales volume, cost of sales, cost of support, and customer lifetime value.

When the response variable is categorical (a variable without meaningful magnitude), we use classification. Examples of response variables without meaningful magnitude are whether a customer buys, whether a customer stays with the company or leaves to buy from another company, and whether the customer recommends a company's products to another customer.

To develop a classification model for targeting, we proceed in much the same way as with regression, except the response variable is now a category or class. For each customer, a logistic regression model can provide a predicted probability of response. We classify responses using a cut-off value for the probability of response. If the cut-off were set at 0.50, for example, then we would target the customer if the predicted probability of response is greater than 0.50, and not target otherwise. Or we could target all customers who have a predicted probability of response of 0.40, or 0.30, and so on. The value of the cut-off will vary from one problem to the next.

When we engage in target marketing, we review data from current customers, particularly sales transaction data. We also assess the costs of sales and support for current customers. We

can think of each customer as an investment, and compute the return on investment for each customer. If the expected lifetime value of a customer is positive, then it makes sense to retain that customer.

Customer lifetime value analysis draws on concepts from financial management. We evaluate investments in terms of cash inflows and outflows over time. Before we pursue a prospective customer, we want to know that discounted cash inflows (sales) will exceed discounted cash outflows (costs). It makes sense to retain a current customer when discounted future cash flows are positive.

Customer lifetime value is computed from our experience with each customer. For the cash inflows, we note a customer's purchasing history as recorded in sales transactions. For the cash outflows, we note past sales and support costs as recorded in customer relationship management systems. Customer lifetime value analysis is best executed when detailed records are maintained for customers.

Data for valuing customers may be organized as panel or longitudinal data. Rows correspond to customers and columns correspond to time periods. Data from past transactions may be incomplete, and future cash-flows are unknown. So, we use predictive models to estimate cash-flows. We draw on available data to impute missing observations from the past, and we use observations from the past to forecast observations in the future.

Direct marketers are the quintessential target marketers. Their work involves contacting prospective and former customers directly through telephone, mail, e-mail, and online channels. Direct marketers collect and maintain information about past

contacts, mailings, incoming and outgoing communications, and business transactions. And they do this on a customer-by-customer basis. These data are used to guide sales promotions and direct marketing programs. Direct mailings and outgoing communications include product brochures and announcements, as well as coupons and information about product prices, bundles, and promotions.

Each direct marketing promotion may be evaluated in terms of its contribution to the profit of the firm. There are costs associated with mailings and online activities. There are revenues coming from people who order. The hit rate or proportion of people who respond to a direct mail or online offer is a critical number to watch because it determines the success or failure of the promotion. Direct marketing promotions, properly constructed, represent field experiments. Rarely is it wise to mail to an entire list at once.

When conducting a direct marketing experiment, we can divide the list into sections and vary the direct mail offer or advertising copy across sections. The conditions that yield the highest profit on the test mailings set the stage for subsequent mailings. Numerous treatment conditions can be examined for each direct marketing promotion.

Direct and database marketers build models for predicting who will buy in response to marketing promotions. Traditional models, or what are known as *RFM models*, consider the recency (date of most recent purchase), frequency (number of purchases), and monetary value (sales revenue) of previous purchases. More complicated models utilize a variety of explanatory variables relating to recency, frequency, monetary value, and customer demographics.

Pricing research presents special challenges. There are many questions to answer. What are consumers willing to pay for a product or service? What do they expect to pay? How will prices across a set of competitive or substitute products affect consumer choice? What prices will generate the highest contribution to profit?

In setting prices, sports businesses need to consider what are called the *three C's of pricing*: costs, consumers, and the competition. “Consumers” in this context refers to willingness to pay and demand, which we cover here. And “competition” refers to substitute products, not sports competition in games. To consider an application in sports management, we can look at willingness to pay for tickets.

Ticket prices deserve special mention due to their importance in contributing to team revenue. Do costs, particularly player costs, drive ticket prices? Probably not. Fort and Winfree (2013) dispel the notion that player salary demands cause ticket prices to rise. Rather, it is consumer demand that engenders both higher ticket prices and higher player salaries. Many teams experience sellouts on a regular bases, suggesting that they could set higher ticket prices. Instead of asking why ticket prices are high, we should be asking why they are not higher.

To do a good job of defining ticket prices, we need to understand consumer demand. In other words, pricing research and demand estimation go hand in hand. As we have seen earlier, price can be included as an attribute in conjoint and choice studies, providing a mechanism for estimating consumer willingness to pay and the consumer demand curve. If a sports team can understand consumer willingness to pay, it

will be better able to set prices for tickets, concessions, and branded merchandise.

Much work in the area of microeconomics and econometrics concerns estimation of the parameters of demand and the *price elasticity* of demand.

From economics we understand that the price elasticity of demand is the percentage change in quantity demanded divided by the percentage change in price. With elasticity less than 1.0, a large change in price results in a small change in quantity demanded, as we would expect for necessities or goods and services essential to life. With elasticity greater than 1.0, a small change in price results in a large change in quantity demanded. This is typical of luxury goods and services, including entertainment and sports.

Defining price elasticity is easy, but estimating elasticity is not so easy because markets and the variables that affect market prices are constantly changing. Economic researchers focus on aggregate consumer behavior, demand functions, and price elasticity, whereas marketing researchers focus on consumer heterogeneity and marketing mix factors that affect demand. How much of a product is demanded and purchased varies with consumer preferences and with the consumer's situation in life, defined by the time available to shop and the money available to spend. The term *price sensitivity* refers to consumer individual differences in demand. And some marketing theorists use the term *reference price* to describe a hypothetical internal standard that a consumer uses in judging the prices of products.

Specialized pricing research methods include monadic tests in which a product or product description, including price, is

placed in context with a likely set of competitive products, and the consumer respondent is asked if she would purchase the product. Across many respondents at many price points, monadic tests may provide meaningful estimates of price sensitivity, helping to find the price that will generate the highest contribution to profit.

Setting ticket prices for sporting events is a complicated process because stadia have numerous classes of seating and associated classes of tickets. With an understanding of willingness to pay and knowledge of stadium seating capacity across various classes of tickets, we could use mathematical programming to maximize (or minimize) an objective. Mathematical programming requires an objective. What is the objective?

Do teams set ticket prices in order to maximize ticket revenue? Curiously, the answer to this question is no. Many teams set ticket prices lower than the revenue-maximizing price. There are many reasons for this, as Fort and Winfree (2013) point out.

Prices lower than willingness to pay are associated with economic *consumer surplus*, which in turn generates good will among fans. Season tickets are heavily discounted to attract purchase. Lower prices ensure that more fans will go to games, resulting in higher concession and merchandise sales. In-stadium advertising is easier to sell when advertisers are assured that games will be sellouts. Lower ticket prices attract new fans—their trial purchases may be followed by repeat purchases. Furthermore, a team’s reputation depends not only on winning but also on having large and active crowds at the stadium.

Sports ticket pricing, airline, and hotel ticket pricing share in having capacity constraints. The objective in airline or hotel ticket pricing is revenue maximization alone. There is no requirement to fill all the seats or hotel rooms. Sports teams, on the other hand, have multiple objectives. They want to achieve the highest possible revenue while filling as many seats as possible. To define a tractable, single-goal mathematical programming model, we could simplify the problem, setting the objective of maximizing ticket revenue while filling all the seats.

Suppose we were working for the Dallas Cowboys. AT&T Stadium, home of the Dallas Cowboys, has many classes of seats, with prices between \$80 for limited visibility seats to more than \$36 thousand for the best luxury suites. Are these prices optimal, given an objective of maximizing ticket revenue while filling all the seats? We can use mathematical programming to answer this question. Perhaps the Cowboys should raise prices at the low end and/or lower prices at the high end.

When developing plans for revenue growth and pricing, it is useful to consider a business in relation to its suppliers and buyers. We draw a picture showing potential new entrants as well as substitute products. In [figure 8.1](#) we show a strategic analysis of the Golden State Warriors using the five-forces model developed by Porter (1980).

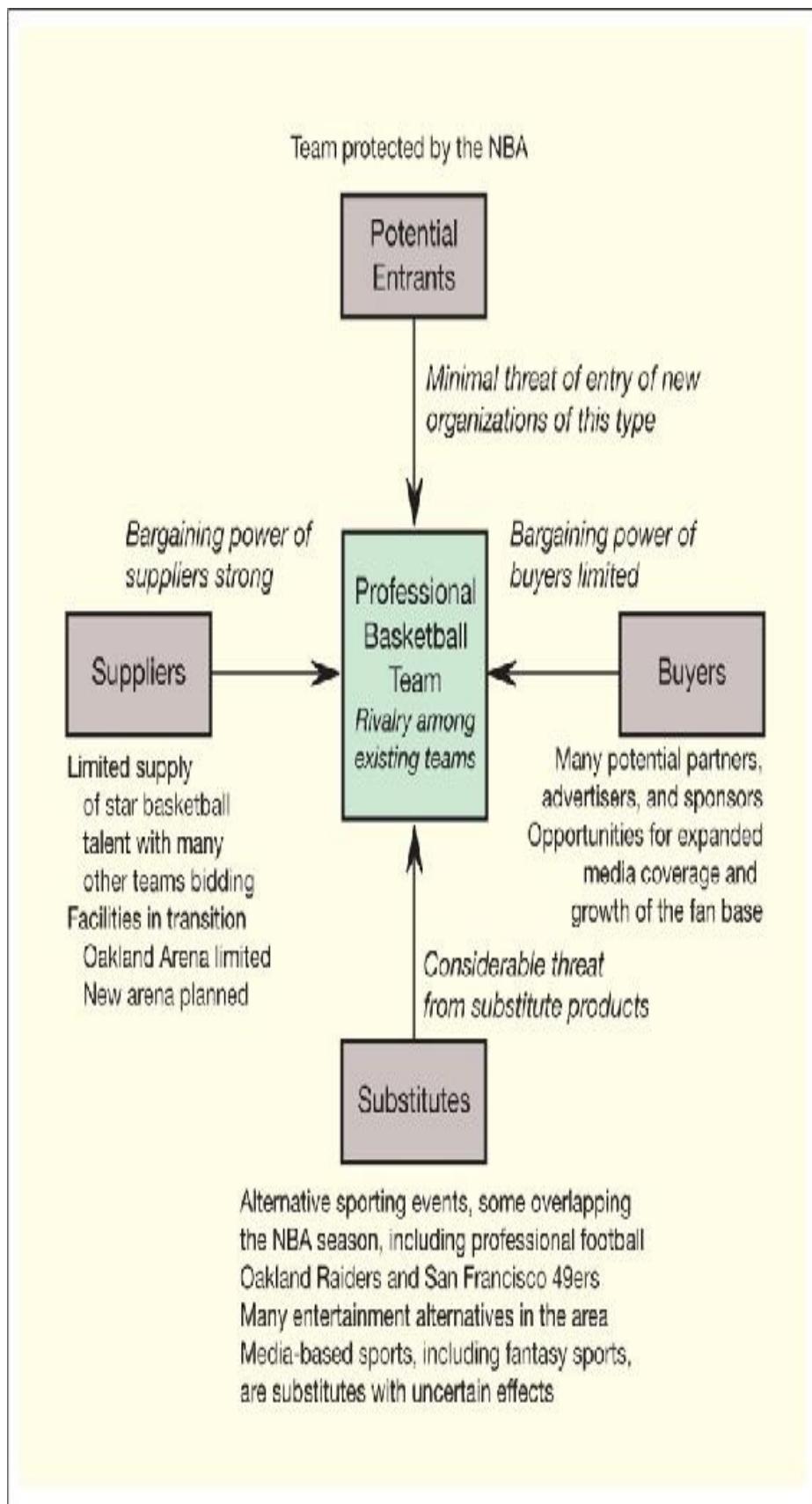


Figure 8.1. Competitive Analysis for an NBA Team: Golden State Warriors

The Golden State Warriors compete with other teams for national attention and brand loyalty. But like other NBA teams they are cooperators in the league, protected from the thread of new entrants in the form of professional basketball teams in the San Francisco Bay area.

The most difficult part of this business is managing resources: players and facilities. The team competes in a tough labor market. Star players are in short supply and in great demand from other NBA teams. NBA faces a tough negotiator in the National Basketball Players Association (NBPA). The current player agreement extends through the 2020–21 season, but either side can opt out in 2017 (Streeter 2015).

The team has a loyal fan base and considerable opportunities for growth. Winning the NBA finals in 2015 had a strong positive effect on fans. There are opportunities for revenue growth through media, business partners, advertisers, and sponsors. These opportunities should grow after the team moves into a newer, larger arena planned at a San Francisco site. In-stadium revenues can grow by employing methods for increased fan engagement, such as those used by the San Francisco Giants (Schlough 2015).

Media growth has been exceptional. NBA commissioner, Adam Silver, has looked to media as a way to expand the reach of member teams internationally. In social media, the NBA has taken the lead among US-based leagues, with a reported 759 million worldwide users following the league or expressing positive feelings (social media “likes”) for NBA teams (Keating 2015). Revenues from television should grow

as a seven-year, \$24 billion deal with ESPN and the Turner Broadcasting System takes effect in 2016 (Streeter 2015)

There is threat from substitute products, including professional sports teams when seasons overlap. There are also many entertainment options in the Bay area. The Golden State Warriors, like other professional sports teams, may have uncertainties about fantasy sports. Perhaps participation in fantasy sports will translate into increased involvement with the team and its players. Fans see their favorite fantasy players on the court. On the other hand, fantasy sports consumes part of a fan's discretionary income, leaving less to spend on tickets and merchandise. It can also be argued that fantasy sports do little to build fan affiliation with teams because fantasy games concern players as individuals, independent of their true teams.

The benefits of market segmentation are described in marketing management references (Dickson 1997; Kotler and Keller 2012). Sternthal and Tybout (2001) and Cespedes, Dougherty, and Skinner (2013) review management issues in segmentation and targeting. Frank, Massey, and Wind (1972), Neal (2000), and Wedel and Kamakura (2000) discuss the objectives and methods of market segmentation.

Reviews of traditional direct marketing are provided by Wunderman (1996), and Nash (2000, 1995). Hughes (2000) discusses strategies associated with database marketing and online direct marketing. Direct and database marketing is a rich area of application in marketing data science. Anand and Büchner (2002) discuss applications in cross-selling, finding prospects for additional products from an existing customer list. Blattberg, Kim, and Neslin (2008) provide a

comprehensive review of modeling methods in direct and database marketing, including extensive discussion of RFM models, lift charts, and alternative methods for setting probability cut-offs.

Bayesians use the term *consumer heterogeneity* to refer to individual differences across customers. The thinking is that describing consumers in terms of their positions along underlying attribute parameters is more informative than describing them as being members of segments. Bayesian methods in marketing, reviewed by Rossi, Allenby, and McCulloch (2005), have been implemented in R packages by Rossi (2014) and Sermas (2014).

Introductions to pricing theory and research may be found in books about microeconomics (Pindyck and Rubinfeld 2004; Hirshleifer, Glazer, and Hirshleifer 2005; Varian 2005) and econometrics (Pindyck and Rubinfeld 2000; Greene 2002). Discussion of pricing research methods may be found in Lyon (2000, 2002), Feldman (2002), and Orme (2006). Marder (1997) reviews monadic pricing methods. Nagle and Hogan (2005) and Krishnamurthi (2001) present reviews of pricing strategy and management issues. Empirical research about reference prices is summarized by Kalyanaram and Winer (1995). Revenue growth and yield management through mathematical programming are well understood for airlines and hotels (Phillips 2005; Karaesmen, van Ryzin, Talluri, and Vulcano 2008). Barlow (2000) shows that similar methods may be employed to fill seats in sports stadia.

9. Managing Finances

Billy: "I can't compete against \$120 million on a payroll with \$38 million."

Steve: "We're not going to compete with these teams that have big budgets. We're gonna work within the constraints that we have, and you're gonna go out and do the best job that you can recruiting new players. We're not going to pay \$17 million for players."

Billy: "I'm not asking you for 10, 20, 30 million dollars. I'm just asking for a little bit of help. Just get me a little bit closer, and I will get you that championship team. I mean, this is why I am here. This is why you hired me. And I gotta ask you what are we doing here . . ."

Steve: "Billy, you know, I . . ."

Billy: ". . . if it's not to win a championship."

Steve: "I want to win just as much as . . ."

Billy: "That's my bar. My bar is here. [Holds his hand high.] My bar is to take this team to the championship."

Steve: "Billy, we're a small-market team, and you're a small-market GM. I'm asking you to be okay not spending money that I don't have. And I'm asking you to take a deep breath, shake off the loss, get back in a room with your guys and figure out how to find replacements for the guys we lost with the money that we do have."

—BRAD PITT AS BILLY BEANE AND STEVE DAKOTA AS STEVE IN
Moneyball (2011)

A *Moneyball* ethos guides many professional sports teams. In a player draft or free-agent bidding situation, a team's objective is to select the best player or combination of players subject to budgetary or salary-cap constraints. "Best" may be defined as being able to contribute the most to the team in additional runs or points across future games. The team's planning horizon could be the remainder of the season and playoffs or future seasons. Player contract costs vary from one player to the next, especially among free agents.

There are many options to consider when adding new players to a team. We need to identify the set of possible players and estimate each player's potential contribution to the team in runs or points scored, or in games won across the team's planning horizon. There are constraints on the number of players on a team, so bringing on a new player means dropping one of the team's current players. There are constraints regarding player positions. If a professional basketball team acquires a free-agent guard, for example, it may well drop another guard from its player roster. There are costs associated with each player being acquired and each player being dropped. Furthermore, player contract terms may specify that dropped players continue to earn all or a portion of their salaries.

In theory, maximizing output subject to constraints may be addressed by mathematical programming. In practice, player draft and free-agent acquisitions represent challenging problems in operations research and data science. Costs are difficult to specify with any degree of confidence. This is a labor marketplace with many teams bidding for the same players. Furthermore, a player's contribution to a team is

better estimated as a posterior probability distribution than as a single number, making the problem one of stochastic programming, which we return to later in this chapter.

Let us begin our consideration of costs by looking at cost-volume-profit analysis from management accounting, illustrated in figure 9.1. This is a common method for assessing costs and benefits associated with alternative business practices. Although cost-volume-profit analysis would be difficult to apply to player or stadium costs, we can use it to assess particular aspects of sports business, such as ticket, concession, or merchandise sales.

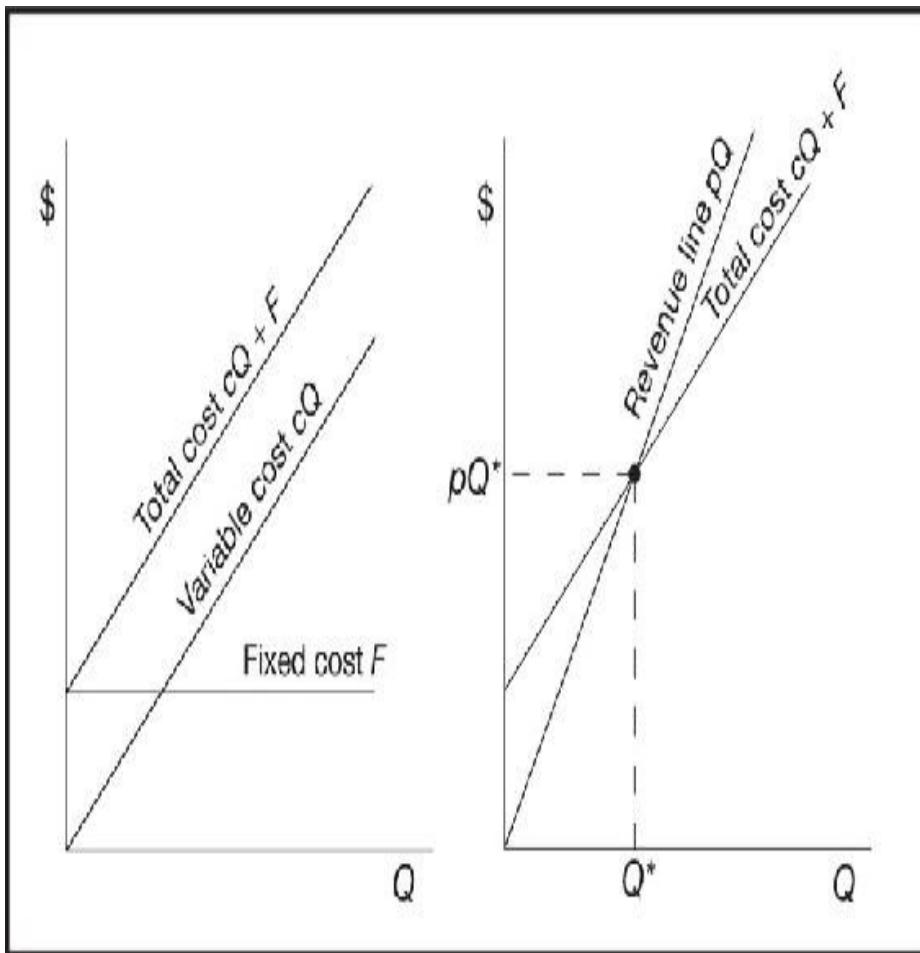


Figure 9.1. Cost-Volume-Profit Analysis

Suppose we consider hot dogs sold at baseball games, in particular, the Dodger Dog sold at Dodger Stadium. Supplied by the company Farmer John®, the Dodger Dog is sold at a particular price at Dodger Stadium, call it p . For the purposes of our analysis, we consider the unit cost of ingredients and wrappings to be c . These costs are known as variable costs because they vary with the number of hot dogs sold.

Fixed costs do not vary with the number of units produced and sold. Typical fixed costs include start-up costs, licenses, contracting costs, management overhead, and advertising. Areas within each level of Dodger Stadium are designated for concessions and these areas are staffed by food workers—fixed costs F .

Variable costs vary with the number of units produced and sold—they vary with quantity Q , the quantity of hot dogs produced and sold. Then total variable cost is given by cQ . Adding variable and fixed costs gives total cost: $cQ + F$. Total revenue is unit price p times quantity sold Q , and profit is total revenue minus total cost. Using π to represent profit, we have the formula $\pi = pQ - (cQ - F)$.

The figure for cost-volume-profit analysis, reproduced with shaded areas in [figure 9.2](#), shows fixed costs as a horizontal line. We represent variable costs as a straight line with slope c and vertical intercept zero. And we draw total cost as a straight line parallel to the variable cost line with a vertical intercept F . From the figure, we can see that total costs equal total revenues at the breakeven quantity Q^* , with associated breakeven sales pQ^* .

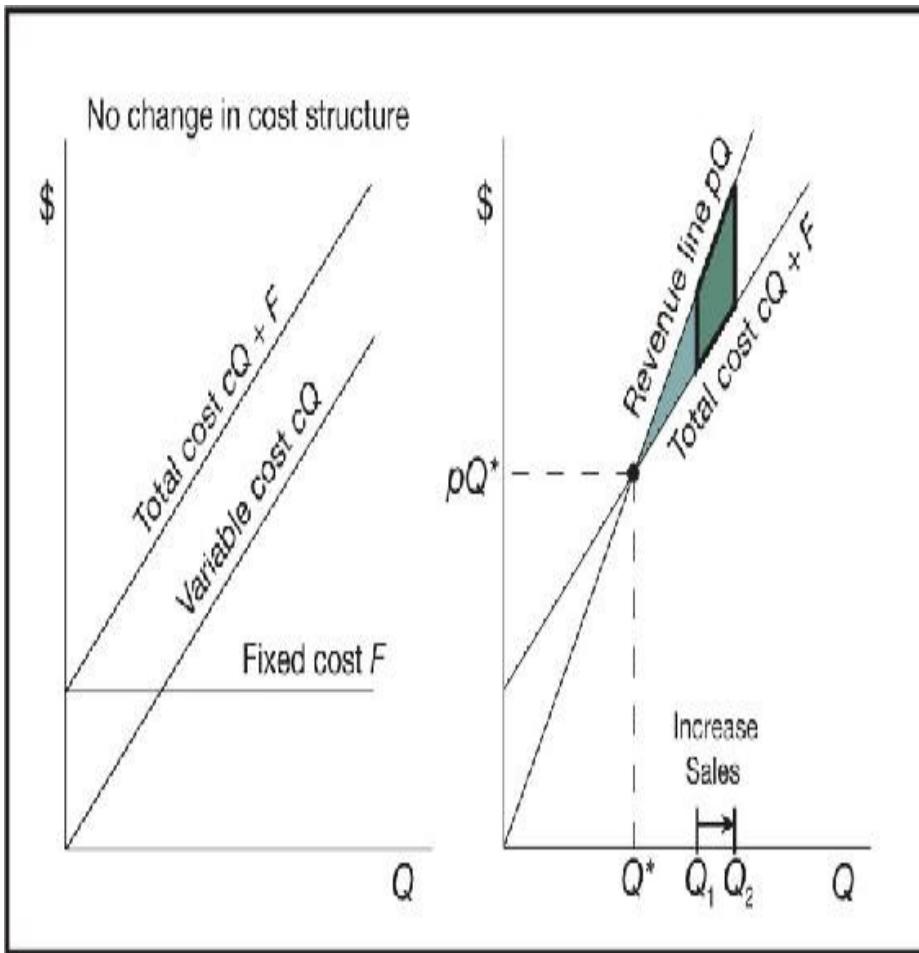


Figure 9.2. Higher Profits Through Increased Sales

To make money on hot dog sales, the team must sell more than Q^* Dodger Dogs. If the team sells Q_1 , its profit will be represented by the lighter shaded area in the figure between the revenue line and the total cost line. One path to higher profits is to increase sales volume from Q_1 to Q_2 . This would add to profits, with the amount added shown by the darker shaded area in figure 9.2.

Two other paths to increased profit, represented in figures 9.3 and 9.4, are to lower fixed costs or to run a more efficient operation. Thinking of Dodger Dog sales, the team could lower fixed costs by using fewer employees or less stadium

area for concessions. This would lower the vertical axis intercept of both the fixed and total cost lines. Variable costs are affected by unit costs or the ingredients and wrappings that come with each Dodger Dog. Lowering unit variable cost reduces the slope of the variable and total cost lines.

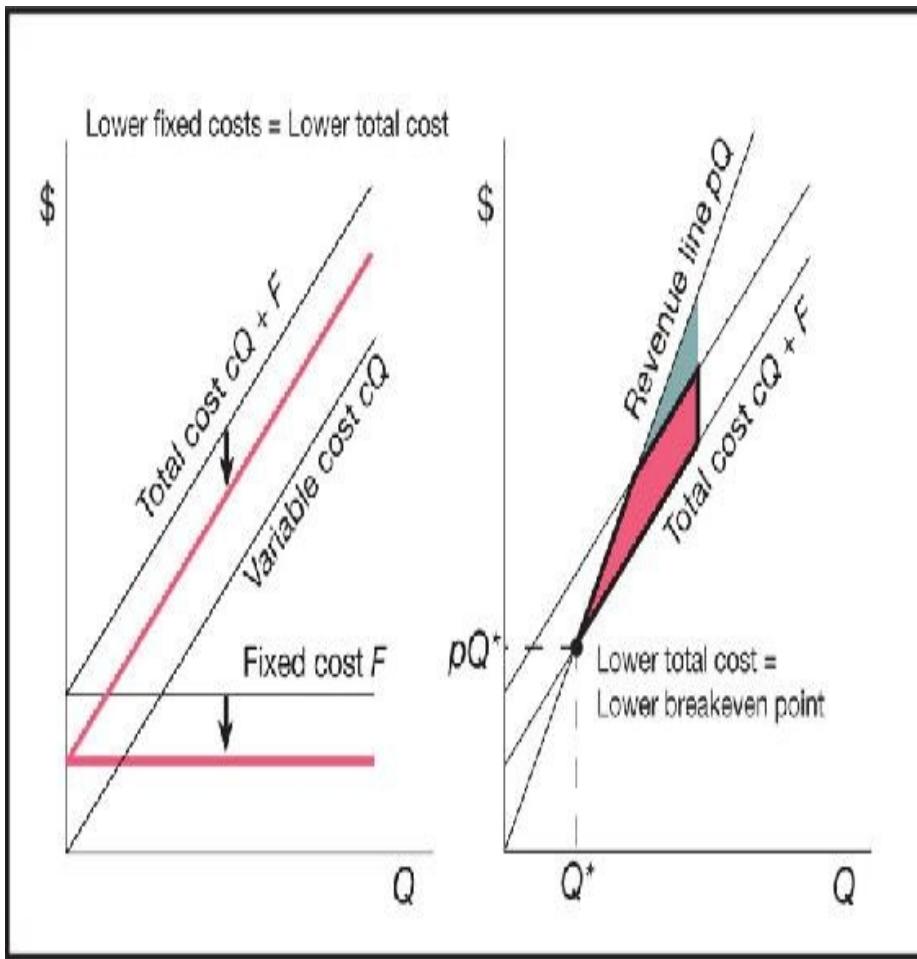


Figure 9.3. Higher Profits Through Lower Fixed Costs

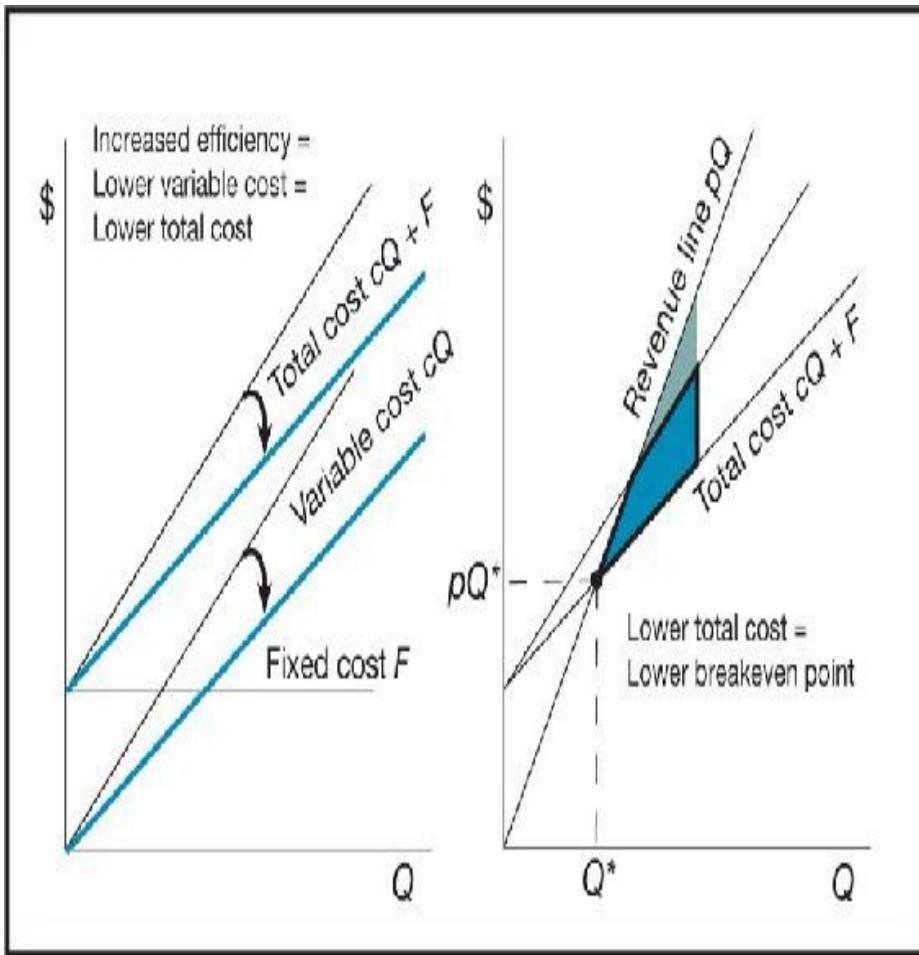


Figure 9.4. Higher Profits Through Increased Efficiency

Cost-volume-profit analysis provides a convenient mechanism for showing three distinct paths to higher profit: (1) increase sales volume, (2) reduce fixed costs, or (3) become more efficient, lowering variable costs. Cost-volume-profit analysis, sometimes referred to as breakeven analysis, assumes linear functions for sales revenue and costs. The quantities c and F are given, providing the slope and intercept of the total cost line. There is the quantity p , providing the slope of the revenue line. This implies that a single price p is being charged for each item sold.

Cost-volume-profit analysis is useful in assessing short-term

product and service problems in which costs and prices are constant across all items being sold. As we have seen earlier, professional sports teams may use variable or dynamic pricing methods, rather than use a single price p . Furthermore, many decisions made by professional sports teams are long-term decisions.

Decisions about stadium upgrades, information systems, and long-term player contracts are investment decisions, affecting a team's expenses and revenues for many years. We use methods of financial analysis, such as discounted cash flow analysis, to evaluate investment decisions.

Consider the Los Angeles Dodgers' decision to establish a long-term contract with left-handed ace Clayton Kershaw, which made him the highest paid player in professional sports. The term of the contract was seven years at \$215 million, with an opt-out after five years. As Knight (2015) points out, with this contract Kershaw joined Michael Jordan as the only American athlete to make an average of at least \$30 million a season over a multiyear deal.

For Kershaw to be a good investment for the Dodgers, the team must anticipate substantial increases in annual revenues coming from media, ticket sales, sponsorships, and other sources. There is the expectation that Kershaw's presence on the team will generate additional wins, and additional wins are associated with additional revenues. Kershaw's wins above replacement player (WARP) was estimated as 3.7, 5.7, and 6.1 across the 2012, 2013, and 2014 seasons (Miller and Wojciechowski 2015). In 2014, with contract in hand, Kershaw won both the National League Cy Young award and the National League Most Valuable Player award, the first

player to do so since Bob Gibson in 1968.

Most businesses evaluate capital investments and their associated annual expenses or costs by discounted cash flow analysis.¹ Table 9.1 shows a discounted cash flow analysis for the Dodgers' investment in Clayton Kershaw, assuming a 10 percent cost of capital (applied at the end of years for ease of calculation). Cash inflows and outflows are in nominal dollars. These are discounted by the cost of capital in all but the initial year of the investment. The 2014 cash out includes Kershaw's \$18 million signing bonus, which is not discounted, with his \$4 million salary, which is discounted. We might imagine that the Dodgers had translated Kershaw's contribution to wins into dollars of additional revenue. Revenues in table 9.1 are guesses or plugs to make net present values positive.

¹ Annual expenses represent cash outflows, and annual revenues represent cash inflows. Both flows are discounted for the term of the contract using the cost of capital for the organization. To justify capital investments, the sum of the discounted cash flows must be positive. The sum of the discounted cash flows is called the *net present value* of the investment. Discounting is used to adjust for the time value of money. Recall the idea that money today is worth more than money in the future. If we want money today, we have to pay for it. What is the price? The price of money is the interest rate or cost of capital. Lenders want to realize a return on the money they lend. Sometimes we adjust for inflation or deflation. Sometimes we adjust for the interest rate. And when a firm evaluates alternative investment projects, it adjusts for its cost of capital, which is a linear combination of interest rate and dividend rate paid to creditors and stockholders, respectively. Let r be the rate we are working with—the discount rate. Then an amount X to be received at the end of N periods has present value Y as follows:

$$Y = \frac{X}{(1+r)^N}$$

Year of Contract	Season	Cash Out (\$ millions)	Cash In (\$ millions)	Cash In-Out (\$ millions)	Discounted Cash In-Out (\$ millions)
0	Signing	18	0	-18	-18.000
1	2014	4	30	26	23.636
2	2015	30	31	1	0.826
3	2016	32	32	0	0.000
4	2017	33	33	0	0.000
5	2018	33	34	1	0.621
6	2019	32	35	3	1.693
7	2020	33	36	3	1.539

Cost of Capital (%): 10	Net Present Value	
	after 5 Years:	7.084
	after 7 Years:	10.317

Source for Kershaw contract terms: Stephen (2014).

Table 9.1. Discounted Cash Flow Analysis of a Player Contract: The Case of Clayton Kershaw

Discounted cash flow analysis is the foundation of investment analysis. Investment projects should have positive net present value when evaluated across their potential lifetimes. When comparing projects, we use net present value, choosing the project with the highest net present value.

When comparing investment projects, some financial managers like to use *return on investment (ROI)*. To compute ROI, we assume knowledge of cash inflows and outflows and compute the discount rate that would yield a net present value of zero. This discount rate corresponds to the ROI. To compare projects on ROI, it is important to ensure that the projects have the same duration or investment horizon.

Some managers like to think in terms of *payback period* or the

length of time it takes for the discounted cash inflows of a project to exceed the discounted cash outflows. It is not a good idea to compare projects on the basis of payback period. It encourages managers to be short-sighted when making investment decisions. Net present value is preferred to ROI and payback period when evaluating investment decisions.

We can use discounted cash flow analysis in assessing the investment value of sports franchises. A potential investor would consider the asking price for a team, anticipated revenues and expenses (thinking of these as cash inflows and outflows), and an investment horizon. The investor needs to include capital appreciation in the analysis because team values change dramatically across time.

The Brooklyn Nets were purchased for \$365 million in 2010. The franchise was valued at \$1.5 billion in January 2015, sixth in the NBA. Gate receipts in 2014 were \$64 million, and total revenues were \$212 million. But with player salaries at \$115 million, the team lost almost \$100 million in 2014. The Brooklyn Nets were the only NBA team with an operating income loss in 2014 ([Badenhausen, Ozanian, and Settimi 2015a](#); [Forbes 2015](#)).

Investing in the Brooklyn Nets can be analyzed using discounted cash flow analysis. Revenues are likely to rise due to inflation and the possibility of increased ticket prices, but we are unsure about our ability to control costs. We expect capital appreciation, given what we have seen in recent years with NBA teams. Suppose our investment horizon is five years. Discounted cash flow analyses are illustrated in [table 9.2](#).

Outcome A: \$2 Billion Sales Price after Five Years, No Cost Control

Index of Year	Season or Event	Cash In (\$ millions)	Cash Out (\$ millions)	Cash In-Out (\$ millions)	Discounted
					Cash In-Out (\$ millions)
0	Purchase	0	1,500	-1,500	-1,500
1	2015/16	220	320	-100	-95
2	2016/17	230	330	-100	-91
3	2017/18	240	340	-100	-86
4	2018/19	250	350	-100	-82
5	2019/20	260	360	-100	-78
5	Sale	2,000	0	2,000	1,567
Discount Rate (%):		5	Net Present Value		-366

Outcome B: \$2 Billion Sales Price after Five Years, Cost Control

Index of Year	Season or Event	Cash In (\$ millions)	Cash Out (\$ millions)	Cash In-Out (\$ millions)	Discounted
					Cash In-Out (\$ millions)
0	Purchase	0	1,500	-1,500	-1,500
1	2015/16	220	250	-30	-29
2	2016/17	230	250	-20	-18

3	2017/18	240	250	-10	-9
4	2018/19	250	250	0	0
5	2019/20	260	250	10	8
5	Sale	2,000	0	2,000	1,567
Discount Rate (%):		5	Net Present Value		20

Outcome C: \$2.5 Billion Sales Price after Five Years, Cost Control

Index of Year	Season or Event	Discounted			
		Cash In (\$ millions)	Cash Out (\$ millions)	Cash In-Out (\$ millions)	Cash In-Out (\$ millions)
0	Purchase	0	1,500	-1,500	-1,500
1	2015/16	220	250	-30	-29
2	2016/17	230	250	-20	-18
3	2017/18	240	250	-10	-9
4	2018/19	250	250	0	0
5	2019/20	260	250	10	8
5	Sale	2,500	0	2,500	1,959
Discount Rate (%):		5	Net Present Value		411

Table 9.2. Would you like to buy the Brooklyn Nets?

We are not sure we want to invest in the Brooklyn Nets. In five years the sales price might be \$2 billion. Perhaps there is a bridge in the mix, and the sales price is \$2.5 million. Should we invest in the Nets or put our money in the stock market? Decisions do not lead to perfectly predictable outcomes. We have uncertainty about outcomes. As rational decision makers, we can take the net present values across the three outcomes and multiply by their probabilities of occurrence. Summing these products, we obtain the expected value of the investment, a probability-weighted combination of net present

values.

To continue with the analysis, we compare this expected value of an investment in the Brooklyn Nets with the expected value of an investment in the stock market across the same investment horizon. Decision analysis is a way of dealing with uncertainty. Decision analysis considers decision options, possible outcomes, and probabilities of outcomes. A rational decision maker, economists tell us, will choose the option with the highest expected value. A summary of decision analysis may be displayed in a decision tree, as shown in figure 9.5.

Decision: Buy the Brooklyn Nets

Possible Outcomes	Net Present Value (\$ millions)	Probability	Probability x Net Present Value (\$ millions)
Outcome A: \$2 Billion Sale, No Cost Control	-366	0.20	-73
Outcome B: \$2 Billion Sale, Cost Control	20	0.50	10
Outcome C: \$2.5 Billion Sale, Cost Control	411	0.30	123
Expected Value			60

Decision: Invest in the Stock Market

Possible Outcomes	Net Present Value (\$ millions)	Probability	Probability x Net Present Value (\$ millions)
Market Up: Make \$100 Million on \$1.5 Billion	100	0.60	60
Market Even (Earnings Rate = Inflation Rate)	0	0.20	0
Market Down: Lose \$100 Million on \$1.5 Billion	-100	0.20	-20
Expected Value			40

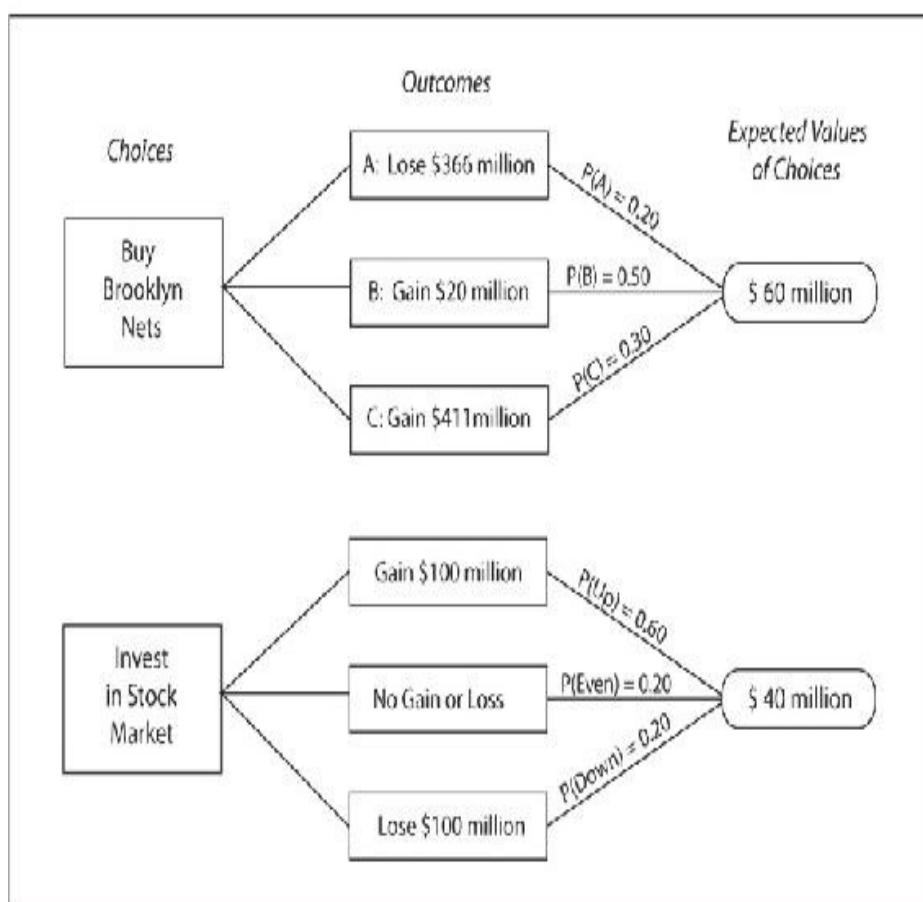


Figure 9.5. Decision Analysis: Investing in a Sports Franchise (Or Not)

Are sports teams a good investment? We can try to answer this question by reviewing sales prices for professional sports teams over time. In Major League Baseball, for example, sports franchises have been a good investment over the history of the sport, offering higher returns (capital gains) than investments in comparably sized businesses outside of sports. Performance results have been mixed, however, with selected decades (1970s and 1990s) showing negative or zero returns (Fort 2006).

Fort (2011) presents methods for the valuation of sports franchises. See Weil *et al.* (2005) for an introduction to accounting principles and Higgins (2015) or Brealey, Myers, and Allen (2013) for discussion of financial analysis methods, including discounted cash flow analysis. Hirshleifer, Glazer, and Hirshleifer (2005) show how discounted cash flow analysis may be understood within the context of economics and the time-value of money. Raiffa (1968) introduces methods of decision analysis which have a wide range of applications in business, public policy and everyday life (Stokey and Zeckhauser 1978; Thaler 1994; Thaler and Sunstein 2009).

Decision analysis shows a way to deal with uncertainty. If we trust our estimates of gains and losses across discrete outcomes and if we trust probabilities assigned to those outcomes, we are well on our way to a rational choice. The calculations are easy. What is not so easy is the estimation of gains, losses, and probabilities.

Regarding the estimation of probabilities, there are additional

methods and models to employ. Following Bayesian thinking, probability estimation may be carried out in a rational manner, going from prior, subjective probability distributions to posterior distributions. This is another calculation, sometimes easy, sometimes not so easy. Bayesian statistics shows how to measure uncertainty. As we continue to collect data, updating our priors, we become less and less dependent on subjective probability estimates.

As indicated earlier, many sports management decisions can be formulated as constrained optimization problems. We maximize revenues or minimize costs, subject to constraints. With perfect information or precise measures in hand, all would be well. We would input data for parameters and let the programs find an optimal solution. This is the deterministic world of mathematical programming, as reviewed in appendix A (page 200).

But the life of a sports analyst or data scientist is not so easy. Consider the problem posed at the beginning of this chapter—selecting the best combination of players subject to the budgetary constraint of a salary cap. We might simplify the problem by assigning a single contract cost to each player, fixing some of the parameters required for mathematical programming or, to be precise, integer programming with binary decision variables. But what should we do about player contributions to the team?

However we measure players' contributions to a team, those measures have uncertainty. We need a way to deal with that uncertainty. One approach would be to solve the integer programming problem numerous times, each with a different set of values for player contribution.

We could consider three values for player contribution: best value, expected value, and worst value. We run the integer program three times, choosing the optimal combination of players in each run. Then we identify common players across the three solutions, if indeed there are common players. We call this the best case/worst-case approach.

A similar approach, which we might call the brute-force approach, would draw a single value from each player's distribution and use that value as input to the model. Each run of the model would have a randomly selected value for each player, fixing the input data for the parameters. We run the integer program thousands of times and count the frequency with which each player is included in the solutions. The brute-force approach is similar to a *sensitivity analysis* in which we vary input parameters across many runs and see what happens to the solution.

A third approach, stochastic programming, represents mathematical programming with uncertain input data. Rather than specify parameters as fixed input data, we use probability distributions. For example, we might begin by assuming that a player's contribution to wins above replacement has an exponential distribution, continuous on the open interval of positive real numbers.

Stochastic programming also gives us a way of specifying player costs with uncertainty. Using stochastic programming, we can obtain a single optimal set of players for the team, accounting for uncertainties inherent in the player selection problem.

The rationale for stochastic programming is presented by Conejo, Carrión, and Morales (2010). A mathematical

introduction to stochastic programming methods is provided by Birge and Louveaux (2011). Gershkov and Moldavanu (2014) develop stochastic programming theory for capacity-constrained dynamic pricing.

Hart, Laird, Watson, and Woodruff (2012), Watson and Woodruff (2012), Sirona (2014), and Hart and Woodruff (2015) review a Python-based algebraic modeling system for mathematical and stochastic programming. Algebraic modeling systems allow analysts to construct abstract models, defining parameters prior to providing input data for those parameters. Such systems are most useful in sensitivity testing and stochastic programming. And with an open-source, Python-based system, analysts can interrogate and manipulate model parameters and their distributions, integrating stochastic programming and Bayesian estimation methods.

10. Playing What-if Games

Maggie: “Mr. Dunn?”

Dunn: “I owe you money?”

Maggie: “No, Sir.”

Dunn: “I know your Mamma?”

Maggie: “I don’t rightly know, Sir.”

Dunn: “What do you want?”

Maggie: “I was on the other card.

I won my fight, too—Maggie Fitzgerald.”

Dunn: “Well, Maggie Fitzgerald, what do you want?”

Maggie: “Did you happen to see it?”

Dunn: “Nope.”

Maggie: “I did pretty good.

Thought you might be interested in training me.”

Dunn: “I don’t train girls.”

Maggie: “Maybe you should.

People seen me fight say I’m pretty tough.”

—HILARY SWANK AS MAGGIE FITZGERALD, AND CLINT EASTWOOD AS FRANKIE DUNN IN *Million Dollar Baby* (2004)

Playing what-if games means running simulations. To illustrate the process, we can use simulations to pick winning teams. Suppose we consider the 2007 baseball season and use selected interleague games as an example. We note that there are games scheduled between the Mets and Yankees at Yankee Stadium for Friday, Saturday, and Sunday, June 15, 16, and 17, 2007. Just prior to June 15 we ask these questions: How

many runs can the Mets be expected to score against the Yankees when playing at Yankee Stadium? How many runs can the Yankees be expected to score? And, consequently, which team is expected to win these three games?

To predict straight-up runs scored in the future, we start with runs scored in the past. We select Mets' and Yankees' data for the current season, ignoring team statistics from previous seasons and the 2007 preseason. June 15 would be the first meeting of the Mets and Yankees at Yankee Stadium in the regular 2007 season, so we need to use runs scored against other teams. For the Mets, we can select all Mets' regular season away games between opening day in St. Louis with the Cardinals on May 1 to an away game on June 13 with the Los Angeles Dodgers. And for the Yankees, we can select all Yankees home games prior to June 15, which begin with the home opener with the Devil Rays on April 2 to a home game with the Diamondbacks on June 14.

Table 10.1 provides a complete list of Mets' games and scores between April 1 and June 15, and table 10.2 provides a complete list of Yankees' games and scores for the same period. We can see that the Mets have been scoring more runs than their opponents in away games, but fewer in home games. The Yankees, on the other hand, have been scoring more runs than their opponents both home and away.

Away Games (<i>n</i> = 31)				Home Games (<i>n</i> = 33)			
		Runs Scored				Runs Scored	
Date	Team	Home Mets	Home Team	Date	Team	Away Team	Away Mets
4/1/07	STL	6	1	4/9/07	PHI	5	11
4/3/07	STL	4	1	4/11/07	PHI	5	2
4/4/07	STL	10	0	4/12/07	PHI	3	5
4/6/07	ATL	11	1	4/13/07	WSH	2	3
4/7/07	ATL	3	5	4/14/07	WSH	6	2
4/8/07	ATL	2	3	4/20/07	ATL	7	3
4/17/07	PHI	8	1	4/21/07	ATL	2	7
4/18/07	FLA	9	2	4/22/07	ATL	9	6
4/19/07	FLA	11	3	4/23/07	COL	1	6
4/27/07	WSH	3	4	4/24/07	COL	1	2
4/28/07	WSH	6	2	4/25/07	COL	11	5
4/29/07	WSH	1	0	4/30/07	FLA	9	6
5/3/07	ARI	9	4	5/1/07	FLA	5	2
5/4/07	ARI	5	3	5/2/07	FLA	3	6
5/5/07	ARI	6	2	5/11/07	MIL	4	5
5/6/07	ARI	1	3	5/12/07	MIL	12	3
5/7/07	SF	4	9	5/13/07	MIL	1	9
5/8/07	SF	4	1	5/14/07	CHC	4	5
5/9/07	SF	5	3	5/15/07	CHC	10	1
5/22/07	ATL	1	8	5/16/07	CHC	1	8
5/23/07	ATL	3	0	5/17/07	CHC	5	6

5/24/07	ATL	1	2	5/18/07	NYY	2	3
5/25/07	FLA	6	2	5/19/07	NYY	7	10
5/26/07	FLA	7	2	5/20/07	NYY	6	2
5/27/07	FLA	6	4	5/29/07	SF	4	5
6/8/07	DET	3	0	5/30/07	SF	3	0
6/9/07	DET	7	8	5/31/07	SF	2	4
6/10/07	DET	7	15	6/1/07	ARI	5	1
6/11/07	LAD	3	5	6/2/07	ARI	1	7
6/12/07	LAD	1	4	6/3/07	ARI	4	1
6/13/07	LAD	1	9	6/5/07	PHI	4	2
				6/6/07	PHI	4	2
				6/7/07	PHI	6	3
Average Runs		4.97	3.45	Average Runs		4.67	4.33

Table 10.1. *New York Mets' Early Season Games in 2007*

Away Games (<i>n</i> = 31)				Home Games (<i>n</i> = 33)			
		Runs Scored				Runs Scored	
Date	Team	Home Yankees	Home Team	Date	Team	Away Team	Away Yankees
4/9/07	MIN	8	2	4/2/07	TB	5	9
4/10/07	MIN	10	1	4/5/07	TB	7	6
4/11/07	MIN	1	5	4/6/07	BAL	6	4
4/13/07	OAK	4	5	4/7/07	BAL	7	10
4/14/07	OAK	4	3	4/8/07	BAL	6	4
4/15/07	OAK	4	5	4/17/07	CLE	3	10
4/20/07	BOS	6	7	4/18/07	CLE	2	9
4/21/07	BOS	5	7	4/19/07	CLE	6	8
4/22/07	BOS	6	7	4/26/07	TOR	6	0
4/23/07	TB	8	10	4/27/07	BOS	11	4
4/24/07	TB	4	6	4/28/07	BOS	1	3
5/1/07	TEX	10	1	4/29/07	BOS	7	4
5/3/07	TEX	4	3	5/4/07	SEA	15	11
5/3/07	TEX	5	2	5/5/07	SEA	1	8
5/11/07	SEA	0	3	5/6/07	SEA	0	5
5/12/07	SEA	7	2	5/7/07	SEA	3	2
5/13/07	SEA	1	2	5/8/07	TEX	2	8
5/16/07	CWS	3	5	5/9/07	TEX	2	6

5/16/07	CWS	8	1	5/10/07	TEX	14	2
5/17/07	CWS	1	4	5/21/07	BOS	2	6
5/18/07	NYM	2	3	5/22/07	BOS	7	3
5/19/07	NYM	7	10	5/23/07	BOS	3	8
5/20/07	NYM	6	2	5/25/07	LAA	10	6
5/28/07	TOR	2	7	5/26/07	LAA	3	1
5/29/07	TOR	2	3	5/27/07	LAA	4	3
5/30/07	TOR	10	5	6/8/07	PIT	4	5
6/1/07	BOS	9	5	6/9/07	PIT	3	9
6/2/07	BOS	6	11	6/10/07	PIT	6	13
6/3/07	BOS	6	5	6/12/07	ARI	1	4
6/4/07	CWS	4	6	6/13/07	ARI	2	7
6/5/07	CWS	7	3	6/14/07	ARI	1	7
6/6/07	CWS	5	1				
6/7/07	CWS	10	3				
Average Runs		5.30	4.39	Average Runs		4.84	5.97

Table 10.2. *New York Yankees' Early Season Games in 2007*

Overall, Yankees' games appear to have more scoring than Mets' games. This is understandable because American League games have a designated hitter. All but three Yankees' games were played in American League parks. The three games that the Yankees played in a National League park prior to June 15, 2007 happened to be at Shea Stadium against the Mets:

May 18	Yankees 2	Mets 3
May 19	Yankees 7	Mets 10
May 20	Yankees 6	Mets 2

One might be tempted to use the results of these three games to predict the outcome of the games on June 15, 16, and 17. But three games do not provide sufficient data from which to draw trustworthy inferences. Furthermore, the upcoming

Yankees-Mets games will be played at Yankee Stadium with the designated hitter rule in force, a fact that we would like to accommodate (at least in part) through our simulation.

Game-day simulation uses data from real games in the past to generate data about hypothetical games, past and future. Many sports fans are familiar with simulation from fantasy sports. A fantasy baseball game can have Babe Ruth hitting against Bob Gibson or Honus Wagner hitting against Sandy Koufax.

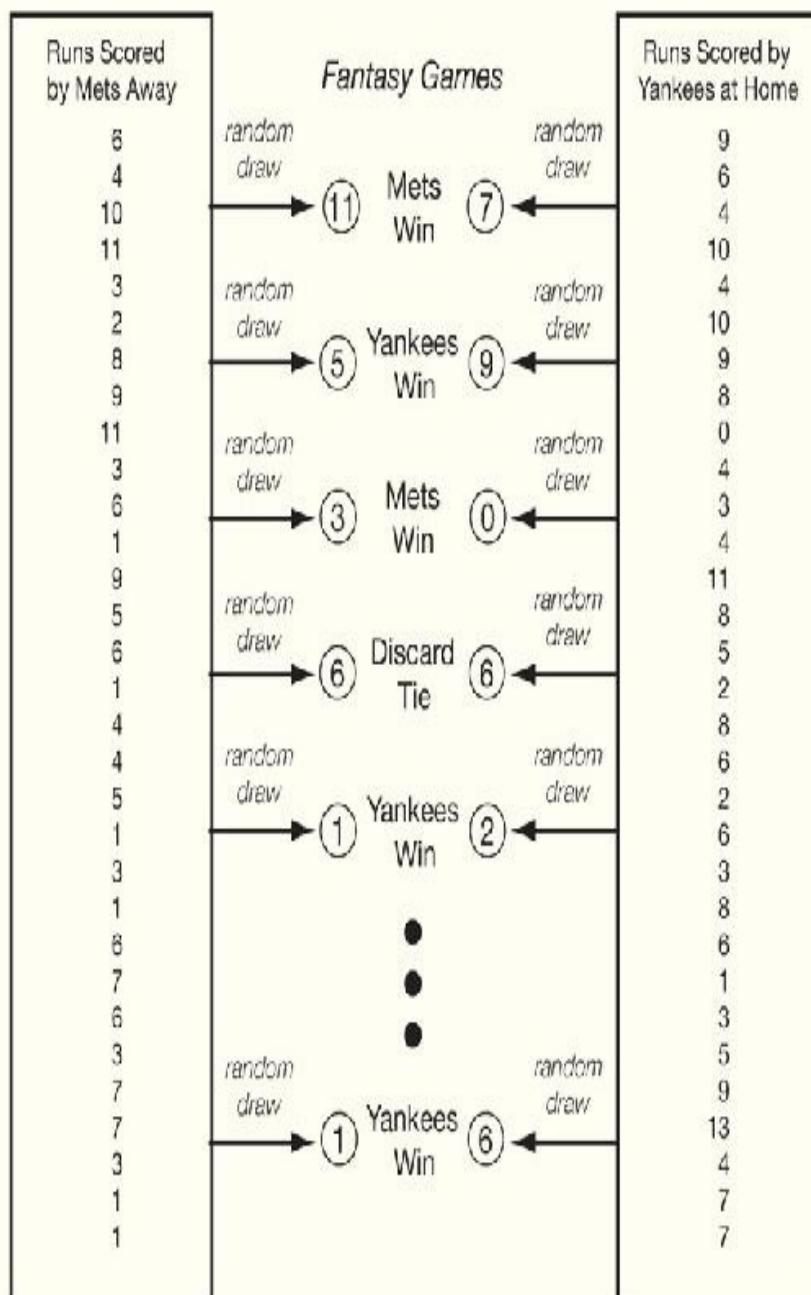
Simulations are executed by computer. They utilize baseball statistics from the history of the game and its players.

Randomness is introduced by random number generators, so that playing the same game again and again provides a distinct outcome each time.

Taking the lead from fantasy sports, we can create fantasy games with an objective to estimate the probability that one team will beat another. For example, suppose we use a simple game-day simulation to pick the winning team in the June Mets-Yankees' series. Drawing from the away runs-scored distribution of the Mets and the home runs-scored distribution of the Yankees on each playing of our fantasy game (each iteration of the simulation), we observe the runs scored by the Mets and the runs scored by the Yankees. If tied, we discard the observation. If the Mets' score is higher than the Yankees' score, we count a win for the Mets. If the Yankees' score is higher than the Mets' score, we count a win for the Yankees. Dividing the number of times the Mets win by the number of times we play the game without a tie provides an estimate of the probability of the Mets winning. Dividing the number of times the Yankees win by the number of times we play the game without a tie gives an estimate of the the probability of

the Yankees winning, which is one minus the probability of the Mets winning.

The game-day simulation we are describing is an empirical simulation because it is based on empirical distributions of runs scored by the two teams. The simulation uses only past runs scored and an identification of the Mets and Yankees as the teams. Figure 10.1 illustrates the results of such a simulation.



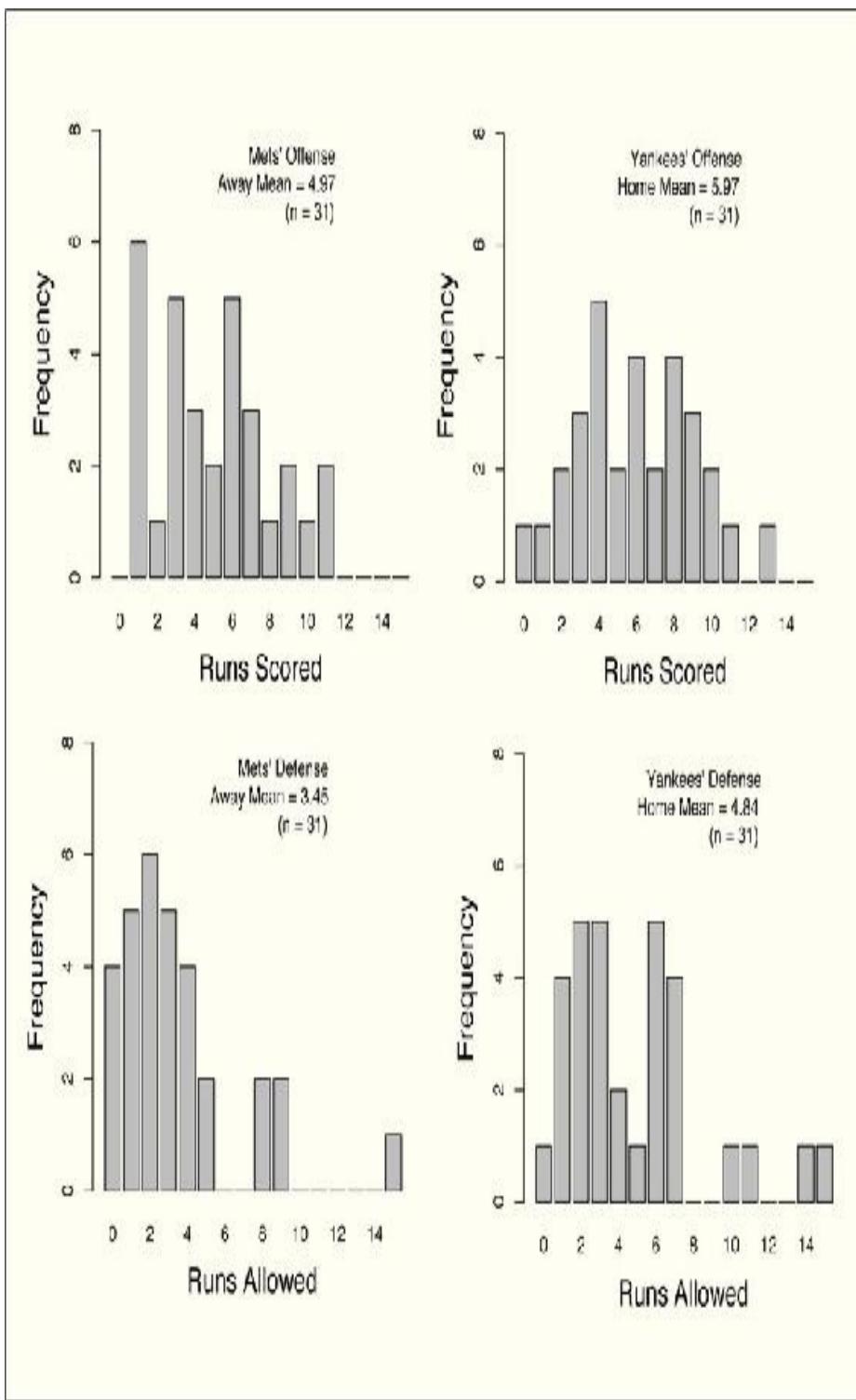
Results from 50,000 fantasy games without ties:

19,945 Mets' wins → 0.40 probability of winning

30,055 Yankees' wins → 0.60 probability of winning

Figure 10.1. *Game-day Simulation (Offense Only)*

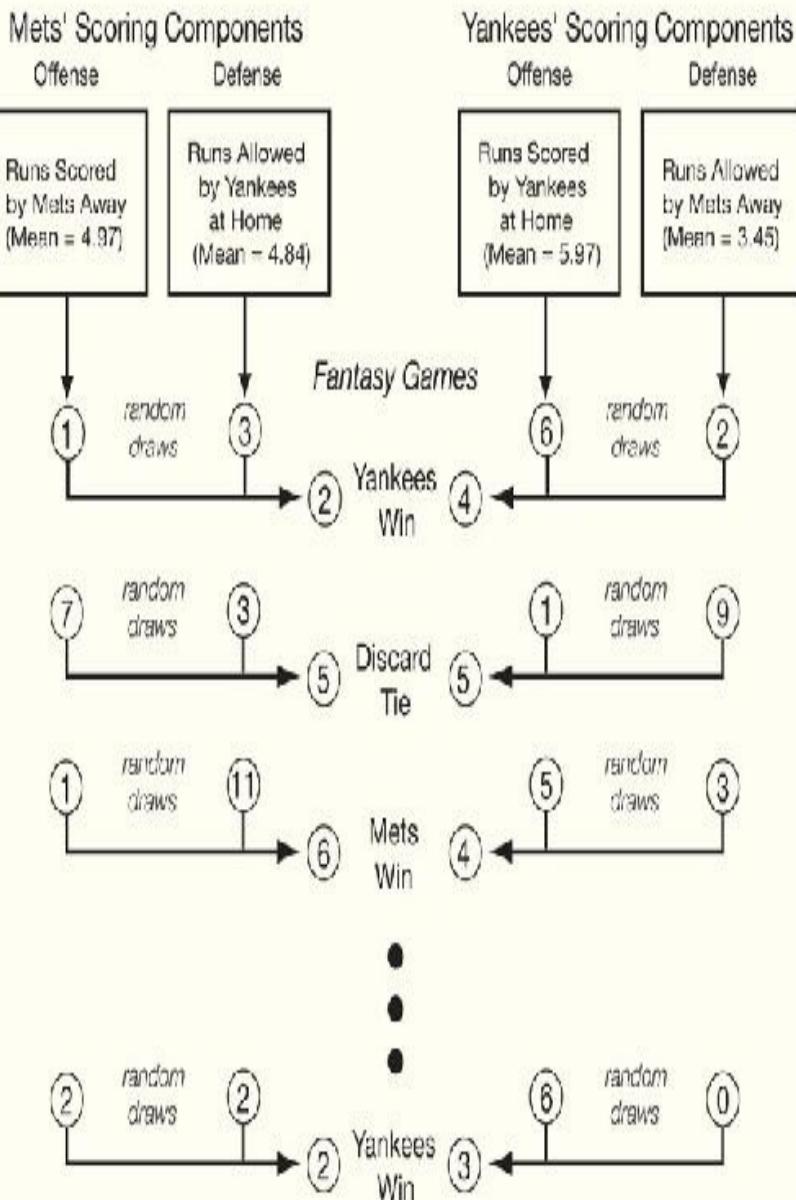
We see that one way to pick a winning team is to simulate play between teams. We should not expect too much from this simple game-day simulation. Knowing visiting and home teams, where the game is played, and past team scores is only the beginning. Our simulation ignores the fact that the Mets play most of their games in National League parks without a designated hitter and the Yankees play most of their games in American League parks with a designated hitter. Furthermore, this first simulation is focused solely on offense, considering runs scored by each team with no recognition of runs allowed. It gives a picture of offense without defense. Based only on runs scored, the Yankees look like they could beat the Mets. To get a better picture of how the Mets and Yankees stack up against each other, we juxtapose the teams' offensive and defensive performance data. For games played at Yankee Stadium, we set the Mets' away offense against the Yankees' home defense and the Yankees' home offense against the Mets away defense. See figure 10.2.



Data drawn from Mets and Yankees games April 1–June 14, 2007.

Figure 10.2. Mets' Away and Yankees' Home Data (Offense and Defense)

A balanced simulation would consider both offense and defense, as shown in [figure 10.3](#), with offense being runs scored and defense being runs allowed (runs scored by the other team). Here we take random drawings from the empirical distributions of runs scored and allowed by opposing teams to estimate expected runs scored by each team. Our approach in this example is to average opposing team offensive and defensive numbers. Taking both offense and defense into consideration, the Mets and Yankees appear to be more evenly matched, with the Mets' probability of winning 0.52 and the Yankees' probability of winning 0.48.



Results from 50,000 fantasy games without ties:

26,011 Mets' wins → 0.52 probability of winning

23,989 Yankees' wins → 0.48 probability of winning

Figure 10.3. *Balanced Game-day Simulation (Offense and Defense)*

What actually happened in these three interleague games? Friday evening, June 15, 2007 the Mets beat the Yankees 2 to 0. The following day the Yankees beat the Mets 11 to 8. And on Sunday, June 17, the Yankees won again, beating the Mets 8 to 2.

When working with the number of runs scored (a count), we use probability distributions such as the Poisson and negative binomial distributions. Using a Poisson distribution to represent runs scored, zero-run games are rare events, especially for teams scoring an average of three or more runs per game.

Baseball scores of more than twenty runs are rare events for teams scoring fewer than seven runs a game on average. The Poisson distribution reflects this fact.¹

¹ Keller (1994) showed that the Poisson probability distribution can be used as a model for baseball scores. The Poisson model works best when the rate or probability of run scoring does not vary substantially from game to game, which might make sense when working with a single team using a stable lineup from day to day. The word *Poisson*, pronounced “poy san,” refers to Siméon D. Poisson (1781–1840), who published the first derivation of the distribution in 1837. The Poisson probability distribution is given by the formula:

$$P(x) = \frac{e^{-\lambda} \lambda^x}{x!}$$

where e is the exponential constant and λ (Greek letter lambda) is its one parameter, giving both the mean and variance of the distribution. With no runs scored, the probability is $P(0) = e^{-\lambda}$. For one run scored, $P(1) = P(0)\lambda$. And generalizing, for x runs scored, $P(x) = P(x - 1) \frac{\lambda}{x}$. Those interested in the mathematics behind probability distributions as well as other topics in probability theory may refer to Feller (1968).

For the 2007 baseball season, taking runs scored across all teams and all games between April 1 and July 8 (prior to the

All Star Game), we obtain 2,598 observations and an average or mean runs scored of 4.68. The frequency distribution of these runs-scored observations is shown in the upper left-hand panel of figure 10.4. This figure also shows the relative frequency distribution and theoretical probabilities for a Poisson distribution with mean 4.68 and for a negative binomial distribution with mean 4.68 and shape parameter 4.00. Notice that for these runs-scored data the negative binomial appears to provide a better fit or match. This is especially apparent at the low end of the distribution.

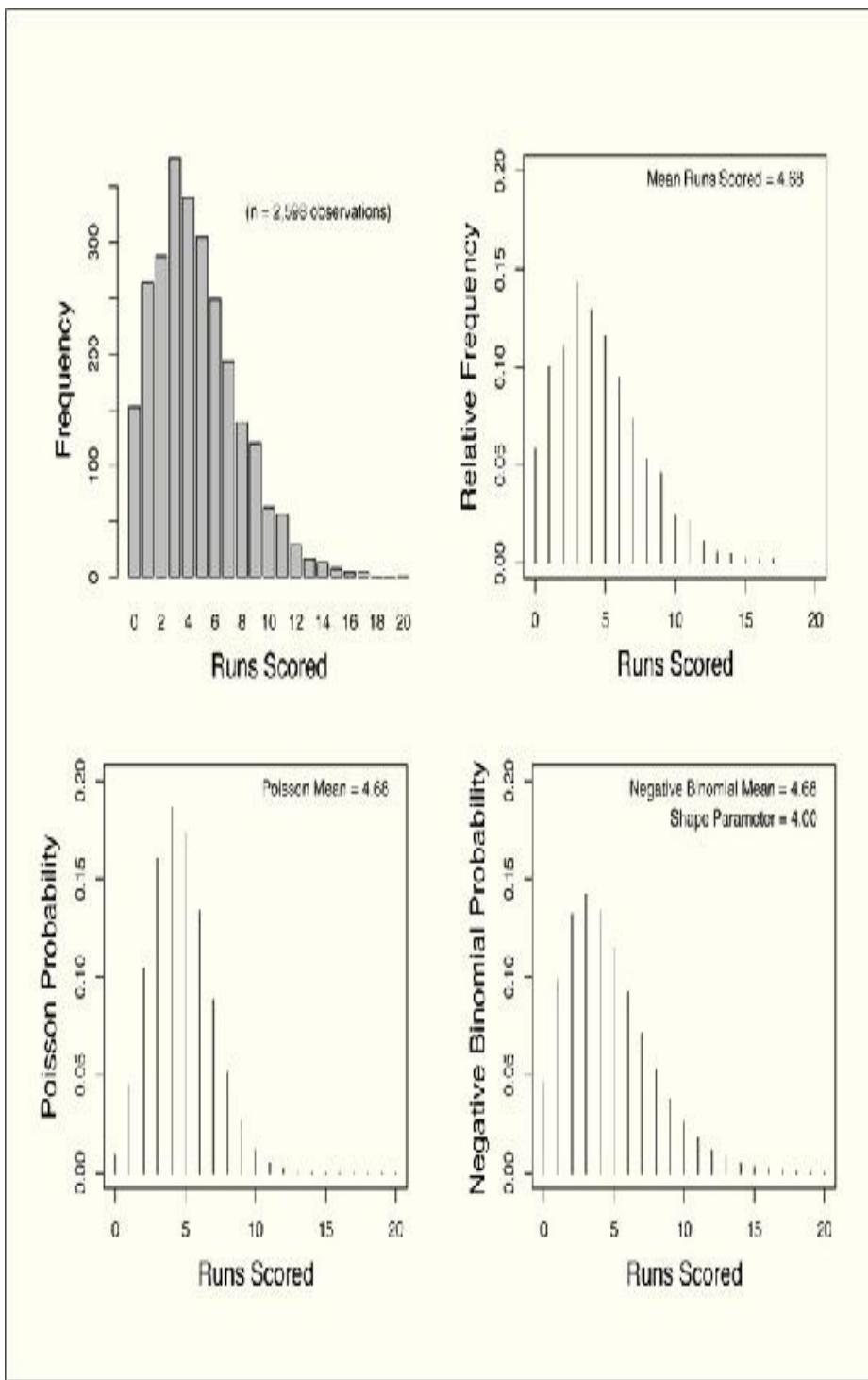


Figure 10.4. Actual and Theoretical Runs-scored Distributions

Mean runs-scored and shape ($k = 4$) are the defining parameters of a negative binomial probability distribution. The

negative binomial distribution represents a good model for runs-scored distributions when the rate or probability of scoring varies from one game to the next, as it would for a group of teams or a team that shows considerable variability in its performance across time. The negative binomial distribution has been shown to be useful for modeling scores in various sports (Reep, Pollard, and Benjamin 1971; Pollard 1973). Again, we refer to Feller (1968) for probability theory and use the computer to do the work. Using a negative binomial distribution to represent runs scored, shutouts are not so rare as with the Poisson distribution. Higher scores are also more common with the negative binomial than with the Poisson distribution.

Another way to execute a game-day simulation is to use theoretical probability distributions. For example, we could simulate Mets-Yankees' games as paired random drawings from a theoretical probability distribution, with the stipulation that drawings with ties be discarded. That is, we draw random pairs of observations from the Mets' away and Yankees' home distributions, executing a balanced simulation of offensive and defensive performance with probability models for runs scored. We often use the negative binomial distribution to represent runs scored in baseball.²

² The negative binomial distribution may be thought of as the reverse or flip-side of the binomial. While the binomial distribution gives the number of successes in n trials, the negative binomial gives the number of trials x until we observe some number of success. The negative binomial may be derived as a mixture of Poisson distributions and is sometimes called the compound Poisson distribution. It is defined by two parameters, the mean μ and the shape parameter k . The probability of observing x runs in a game is given by the formula

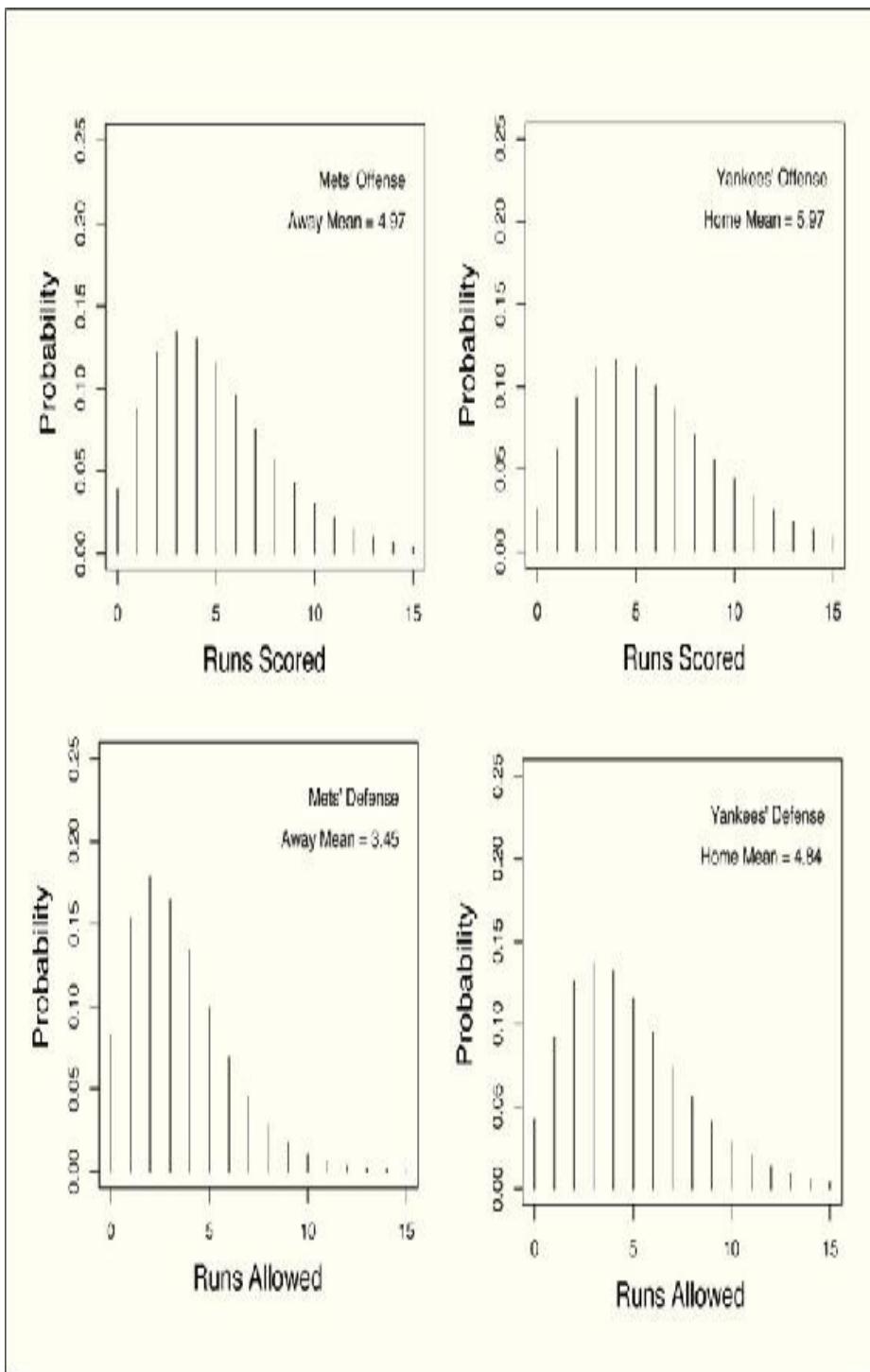
$$p(x) = \left(1 + \frac{\mu}{k}\right)^{-k} \frac{(k+x-1)!}{x!(k-1)!} \left(\frac{\mu}{\mu+k}\right)^x$$

For zero runs scored, we obtain a much simpler expression:

$$p(0) = \left(1 + \frac{\mu}{k}\right)^{-k}$$

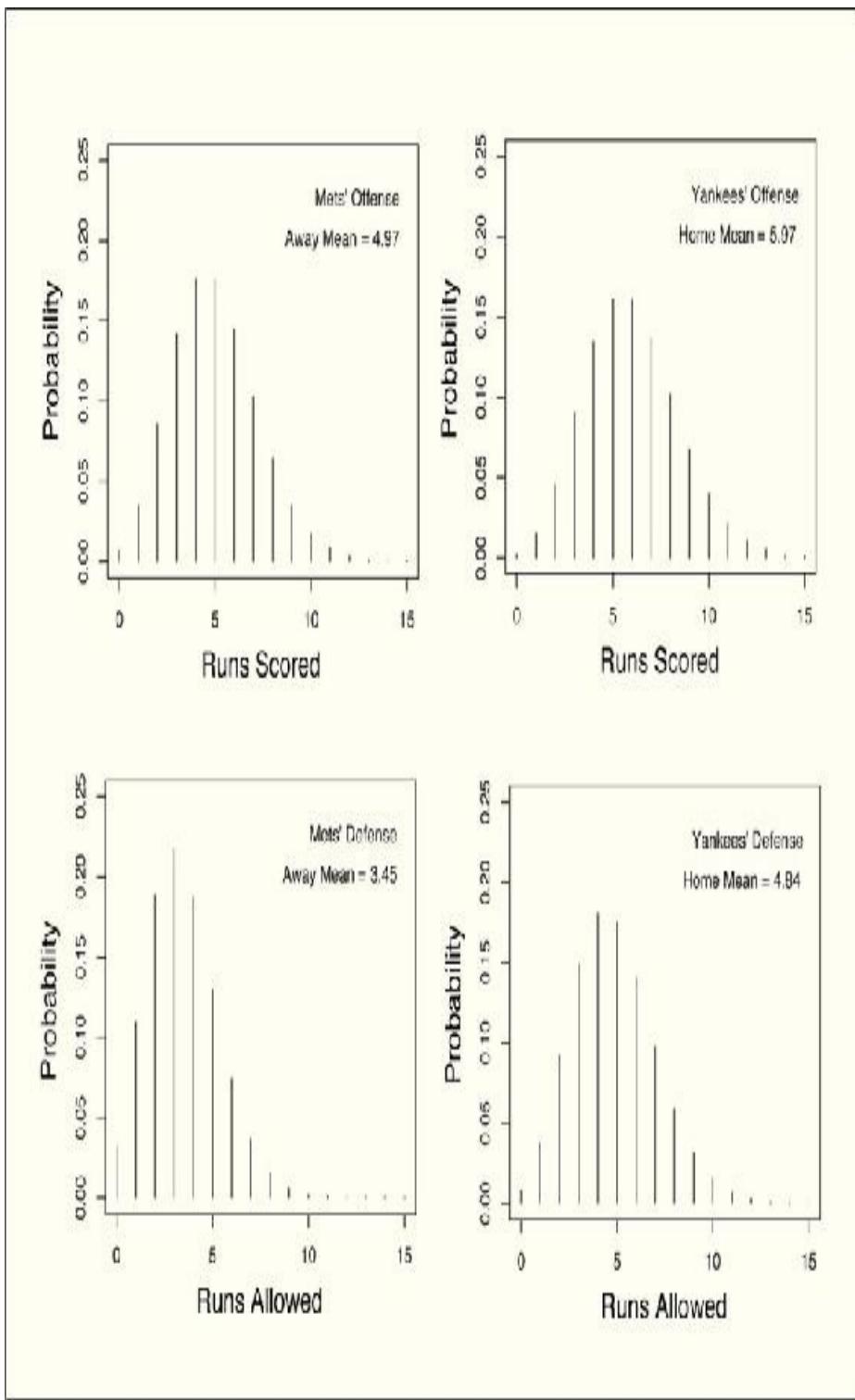
The negative binomial distribution is especially useful for working with counts for which the variance is larger than the mean, a situation called *overdispersion*. The shape or dispersion parameter k may be estimated by comparing the mean and variance of the observed counts. Our research with baseball runs scored suggests that a shape parameter k between 3 and 5 works well for the number of runs scored by a group of major league baseball teams. For the examples in this chapter we set k to 4, which was determined to be a good fit to the actual runs-scored data.

We show a Poisson model for how the Mets and Yankees stack up against each other in [figure 10.5](#). The corresponding negative binomial model is shown in [figure 10.6](#). With the Poisson model, executing the Mets-Yankees simulation for 50,000 games without ties, we observe the Mets beating the Yankees in 26,863 games, obtaining an estimated probability of 0.54.



Poisson means estimated from Mets and Yankees games April 1–June 14, 2007.

Figure 10.5. Poisson Model for Mets vs. Yankees at Yankee Stadium



Negative binomial means estimated from Mets and Yankees games April 1–June 14, 2007.

The negative binomial shape parameter k set to 4.

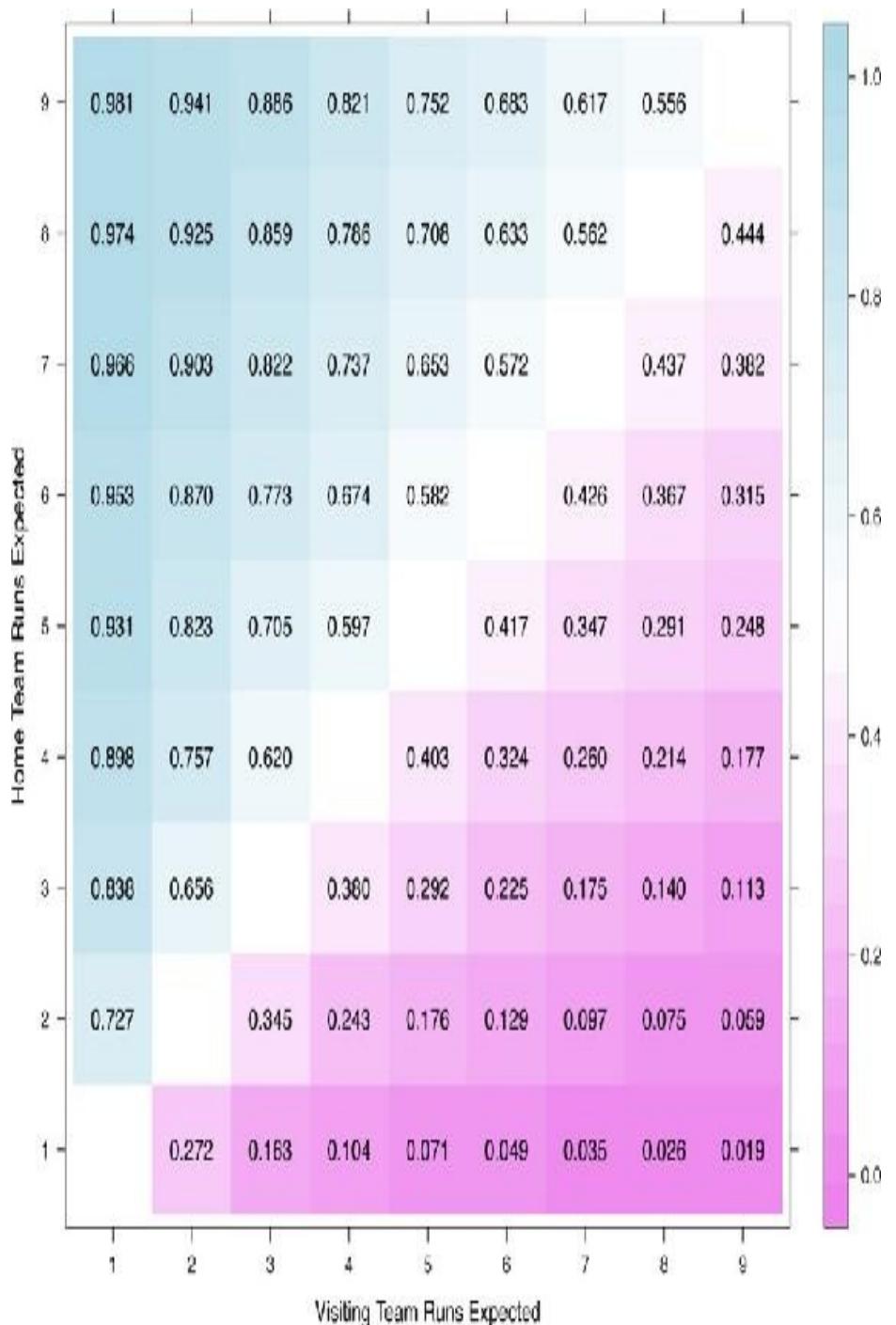
Figure 10.6. Negative Binomial Model for Mets vs. Yankees

at Yankee Stadium

Moving away from the Poisson model and using a negative binomial model with shape parameter k set to 4, we observe the Mets beating the Yankees in 26,202 of 50,000 games without ties, obtaining an estimated probability of 0.52. The Poisson estimate is close to the estimate from the balanced empirical runs-scored simulation, and the negative binomial estimate, rounded to two significant digits, is the same as the estimate from the empirical simulation. Both simulations predict the Yankees winning about two out of three games.

When we use game-day simulator to estimate a team's probability of winning, we are employing a model-dependent approach to predictive modeling. We generate data from a model and note how well they conform to real or empirical data. The Poisson and negative binomial distributions conform well, so we use them to build models. We can use them to estimate the probability that one team will beat another and to pick a winning team. The simulator is model-dependent, but the inputs to the simulator come from predictive models fit to observed data.

Figure 10.7 shows results from a probability simulator for home and away teams scoring between one and nine runs each, again using a negative binomial probability model with shape parameter k set to 4. This probability heat map is the result of running $\frac{(9 \times 8)}{2}$ or 36 game-day simulations, with each simulation consisting of 100,000 games.



The negative binomial shape parameter k set to 4.

Figure 10.7. Probability of Home Team Winning (Negative Binomial Model)

Precise mathematical calculations and well-designed simulations are the easy part of predictive inference in sports.

We have algorithms and computer programs to do the work. What is not so easy is obtaining the runs-scored estimates for input to the programs.

Do not underestimate the difficulty of predicting which team will win the next game. Predictive analytics is a precarious enterprise. It is a process of extrapolating and forecasting. In sports, it takes a very good model to do a better job than the bookmakers, to win against the spread and cover betting fees in the process. Accurate models for predicting the outcome of sporting events may be elusive. Nonetheless, it is comforting to know that we have a plan for finding accurate models if indeed they do exist.

A game day simulation is a statistical simulation, thinking of the game as a single complete event. It is useful to anyone interested in predicting game outcomes, including coaches, managers, owners, and sports betters. By changing the composition of a team, an owner can influence a team's scoring, an opponent's scoring, and the outcome of a game. The input to the model and the simulator consists of data for a newly constructed or hypothetical team. We predict average runs scored for the teams and run the game-day simulation.

What we show in Mets/Yankees simulation considers a game as a single, complete event. Suppose we break a game into its component parts, play-by-play events, model each event individually, and then observe simulated outcomes from a sequence of events. This is the idea behind methods known as discrete event simulation.

Every play in a game has alternative outcomes, setting the stage for the next play. Plays occur in sequence until time runs out in a time-limited game, such as football, basketball, or

soccer, or until a winning score is obtained in an untimed game, such as baseball or tennis. Games may include overtime periods or extra innings, and tied games are permitted in some circumstances. The contingencies of each game define the structure of the discrete event simulation.

We can think of discrete event simulation as a micro-model. We define the characteristics of players and their interactions with one another. In baseball, for example, we note possible outcomes when a particular pitcher faces a particular hitter. These outcomes have a multinomial probability distribution. For each simulation run, one of those outcomes is observed, setting the stage for the next pitcher/hitter interaction.

Methods of discrete event simulation can be especially useful in providing advice to coaches and managers regarding game-day decisions. Alternative game-day, in-game strategies may be explored by running the simulation thousands of times under each alternative strategy. A coach or manager can observe the range of scoring outcomes and select the outcome that provides the highest probability of winning.

Of course, every coach or manager has an opposing coach or manager equally capable of exploring alternative strategies. As we model this battle of coaching minds, marrying discrete event simulation with the mathematics of game theory, our computer programs take on additional complexity.

With agent-based models, we anticipate the future of sports analytics and data science. Individual players differ from one another in athleticism, playing technique, positioning on the field, and objectives of play as defined by managerial in-game strategies. Agent-based models present a potentially rich platform for modeling complexity, most useful in exploring

alternative in-game strategies.

Figure 10.8 provides an overview of strategic modeling techniques, beginning with mathematical models and ending with agent-based models. With mathematical models, known solutions follow from mathematics and probability theory—they are proven to be correct. Mathematical models provide the foundation of in-game strategic analysis as we have seen with run expectancy and state-to-state transition probabilities in baseball.

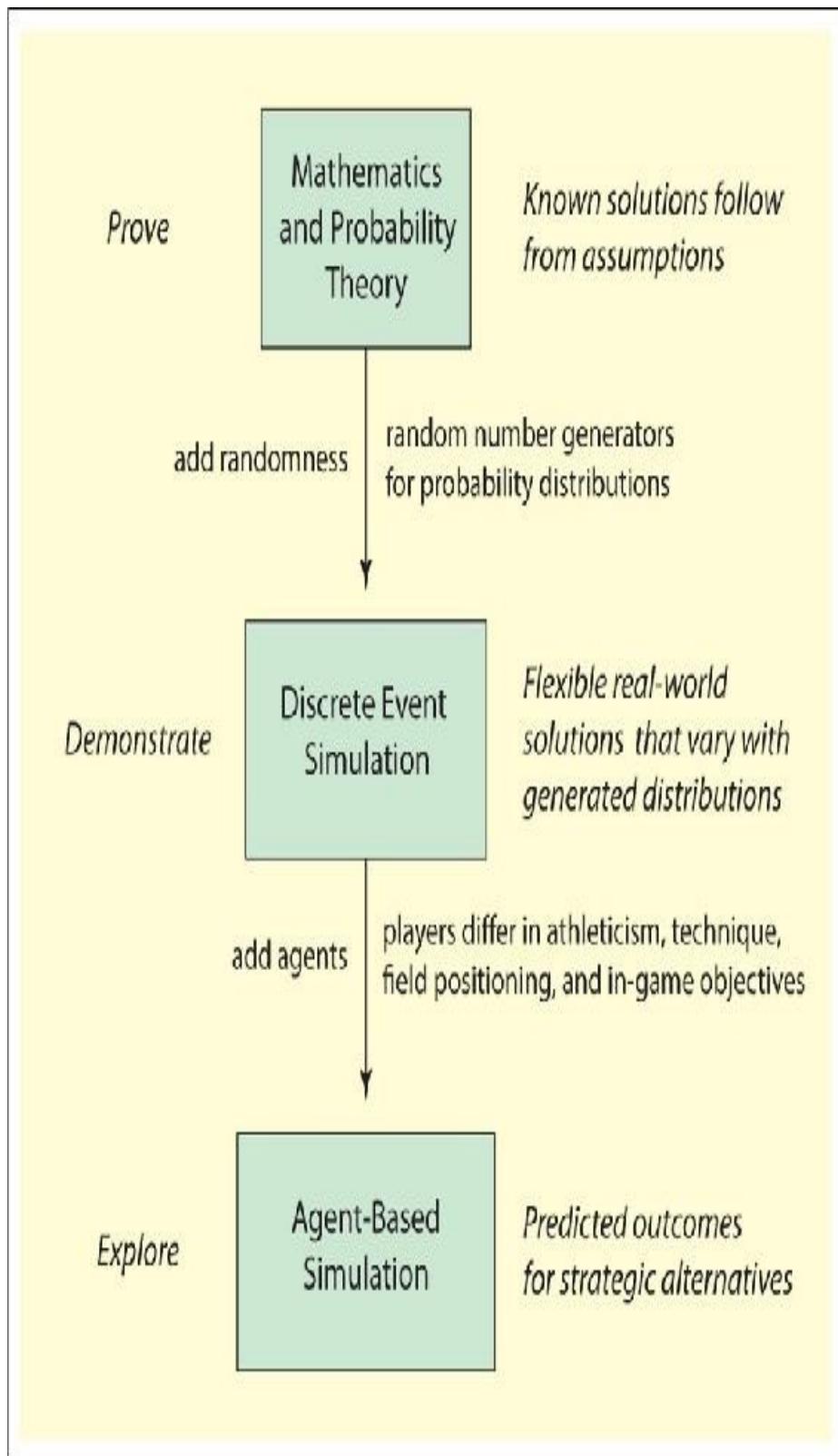


Figure 10.8. Strategic Modeling Techniques in Sports

Discrete event simulations, as demonstrated with the Mets-Yankees example, compute the likelihood of events by repeatedly sampling from probability distributions. Agent-based modeling builds on discrete event simulation.

Acknowledging the fact that individual players differ from one another in athleticism, playing technique, positioning on the field, and objectives of play, we have an opportunity to explore alternative in-game strategies and fine-tune those strategies.

Discussions of agent-based models and complexity go hand-in-hand. This is because agent-based models are most useful when the entire system is less well understood than its individual components. So rather than build a complex systems model, we build many small models. Agent-based models also have a role to play when underlying processes are coincidental rather than sequential. This is precisely the type of world we experience in sport with many players moving simultaneously across playing fields.

Simulations and the methods of data science are data-driven rather than expert-or rule-driven. And, being data-driven, we can test their accuracy before we use them to make decisions about players, teams, and sports business marketing or operations.

To move from the academic world into sports—to be viewed as more than a modeler’s playground—agent-based models need to do things that are useful and understood by people outside the modeling community. Validation of agent-based models presents challenges. Will it be sufficient to see the simulation acting as an actual game? And how shall we judge the degree of similarity between a simulation and an actual

game?

For reading in probability, refer to the Mosteller, Rourke, and Thomas (1970), the classic references of Feller (1968, 1971), and (for fun) a book of problems by Mosteller (1965). A Bayesian perspective is presented by Robert (2007), Albert (2009), and Hoff (2009).

Ross (2014) provides an introduction to probability modeling. Generating probability distributions and simulation programming in R are discussed by Robert and Casella (2009), Suess and Trumbo (2010), Chihara and Hesterberg (2011), Grolemund (2014), and Jones, Maillardet, and Robinson (2014). Law (2014) discusses the domain of simulation modeling, including discrete event simulation and agent-based modeling.

Much work with agent-based models has been biologically inspired, modeling the behavior of individual organisms to see what happens in a community (Resnick 1998; Mitchell 2009).

For discussion of complexity theory and agent-based modeling, see North and Macal (2007), Miller and Page (2007), and Šalamon (2011).

An R program for the baseball probability simulator is shown in exhibit 10.1. It draws on a graphics package developed by Sarkar (2014). A comparable Python program is shown in exhibit 10.2

Exhibit 10.1. Team Winning Probabilities by Simulation (R)

[Click here to view code image](#)

```
# Game-day Simulator for Baseball (R)
library(lattice) # graphics package for
```

```

probability matrix visual
simulator <-
function(home_mean,away_mean,niterations) {
    # input runs scored means, output
    probability of winning for home team
    set.seed(1234) # set to obtain
    reproducible results
    away_game_score <- numeric(niterations)
    home.game.score <- numeric(niterations)
    home_win <- numeric(niterations)
    i <- 1
    while (i < niterations + 1) {
        away_game_score[i] <-
        rnbinom(1,mu=away_mean, size = 4)
        home.game.score[i] <-
        rnbinom(1,mu=home_mean, size = 4)
        if(away_game_score[i] >
        home.game.score[i]) home_win[i] <- 1
        if(away_game_score[i] >
        home.game.score[i] ||

        away_game_score[i] <
        home.game.score[i]) i <- i + 1
    }
    n_home_win <- sum(home_win)
    n_home_win/niterations # return
    probability of away team winning
}
niterations <- 100000 # use smaller number
for testing
# probability matrix for results... home team
is rows, away team is columns
probmat <- matrix(data = NA, nrow = 9, ncol =
9,
    dimnames = list(c(as.character(1:9)),
c(as.character(1:9))))
for (index_home in 1:9)
for (index_away in 1:9)
if (index_home != index_away) {
    probmat[index_home,index_away] <-
    simulator(index_home, index_away,

```

```
niterations)
    }
pdf(file =
"fig_sports_analytics_prob_matrix.pdf", width
= 8.5, height = 8.5)
x <- rep(1:nrow(propmat),times=ncol(propmat))
y <- NULL
for (i in 1:ncol(propmat)) y <-
c(y,rep(i,times=nrow(propmat)))
probtext <- sprintf("%0.3f",
as.numeric(propmat)) # fixed format 0.XXX
text_data_frame <- data.frame(x, y, probtext)
text_data_frame$probtext <-
as.character(text_data_frame$probtext)
text_data_frame$probtext <-
ifelse((text_data_frame$probtext == "NA"),
NA,text_data_frame$probtext) # define
diagonal cells as missing
text_data_frame <- na.omit(text_data_frame) # 
diagonal cells
print(levelplot(propmat, cuts = 25,
tick.number = 9,
col.regions=colorRampPalette(c("violet",
"white", "light blue")),
xlab = "Visiting Team Runs Expected",
ylab = "Home Team Runs Expected",
panel = function(...) {
    panel.levelplot(...)
    panel.text(text_data_frame$x,
text_data_frame$y,
labels = text_data_frame$probtext)
}}))
dev.off()
```

**Exhibit 10.2. Team Winning Probabilities by Simulation
(Python)**

[Click here to view code image](#)

```

# Game-day Simulator for Baseball (Python)

from __future__ import division,
print_function

import numpy as np
from scipy.stats import nbinom

def simulator(home_mean, away_mean,
niterations):
    # estimates probability of home team win
    seed(1234) # set to obtain reproducible
results
    home_game_score = [0] * niterations
    away_game_score = [0] * niterations
    home_win = [0] * niterations
    i = 0
    while (i < niterations):
        home_game_score[i] = \
            nbinom.rvs(n = 4.0, p = 4.0/(4.0 +
home_mean), size = 1)[0]
        away_game_score[i] = \
            nbinom.rvs(n = 4.0, p = 4.0/(4.0 +
away_mean), size = 1)[0]
        if (home_game_score[i] >
away_game_score[i]):
            home_win[i] = 1
        if ((away_game_score[i] >
home_game_score[i]) or \
            (away_game_score[i] <
home_game_score[i])):
            i = i + 1
    n_home_win = sum(home_win)
    return n_home_win / niterations

niterations = 100000 # use smaller number for
testing
# probability matrix for results... home team
rows, away team columns
probmat = array([[0.0] * 9] * 9)

```

```
# matrix representation of home and away team
# runs for table
homemat = array([[9] * 9, [8] 9, [7] 9, [6]
9, [5] 9,\n    [4] 9, [3] 9, [2] 9, [1] 9])
awayrow = array([1, 2, 3, 4, 5, 6, 7, 8, 9])
awaymat = array([awayrow] * 9)

# generate table of probabilities
for index_home in range(9):
    for index_away in range(9):
        if (homemat[index_home, index_away] !=
awaymat[index_home, index_away]):
            print(index_home, index_away)
            probmat[index_home, index_away] =
\
                simulator(float(homemat[index_home
index_away]), \
                    float(awaymat[index_home
index_away]), niterations)

print(probmat)
```

11. Working with Sports Data

Ben: "You know what's really great about baseball?"

Lindsey: "Hmm?"

Ben: "You can't fake it. Anything else in life, you don't have to be great in—business, music, art—I mean, you can get lucky."

Lindsey: "Really?"

Ben: "Yeah, you can fool everyone for a while, you know. It's like—not—not baseball. You can either hit a curve ball or you can't. That's the way it works, you know?"

Lindsey: "Hmm."

Ben: "You know, you could have a lucky day, sure, but you can't have a lucky career. It's a little like math. It's orderly. Win or lose, it's fair. It all adds up. It's, like, not as confusing or as ambiguous as, uh—"

Lindsey: "Life?"

Ben: "Yeah, it's, it's safe."

—JIMMY FALLON AS BEN, AND DREW BARRYMORE AS LINDSEY

IN *Fever Pitch* (2005)

Information technology has undergone great change in recent years. Today's desktop and laptop computers have more power than mainframe computers of the past. Databases and analytics information systems are distributed across many computers in clusters or clouds. We carry the smallest of computers in our pockets. Watches, wearables, and data collection devices abound. Microprocessor chips and sensors

contribute to the glut of data in what is sometimes called the *Internet of Things*.

Sports teams have special requirements for working with physiological and athletic performance measurements. There are wearables, including wrist bands, patches, chest straps, and smart watches. Devices can monitor heart rate, body mass, activity levels, and physiological load. These devices may be used for training and lifestyle measurement. And when permitted by professional sports leagues, wearables may be used in games. Collected data, organized as longitudinal observations for each individual player, provide important information for fitness and conditioning staff, trainers, medical personnel, and coaches.

Professional sports leagues are working with commercial firms to collect in-game data relating to player and ball positions on the fields and courts of play. Major League Baseball tracks every pitch and every player on the field at a rate of thirty times per second. There are data for assessing fielding ability, including fielder distance covered, top speed, and route-to-the-ball efficiency. Fielder arm strength is assessed by noting the speed of balls thrown. Stealing bases may be better understood by looking at the distance of a runner's lead, top running speed, and time to the base being stolen. Batting is being assessed by noting exit velocity from the bat, launch angle, and hang time of balls hit. For pitchers, there are data being collected for position on the rubber, pitch release points, pitch speed, position of the ball within the strike zone, and spin of the ball (Gries 2015).

Commercial firms working with the National Basketball Association offer video and annotated videos, computer-

vision-aided observations and in-person observations for all games. Spatial data provide player and ball positions twenty-five times per second. The National Football League is using radio frequency identification (RFID) technology in conjunction with video to track player and ball positions in games. Timestamped spatial data offer considerable opportunities for spatio-temporal modeling and analysis.

Each professional sports team operates in the public arena. A team needs to know what people are saying about their team, their competitors, and the league. A team wants to know what fans are thinking and saying on talk radio and television. A team wants to follow sports commentators, writers, and analysts. A team wants to see what its players are posting on Twitter, Facebook, and other social media sites. It is difficult for public relations and marketing communications personnel to keep track of the many sources of information and media channels in today's world of 24/7 communication.

With these many devices and data sources and the networks that connect them, data are flowing into sports teams at unprecedented rates. Professional sports teams are struggling to keep pace. Automated, real-time data collection and analysis are needed.

Today's world of data science brings together statisticians fluent in R and information technology professionals fluent in Python. These communities have much to learn from each other. For the practicing sports analyst and data scientist, there are considerable advantages to being technically inclined. It pays to be multilingual, with some understanding of R and Python.

Designed by Ross Ihaka and Robert Gentleman, R first

appeared in 1993. R provides specialized tools for modeling and data visualization. It represents an extensible, object-oriented, open-source scripting language for programming with data. It is well established in the statistical community and has syntax, data structures, and methods similar to its precursors, S and S-Plus. Contributors to the language have provided more than five thousand packages, most focused on traditional statistics, machine learning, and data visualization. R is the most widely used language in data science, but it is not a general-purpose programming language.

Guido van Rossum, a fan of Monty Python, released version 1.0 of Python in 1994. This general-purpose language has grown in popularity in the ensuing years. Many systems programmers have moved from Perl to Python, and Python has a strong following among mathematicians and scientists. Many universities use Python as a way to introduce basic concepts of object-oriented programming. An active open-source community has contributed more than fifty-seven thousand Python packages.

We benefit from Python's capabilities as a general-purpose programming language. Sometimes referred to as a "glue language," Python provides a rich open-source environment for scientific programming and research. For computer-intensive applications, it gives us the ability to call on compiled routines from C, C++, and Fortran. We can also use Cython to convert Python code into optimized C.

Some problems are more easily solved with Python, others with R. For modeling techniques or graphics not currently implemented in Python, we can execute R programs from Python. We draw on R packages for traditional statistics, time

series analysis, multivariate methods, statistical graphics, and handling missing data. There are times as well when we move beyond Python and R to consider additional programming environments.

Java provides an object-oriented framework for building systems that run on many computer platforms. The Java Virtual Machine makes this possible. But Java is not the best choice for fast development or prototyping—Scala is better suited for that. And Scala can call on routines built on the Java Virtual Machine.

Compiled C and C++ programs may be good performers, but C and C++ are difficult languages in which to program. A new language called Go offers the high performance of a compiled language and the programming convenience of a scripting language such as Python or R. The Go language is also well suited for working within a multi-node, multiprocessor environment or cloud. If we were tasked with building a high-performance agent-based simulation from scratch, for example, we might turn to Go with its support for concurrency.

Technology solutions are best understood in layers—an information technology or software stack. Applications written in R or Python build on lower-level languages and databases, which depend on operating systems and software utilities.

Consider an information hierarchy or stack defining a real-time analytics system. One such system, an open-source system, is Apache Spark, which may be accessed by Python. Spark itself is built on the Scala language, which in turn is built on the Java Virtual Machine. Spark may be implemented within various software stacks and works with a number of

database and distributed file systems, including Hadoop.

Elasticsearch, an open-source, near-real-time indexing, search, and selection system, is especially useful for finding information in large document collections. Elasticsearch builds on Lucene, which in turn builds on the Java Virtual Machine. It provides its own distributed file system and application programming interface. As with Spark, Python client programs may be used to interact within the Elasticsearch stack, as shown in figure 11.1.

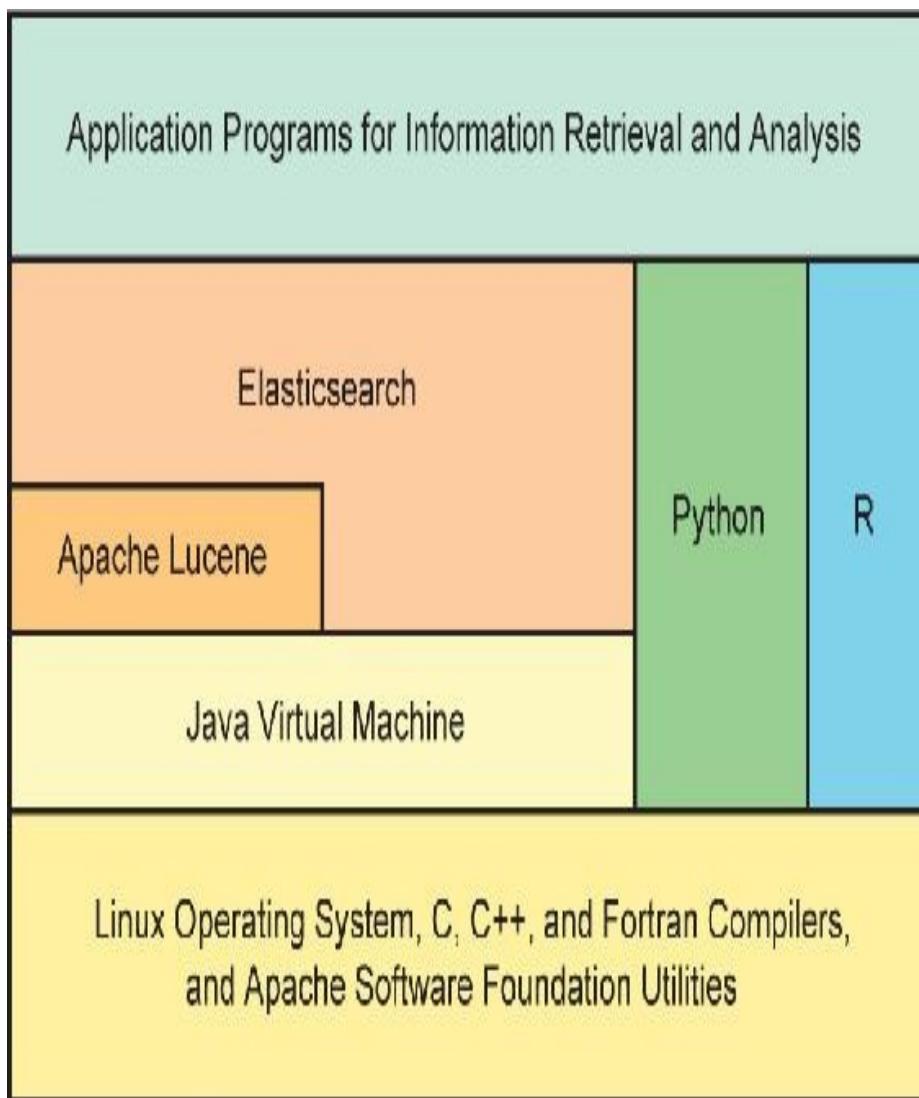
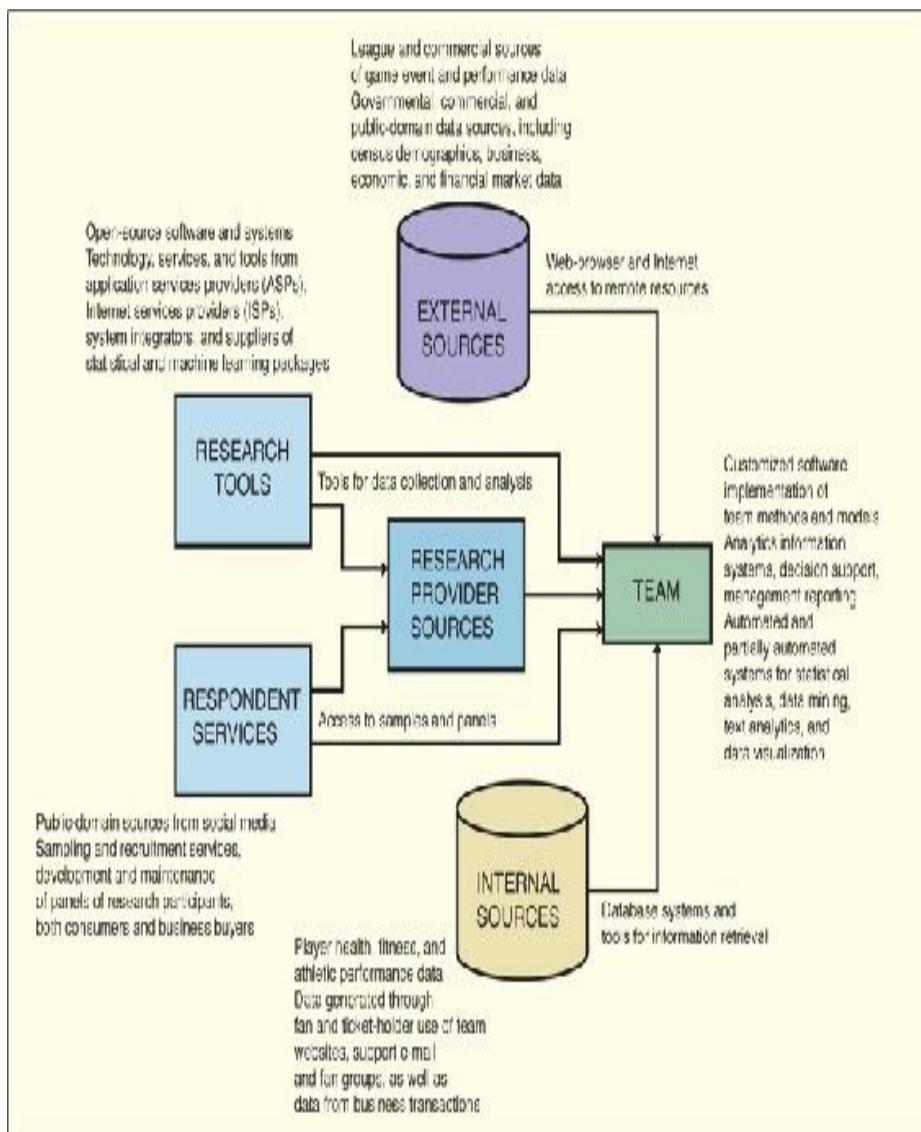


Figure 11.1. Software Stack for a Document Search and Selection System

Data science goes beyond the methods of traditional statistics, incorporating flexible data-adaptive methods. Data science addresses unstructured and semi-structured text as well as numerical data. Data science employs NoSQL document stores as well as spreadsheets and relational databases. And increasingly, data science provides methods for data exploration and discovery that help businesses benefit from large information stores. In a data-intensive, data-driven world, searching and selecting data have become as important as sampling.

Gathering and making use of information is what sports analysts and data scientists do every day. They understand information technology as well as statistical modeling. They work with data. They understand databases as well as box scores. They know about object-oriented programming and play-by-play logs.

Data scientists in professional team sports are knowledge workers and researchers. They rely on an information supply chain, as illustrated in figure 11.2. They utilize data sources, both internal and external, in serving the information needs of management, both on the coaching and player performance side and on the business side.



Adapted from Miller (2015a).

Figure 11.2. The Information Supply Chain of Professional Team Sports

A big part of the data scientist's job is finding information from the World Wide Web. The web represents a critical data source for sports teams, as it does for all organizations. The web holds the promise of unlimited information and connection. It is a huge data repository, a path to the world's knowledge, and the research medium through which we develop new knowledge.

The web's network foundation, the Internet, grew out of earlier wide-area networks, such as ARPANET and Usenet, serving the defense establishment, universities, and technical research centers. ARPANET was initially established for defense purposes, but became a wide-area network for many university researchers as well. Usenet linked many computers together in a simple telephone dial-up mode, permitting e-mail and file transfers. Standards-based networking protocols made Internet communications possible. Lower-level protocols defined how data could be transferred reliably over communication links. Higher-level protocols were developed for communication applications, such as telnet for terminal-to-computer communication and ftp for file transfers.

The World Wide Web arose from the marriage of hypertext and networking technologies. The idea for the World Wide Web was presented in 1989, while ideas behind hypertext date back to the 1960s. Hypertext—text with links to other text—represented a technology for developing dynamic documents, with the user determining the path between sections of text within documents. Hypertext markup language (HTML), originally designed as a text-only system, references files for graphics, audio, and video as well.

On the World Wide Web, hypertext links include uniform resource locators (URLs), which can refer to remote files and web pages on the network or to local files and web pages on the user's computer. Extensible markup language (XML) and JavaScript Object Notation (JSON) provide standards for information interchange.

We live in new world of research. Secondary research dominates primary research. Traditional scientific research,

with formally stated hypotheses followed by data collection, followed by tests of hypotheses, no longer rules the day.

Instead, businesses collect data—mountains of data—often without any idea of how they will use those data.

Firms such as Google, Microsoft, and Yahoo! crawl the entire web, building indexes for their search engines. Professional sports teams lack the computer resources of Google, Microsoft, and Yahoo!, but they are quite capable of conducting focused crawls to meet particular needs. Sports organizations also have the ability to develop business-and team-focused document stores, gathering data from application programming interfaces (APIs) to social media.

The World Wide Web is a huge data repository that changes with each moment. To address sports business problems, we must find our way to the right links, extracting relevant data, and analyzing them in ways that make sense.

We want data that are meaningful. We want data that are representative of a larger domain or population. And we want data that are relevant to the problem at hand. Finding our way to these data often presents a challenge, but this is an important part of data science.

Web crawling is faster than web surfing. It is automated. A crawler or spider does more than gather data from one web page. Like an actual spider in its web, a World Wide Web crawler or spider traverses links with ease. It follows web links from one web page to another, gathering information from many sources.

A focused crawl has a starting point or points, usually a list of relevant web addresses obtained from initial web surfing. A focused crawl has a stopping point. We can restrict the range

of the crawl to named domains, stop the crawl after a specified number of pages have been downloaded, set selection rules for pages being downloaded, and/or limit the total number of pages to be downloaded.

A focused crawl has a defined purpose. We retain those web pages that meet specific criteria. With the crawl well in hand, the task turns to scraping or extracting the specific information we want from web pages. Each web page includes HTML tags, defining a hierarchy of nodes (or tree structure). We call this tree the Document Object Model (DOM). We gather data from the the DOM, such as text within paragraph tags.

Figure 11.3 shows a framework for automated data acquisition on the web that is consistent with Python web crawling and scraping software developed by Hoffman *et al.* (2014). First we crawl, then we scrape, then we parse. This is the work of web-based secondary research. Next, we select an appropriate wrapping for text and associated metadata, JSON or XML perhaps. We build a document store or text corpus for subsequent analysis.

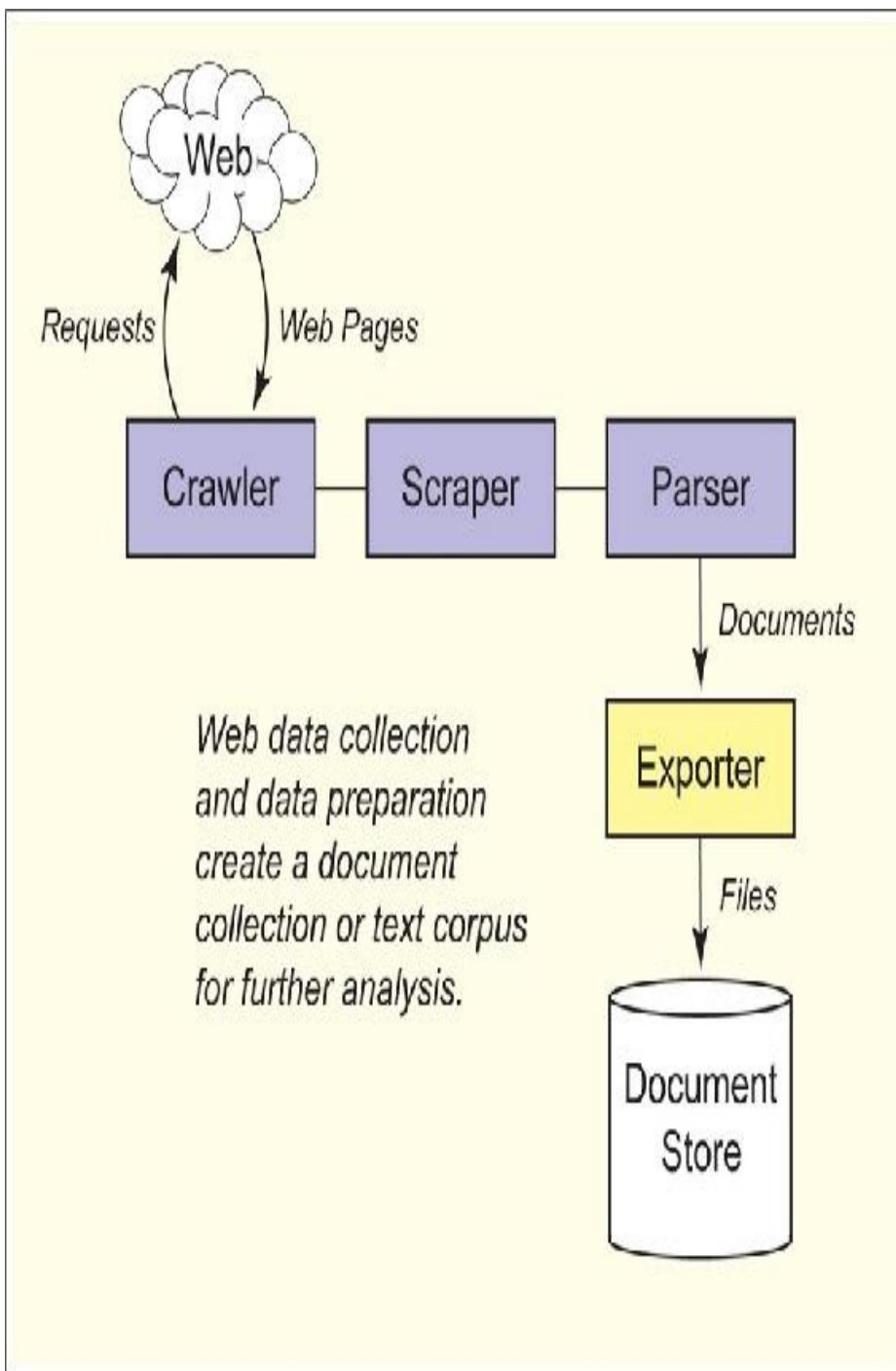


Figure 11.3. Automated Data Acquisition by Crawling, Scraping, and Parsing

Suppose we were working for the commissioner of a major sports league who is concerned about injuries affecting the game. We want to learn as much as we can about the science

of health and exercise physiology relating to injuries, so we search the web for answers. We want to consult experts in the areas of medicine and sports, so we search the web for contacts. Concerned that news about injuries (some career-ending and some with ramifications long after the end of an athlete's career) affects public opinion about the sport, we review postings to social media. We want to know if rule changes have had the beneficial effect of reducing injuries. We consult various sources regarding this problem, including published studies in medical journals and magazines. Our initial path to these sources is the web.

Crawling and scraping work well for data collection. Modeling and analysis begin with data, and the web is a massive store of data. Learning how to extract relevant data in an efficient manner is an essential skill of sports analytics and data science. After extracting text data from a web page, we can parse those data using *regular expressions*. Page formatting codes, unnecessary spaces, and punctuation can be removed from each document before that document is passed to the next step in text analysis.

Professional sports organizations see opportunities for communicating with fans through social media. Many of a team's key stakeholders, including high-visibility players, utilize social media, generating short text messages, personal statements, and blog posts.

Social network data sets from Twitter and Facebook are large but accessible. Obtaining these data involves using the various application programming interfaces (APIs) offered by social media providers. We illustrate the process of obtaining social media data in figure 11.4.

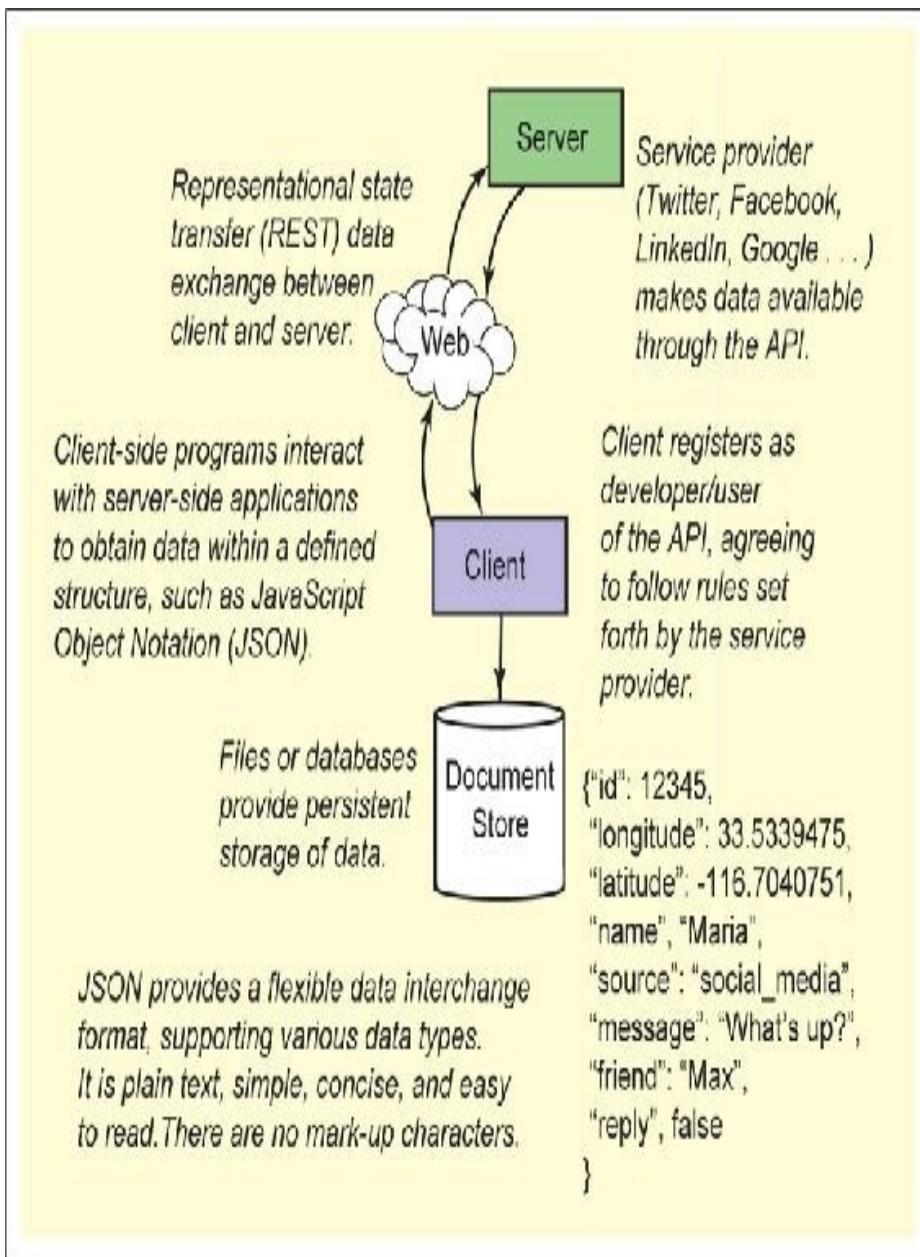


Figure 11.4. Automated Data Acquisition with an Application Programming Interface (API)

Social media provide two classes of data relevant to professional team sports. First, there are the connections among social media participants, which lead to social network analysis, and then there are the raw data of social media, text and images submitted by participants, which afford

opportunities for sentiment analysis or opinion mining.

How shall we deal with text data from the web and social media? At the very least we need to store them and index them so we can find them. Learning how to find relevant data in an efficient manner is an essential skill, and data collection and preparation can be the most time-consuming aspect of sports analytics and data science. Figure 11.5 provides a summary of the process as it relates to data gathered from the web.

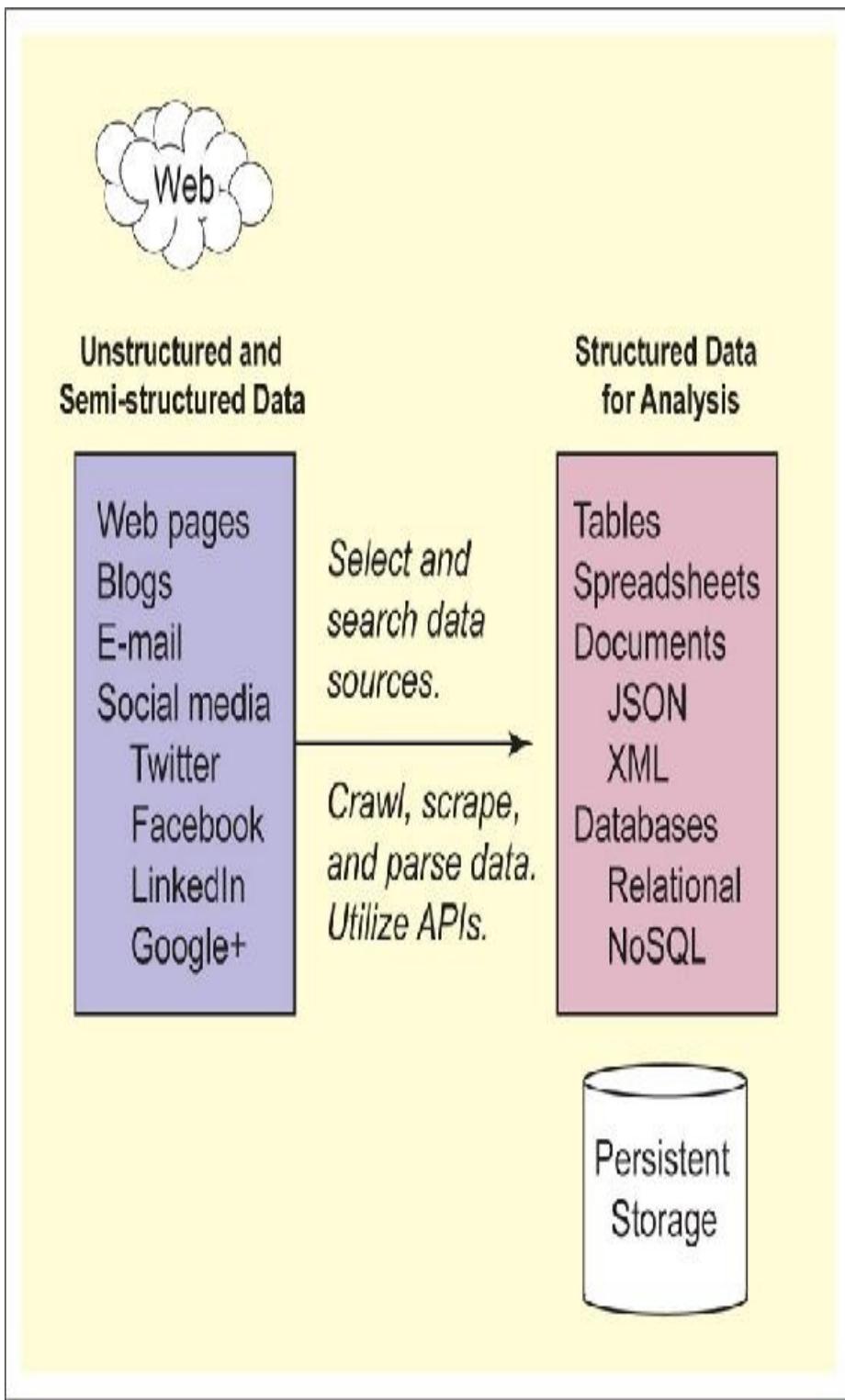


Figure 11.5. Gathering and Organizing Data for Analysis

There have always been more data than we have time to

analyze. What is new today is the ease of collecting data and the low cost of storing data. Data come from many sources. There are unstructured text data from online systems. There are pixels from sensors and cameras. There are data from wearables, mobile phones, tablets, and computers worldwide. Game events are located in space with field-of-play coordinates or geocoded by latitude and longitude and with timestamps showing date and time to the nearest milisecond. We need flexible, scalable, distributed systems to accommodate these data.

Doing data science means being able to gather data from the full range of database systems, relational and non-relational, commercial and open source. Relational databases have a row-and-column table structure, similar to a spreadsheet. We access and manipulate these data using structured query language (SQL). Because they are transaction-oriented with enforced data integrity, relational databases provide the foundation for sales order processing and financial accounting systems.

It is easy to understand why non-relational (NoSQL) databases have received so much attention. Non-relational databases focus on availability and scalability. They may employ key-value, column-oriented, document-oriented, or graph structures.

We employ database query and analysis tools, gathering information across distributed systems, collating information, creating contingency tables, and computing indices of relationship across variables of interest. We use information technology and database systems as far as they can take us, and then we do more, applying what we know about sports

analytics and data science.

We acknowledge an unwritten code in data science. We try to select data that are representative of all data (the population). We do not change data to conform to what we would like to see or expect to see. A two of clubs that destroys the meld is part of the natural variability in the game and must be played with the other cards. We play the hand that is dealt. The hallmarks of science are an appreciation of variability, an understanding of sources of error, and a respect for data. Data science is science.

Raw data are unstructured, messy, and sometimes missing. But to work well in models, data should be organized, clean, and complete. We are often asked to make a model out of a mess. Management needs answers, and the data are replete with miscoded and missing observations, outliers and values of dubious origin. We use our best judgement in preparing data for analysis, recognizing that many decisions we make are subjective and difficult to justify.

Missing data present problems in applied research because many modeling algorithms require complete data sets. With large numbers of explanatory variables, most cases have missing data on at least one of the variables. Listwise deletion of cases with missing data is not an option. Filling in missing data fields with a single value, such as the mean, median, or mode, would distort the distribution of a variable, as well as its relationship with other variables. Filling in missing data fields with values randomly selected from the data adds noise, making it more difficult to discover relationships with other variables. Multiple imputation is preferred by statisticians.

Doing data science with R means looking for task views

posted with the Comprehensive R Archive Network (CRAN). We go to RForge and GitHub. We read package vignettes and papers in *The R Journal* and the *Journal of Statistical Software*. The R programming environment consists of more than five thousand packages, many of them focused on modeling methods. Useful general references for learning R include Matloff (2011), Lander (2014), and Wickham (2015). Venables and Ripley (2002), although written with S/SPlus in mind, remains a critical reference in the statistical programming community.

Doing data science with Python means gathering programs and documentation from GitHub and staying in touch with organizations like PyCon, SciPy and PyData. At the time of this writing, the Python programming environment consists of more than fifty-seven thousand packages. There are large communities of open-source developers working on scientific programming packages like NumPy, SciPy, and SciKit-Learn. There is the Python Software Foundation, which supports code development and education. Useful general references for learning Python include Chun (2007), Beazley (2009), Beazley and Jones (2013), Lubanovic (2015), Slatkin (2015), and Sweigart (2015).

Ellis (2015) introduces ideas behind real-time analytics. Spark real-time analytics is discussed by Karau *et al.* (2015). See Chiusano and Bjarnason (2015) for an overview of Scala and Summerfield (2012) for an introduction to the Go language. Clojure is a modern Lisp-like language that works with the Java Virtual Machine (Fogus and Houser 2014). Scala, Go, and Clojure support concurrency, allowing several computations to execute at the same time, a desirable property

for distributed, high-throughput applications. Sports teams need to manage information in ways that allow efficient information retrieval at a later time. This is the task of automated search and document selection. Elasticsearch provides an integrated document storage and information retrieval system. To learn more about Elasticsearch, we can refer to Gheorghe, Hinman, and Russo (2015) and Gromley and Tong (2015). Information retrieval as a general topic is reviewed in Ceri *et al.* (2013). See Morville and Callender (2010) and Russell-Rose and Tyler-Tate (2013) for guidance regarding the design of search applications.

For the history of the World Wide Web, we can refer to its inventor, Tim Berners-Lee (2000). Further information about today's web may be obtained from the World Wide Web Consortium at www.w3.org. To learn more about crawling and scraping and automated data acquisition from the web, refer to Chakrabarti (2003) and Liu (2011).

Russell (2014) provides Python scripts for accessing social media APIs, and Mitchell (2015) provides Python examples of web crawling and scraping and working with APIs. Nolan and Lang (2014) present R scripts for gathering data from the web and working with those data. Miller (2015d) introduces web and network data science with Python and R examples.

Garcia-Molina, Ullman, and Widom (2009) and Connolly and Begg (2015) review database systems with a focus on the relational model. Worsley and Drake (2002) and Obe and Hsu (2012) review PostgreSQL. White (2011), Chodorow (2013), and Robinson, Webber, and Eifrem (2013) review selected non-relational systems. Copeland (2013) and Hoberman (2014) provide MongoDB document database examples. Dean

and Ghemawat (2004) and Rajaraman and Ullman (2012) discuss map-reduce operations for working with the Hadoop distributed file system.

Osborne (2013) provides an overview of data preparation issues, and the edited volume by McCallum (2013) provides much needed advice about what to do with messy data.

Missing data methods are discussed in various sources (Rubin 1987; Little and Rubin 1987; Schafer 1997; Lumley 2010; Snijders and Bosker 2012), with methods implemented in R packages from Gelman *et al.* (2014), Honaker, King, and Blackwell (2014), and Lumley (2014).

Many organizations are moving from internally-owned, centralized computing systems toward cloud-based services. When designing analytics information systems and document stores, flexibility and scalability are key. Distributed file systems and databases are preferred to single-server systems. Systems need to grow with the data. Bahga and Madisetti (2014) use Python examples to illustrate cloud computing. Most server systems utilize the Linux operating system. See Ward (2015) for an overview of Linux.

Open-source software and systems have been the focus throughout this chapter and this book. By using open-source software and systems, professional sports teams are better able to maintain ownership of their information technology and to protect their intellectual property. Professional sports teams would be well advised to avoid long-term commitments to commercial suppliers of software and systems. It is best to remain flexible and not to tie an organization into a single analytics platform or database system. Utilize open-source, standards-based systems whenever possible. An added benefit

to open-source systems, of course, is that they are less expensive than commercial solutions.

Commercial consulting and support services are available for most popular open-source systems. This is true for database systems like PostgreSQL and MongoDB. It is true for the Elasticsearch document store. As well, there are numerous commercial firms specializing in development and support of software built on Python and R platforms. Going with open-source systems does not mean “going it alone.”

Popular open-source languages and programming environments are also preferred because it is easy to find information technology professionals who already know these languages and environments. When hiring sports analysts and data scientists, teams can look for people who love programming and are willing to learn new things. Database systems and analytics technology change quickly, and it is important to keep pace with current developments.

Professional sports teams want people who understand data processing software and systems, as well as data analysis methods and models.

One research firm estimated a sports analytics market size of \$125 million in 2014, with expected growth to \$4.7 billion by 2021 ([WinterGreen Research 2015](#)). While seven years is a long forecasting horizon, especially for technology markets, we do expect substantial growth in sports analytics and data science. And growth implies investment in analytics information systems and people.

Discounted cash flow analysis, the basis of investment analysis reviewed in [chapter 9](#), applies equally well to investments in analytics. Analytics projects should have

positive net present value when evaluated across their expected lifetimes.

Exhibit 11.1 shows how to crawl and scrape a single website using a Python framework developed by Hoffman *et al.* (2014) and networking utilities from Lefkowitz *et al.* (2014). The program executes a focused crawl of www.toutbay.com, a data science services site maintained by the author. The program crawls the entire site, extracting text within paragraph tags and storing results in a plain text file.

The focus of exhibit 11.2 (page 189) is social media data acquisition. Many have expressed concerns about football injuries (Dwyre 2015), so we see what people have been saying about this topic. We utilize the Twitter REST API, drawing on well-documented code examples from Russell (2014). The word “REST” stands for “representational state transfer,” which means that we make requests for data using HTTP operators such as the GET operator. A uniform resource locator (URL) points to the web address for the Twitter service. Formatted text data are returned. We can request data downloads in JSON, a preferred text data interchange format for working on the web. An alternative to the approach taken here would be to use the Twitter Streaming API.¹

¹ To use programs that access social media data from Twitter, the programmer sets up a personal Twitter account and then obtains user credentials at <http://twitter.com/apps/new/>. The programmer employs these user credentials with each request posted to the API. This process applies to both the Twitter REST API and Streaming API. See Russell (2014) and Mitchell (2015) for further discussion of methods for working with Twitter.

Exhibit 11.1. Simple One-Site Web Crawler and Scraper (Python)

[Click here to view code image](#)

```
# Simple One-Site Web Crawler and Scraper
# (Python)
#
# prepare for Python version 3x features and
# functions
from __future__ import division,
print_function

# scrapy documentation at
http://doc.scrapy.org/

# workspace directory set to outer
# folder/directory wnds_chapter_3b
# the operating system commands in this
# example are Mac OS X

import scrapy # object-oriented framework for
crawling and scraping
import os # operating system commands

# function for walking and printing directory
# structure
def list_all(current_directory):
    for root, dirs, files in
os.walk(current_directory):
        level =
root.replace(current_directory,
'').count(os.sep)
        indent = ' ' 4 (level)
        print('}{}//'.format(indent,
os.path.basename(root)))
        subindent = ' ' 4 (level + 1)
        for f in files:
            print('}{}{}'.format(subindent, f))

# initial directory should have this form
# (except for items beginning with .):
```

```
#      sads_exhibit_11_1
#          run_one_site_crawler.py
#          scrapy.cfg
#          scrapy_application/
#              __init__.py
#              items.py
#              pipelines.py
#              settings.py
#              spiders
#                  __init__.py
#                  one_site_crawler.py

# examine the directory structure
current_directory = os.getcwd()
list_all(current_directory)

# list the available spiders, showing names to
be used for crawling
os.system('scrapy list')

# decide upon the desired format for exporting
output: csv, JSON, or XML

# run the scraper exporting results as a JSON
text file items.jsonlines
# this file provides text information with
linefeeds to provide
# text output that is easily readable in a
plain text editor
os.system('scrapy crawl TOUTBAY -o
items.jsonlines')

# output formats commented out (choose the one
needed for further parsing work)
# run the scraper exporting results as a
comma-delimited text file items.csv
# os.system('scrapy crawl TOUTBAY -o
items.csv')

# run the scraper exporting results as a JSON
```

```
text file items.json
# os.system('scrapy crawl TOUTBAY -o
items.json')

# run the scraper exporting results as a
dictionary XML text file items.xml
# os.system('scrapy crawl TOUTBAY -o
items.xml')

# -----
# MyItem class defined by
# items.py
# -----
# location in directory structure:
#
sads_exhibit_11_1/scrapy_application/items.py

# establishes data fields for scraped items

import scrapy # object-oriented framework for
crawling and scraping

class MyItem(scrapy.item.Item):
    # define the data fields for the item
    (just one field used here)
    paragraph = scrapy.item.Field() #
    paragraph content

# -----
# MyPipeline class defined by
# pipelines.py
# -----
# location in directory structure:
#
sads_exhibit_11_1/scrapy_application/pipelines.py

class MyPipeline(object):
    def process_item(self, item, spider):
```

```
    return item

# -----
# settings for scrapy.cfg
# settings.py
# -----
# location in directory structure:
#
sads_exhibit_11_1/scrapy_application/settings.py

BOT_NAME = 'MyBot'
BOT_VERSION = '1.0'

SPIDER_MODULES =
['scrapy_application.spiders']
NEWSPIDER_MODULE =
'scrapy_application.spiders'
USER_AGENT = '%s/%s' % (BOT_NAME, BOT_VERSION)
COOKIES_ENABLED = False
DOWNLOAD_DELAY = 2
RETRY_ENABLED = False
DOWNLOAD_TIMEOUT = 15
REDIRECT_ENABLED = False
DEPTH_LIMIT = 50

# -----
# spider class defined by
# script one_site_crawler.py
# -----
# location in directory structure:
#
sads_exhibit_11_1/scrapy_application/spiders/one_s

# prepare for Python version 3x features and
functions
from __future__ import division,
print_function
```

```
# each spider class gives code for crawling and
scraping

import scrapy # object-oriented framework for
crawling and scraping
from scrapy_application.items import MyItem # 
item class
from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import
LinkExtractor

# spider subclass inherits from BaseSpider
# this spider is designed to crawl just one
website
class MySpider(CrawlSpider):
    name = "TOUTBAY" # unique identifier for
the spider
    allowed_domains = ['toutbay.com'] #
limits the crawl to this domain list
    start_urls = ['http://www.toutbay.com'] #
first url to crawl in domain

    # define the parsing method for the spider
    def parse(self, response):
        html_scraper =
scrapy.selector.HtmlXPathSelector(response)
        divs = html_scraper.select('//div') #
identify all <div> nodes
        # XPath syntax to grab all the text in
paragraphs in the <div> nodes
        results = [] # initialize list
        this_item = MyItem() # use this item
class
        this_item['paragraph'] =
divs.select('.//p').extract()
        results.append(this_item) # add to
the results list
        return results
```

Exhibit 11.2. Gathering Opinion Data from Twitter: Football Injuries (Python)

[Click here to view code image](#)

```
# Gathering Opinion Data from Twitter:  
Football Injuries (Python)  
  
# prepare for Python version 3x features and  
functions  
from __future__ import division,  
print_function  
  
import twitter # work with Twitter APIs  
import json # methods for working with JSON  
data  
  
windows_system = False # set to True if this  
is a Windows computer  
if windows_system:  
    line_termination = '\r\n' # Windows line  
termination  
if (windows_system == False):  
    line_termination = '\n' # Unix/Linus/Mac  
line termination  
  
# name used for JSON file storage  
json_filename = 'my_tweet_file.json'  
  
# name for text file for review of results  
full_text_filename =  
'my_tweet_review_file.txt'  
  
# name for text from tweets  
partial_text_filename =  
'my_tweet_text_file.txt'  
  
# See Russell (2014) and Twitter site for  
documentation
```

```
# https://dev.twitter.com/rest/public
# Go to http://twitter.com/apps/new to provide
an application name
# to Twitter and to obtain OAuth credentials
to obtain API data

# -----
# Twitter authorization a la Russell (2014)
section 9.1
# Insert credentials in place of the "blah
blah blah" strings
# Sample usage of oauth() function
# twitter_api = oauth_login()
def oauth_login():

    CONSUMER_KEY = 'blah'
    CONSUMER_SECRET = 'blah blah'
    OAUTH_TOKEN = 'blah blah blah'
    OAUTH_TOKEN_SECRET = 'blah blah blah blah'

    auth = twitter.oauth.OAuth(OAUTH_TOKEN,
OAUTH_TOKEN_SECRET,
                                CONSUMER_KEY,
CONSUMER_SECRET)

    twitter_api = twitter.Twitter(auth=auth)
    return twitter_api

# -----
# searching the REST API a la Russell (2014)
section 9.4
def twitter_search(twitter_api, q,
max_results=200, **kw):
    # See
    https://dev.twitter.com/docs/api/1.1/get/search/tweets
    and
    #
https://dev.twitter.com/docs/using-search for
details on advanced
    # search criteria that may be useful for
```

```
keyword arguments

    # See
    https://dev.twitter.com/docs/api/1.1/get/search/tweets
        search_results =
    twitter_api.search.tweets(q=q, count=100,
**kw)

    statuses = search_results['statuses']

    # Iterate through batches of results by
    following the cursor until we
        # reach the desired number of results,
    keeping in mind that OAuth users
        # can "only" make 180 search queries per
    15-minute interval. See
        #
https://dev.twitter.com/docs/rate-limiting/1.1/limits
        # for details. A reasonable number of
    results is ~1000, although
        # that number of results may not exist for
    all queries.

    # Enforce a reasonable limit
    max_results = min(1000, max_results)

    for _ in range(10): # 10*100 = 1000
        try:
            next_results =
    search_results['search_metadata']['next_results']
            except KeyError, e: # No more results
    when next_results doesn't exist
            break

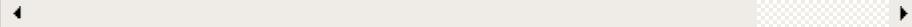
            # Create a dictionary from
    next_results, which has the following form:
            #
?max_id=313519052523986943&q=NCAA&include_entities
            kwargs = dict([ kv.split('=') for kv in
```



```
# is written as a JSON object on a separate
line
item_count = 0 # initialize count of objects
dumped to file
with open(json_filename, 'w') as outfile:
    for dict_item in results:
        json.dump(dict_item, outfile, encoding
= 'utf-8')
        item_count = item_count + 1
        if item_count < len(results):
            outfile.write(line_termination) #  
new line between items

# -----
# working with text file for reviewing
multiple JSON objects
# this text file will show the full contents
of each tweet
# results is a list of dictionary items
obtained from twitter
# these functions assume that each dictionary
item
# is written as group of lines printed with
indentation
item_count = 0 # initialize count of objects
dumped to file
with open(full_text_filename, 'w') as outfile:
    for dict_item in results:
        outfile.write('Item index: ' +
str(item_count) +\
        '
-----' +
line_termination)
        # indent for pretty printing
        outfile.write(json.dumps(dict_item,
indent = 4))
        item_count = item_count + 1
        if item_count < len(results):
            outfile.write(line_termination) #  
new line between items
```

```
# -----
# working with text file for reviewing text
from multiple JSON objects
# this text file will show only the text from
each tweet
# results is a list of dictionary items
obtained from twitter
# these functions assume that the text of each
tweet
# is written to a separate line in the output
text file
item_count = 0 # initialize count of objects
dumped to file
with open(partial_text_filename, 'w') as
outfile:
    for dict_item in results:
        outfile.write(json.dumps(dict_item['text']))
        item_count = item_count + 1
        if item_count < len(results):
            outfile.write(line_termination) #  
new line between text items
```



12. Competing on Analytics

“. . . in either game, life or football, the margin for error is so small. I mean, one-half a step too late or too early, and you don’t quite make it. One-half second too slow, too fast, and you don’t quite catch it. The inches we need are everywhere around us. They’re in every break of the game, every minute, every second. On this team we fight for that inch.”

—AL PACINO AS COACH TONY D’AMATO IN *Any Given Sunday*
(1999)

An alert runner on second base sees a pitch in the dirt and breaks for third. The ball hits the catcher’s chest protector and bounces a couple feet in front of the catcher. Surprised by the runner’s aggressiveness, the catcher rushes his throw to third, a throw that bounces off the third baseman’s glove and rolls into left field. The throwing error allows the runner to come home, scoring what might be called a manufactured run.

Small differences make all the difference in sports. It may be hard to document fully the manufactured runs and points attributed to sports analytics. The extra inches in football, assists and blocked shots in basketball, headers into soccer goals, or hockey face-offs won or lost by milliseconds—these represent the difference between winning and losing. And small differences justify investments in information technology and data science.

While detractors of sports analytics might imagine a dystopian future of data-and analytics-driven sports, most people in the know see higher levels of competition resulting from the

intelligent use of analytics. Improved measures aid player evaluation. Performance data and models guide player selection and game-day strategies. Medical data and models help players avoid injuries and recover from injuries more quickly. Coaches and managers who know what to expect from their decisions make better rosters and player assignments. They make informed choices about player positioning, competitive strategy, and in-game tactics.

Davenport and Harris (2007) used the term “competing on analytics” to describe an emerging ethos in business. In a data-intensive world, a world driven by information technology and communications, it makes sense to use data and models to guide business decisions. Nowhere in their book do Davenport and Harris suggest that models replace decision-makers. Rather their argument is that business managers do a better job of defining strategy and tactics when they are guided by analytics.

Those who would pit the opinions of scouts against the predictions of models are framing the question incorrectly. The question is not, “Should teams use scouts or sports analytics?” Scouts are not going away—they play an essential role in all sports, as do trainers and coaches. The question is, “Will scouts do a better job after being informed by analytics?” And the answer to this question is a resounding yes.

Our work with models and methods shows that sports analytics is more than a matter of defining new metrics. We must think more broadly, considering the extent to which performance measures may be used to make predictions about the future. Predictive inference is at the core of selecting the

right players for teams and helping players perform at their best.

We have argued that sports analytics, as it is commonly understood, is not sufficient. Teams are bombarded by data from sporting events and social media. They need to utilize information technology and the tools of data science if they are to remain competitive. To win the game with methods and models, teams need to utilize information systems that serve the needs of the entire organization—players, trainers, coaches, and business managers alike. The good news is that information systems designed for sports performance analytics are also applicable to sports management analytics.

General managers who understand the competitive advantage of analytics ensure that analytics information systems are in place to serve the data needs of player health and performance, team management, and business operations. These general managers ensure that there are appropriate staff members to do the work of analytics.

To compete on analytics effectively, to have competitive advantage, professional sports teams need to do more than hire one or two statisticians or economists. Teams need information technology professionals, database administrators, and data scientists, people who can work with data and program computers. Teams benefit by organizing in ways that promote creativity and learning from data. Decision-making should be informed by data, recognizing the limitations of measures and predictive models.

Doing sports analytics and data science means finding people who understand methods and models and can communicate with management. These are people who understand both the

performance and management sides of sports.

The number of analysts and data scientists in professional sports will grow in the coming years. Sports is “the winning business,” and sports teams do what they can to win. This means finding the most talented analysts and data scientists as well as the most talented players. It also means paying team members what they are worth.

Each sports team has a playbook to guide game-day decisions. That playbook is team-proprietary, a set of trade secrets shared only among coaches and players on the team. A similar strategy would seem to make sense regarding data science methods and models. Analytics competitive advantage emanates from the analysts and data scientists the team is able to hire and retain. Competitive advantage grows with the development of innovative methods and models.

Recommendations derived from data science methods and models contribute to the team’s playbook.

Front offices may be well advised to let all good ideas in, while keeping many good ideas to themselves. Letting good ideas in means drawing on the literature of data science and using open-source software and systems. Keeping good ideas to themselves means protecting proprietary measurements, methods, and models.

It was Thursday, June 18, 2015, and the Los Angeles Dodgers were playing an inter-league game with the Texas Rangers. It was a home game at Dodger Stadium with no score in the bottom of the ninth inning.

Rangers’ reliever Keone Kela walked Yasmani Grandal and Andre Ethier to begin the inning. Enrique “Kiki” Hernandez was running for Grandal when Alberto Callaspo grounded into

a 3-6-3 double play. “Kiki” went to third on the play. Jimmy Rollins, an excellent bunter with acceptable speed, but batting only 0.198 at the time, came to the plate. Dodgers manager Don Mattingly had many options to consider.

I may be a data scientist, sufficiently informed about event probabilities and expected runs. I may have a good idea of what a manager should call with two outs and a runner on third. But I am also a baseball fan.

On June 18, 2015, what I really wanted was to see “Kiki” Hernandez steal home. Whatever the probability of success, I wanted to be entertained.

As it turned out, a most improbable thing happened. “Kiki” jumped off third, faking a move to home plate. This disturbed the pitcher Kela, who committed a balk. “Kiki” was awarded home, and the Dodgers won the game—it was a walk-off balk ([Carr 2015](#)).

Using analytics to provide competitive advantage in no way reduces the excitement or joy of sports. Players and teams must still compete in matches and games. They fight for every inch, whether guided by analytics or not. As long as there is sport, there is the challenge, uncertainty, and joy of winning. And every once in a great while we see a player take a page from the Jackie Robinson playbook and steal home.

A. Data Science Methods

This book is different from other sports analytics books because it views sports analytics in the broader context of data science. Data scientists speak the language of business—accounting, finance, marketing, and management. They know about information technology, including data structures, algorithms, and object-oriented programming. They understand statistical modeling, machine learning, mathematical programming, and simulation methods. These are the things that data scientists do:

- **Information search and selection.** We begin by reviewing the research literature in the field, learning what others have done in the past. Then we search for relevant data sources and select sources for analysis and modeling.
- **Preparing text and data.** Text is unstructured or partially structured data that must be prepared for analysis. We extract features from text. We define measures. Quantitative data are often messy or missing. They may require transformation prior to analysis. Data preparation consumes much of a data scientist's time.
- **Looking at data.** We do exploratory data analysis, data visualization for the purpose of discovery. We look for groups in data. We find outliers. We identify common dimensions, patterns, and trends.
- **Predicting how much.** We are often asked to predict how many units or dollars of product will be sold, the

price of financial securities or real estate. Regression techniques are useful for making these predictions.

- **Predicting yes or no.** Many business problems are classification problems. We use classification methods to predict whether or not a person will buy a product, default on a loan, or access a web page.
- **Testing it out.** We examine models with diagnostic graphics. We see how well a model developed on one data set works on other data sets. We employ a training-and-test regimen with data partitioning, cross-validation, or bootstrap methods.
- **Playing what-if.** We manipulate key variables to see what happens to our predictions. We play what-if games in simulated marketplaces. We employ sensitivity or stress testing of mathematical programming models. We see how values of input variables affect outcomes, payoffs, and predictions. We assess uncertainty about forecasts.
- **Explaining it all.** Data and models help us understand the world. We turn what we have learned into an explanation that others can understand. We present project results in a clear and concise manner.

Prediction is distinct from explanation. We may not know why models work, but we need to know when they work and when to show others how they work. We identify the most critical components of models and focus on the things that make a difference.¹

¹ Statisticians distinguish between explanatory and predictive models.

Explanatory models are designed to test causal theories. Predictive models are designed to predict new or future observations. See Geisser (1993), Breiman

(2001), and Shmueli (2010).

Data scientists are methodological eclectics, drawing from many scientific disciplines and translating the results of empirical research into words and pictures that management can understand. These presentations benefit from well-constructed data visualizations. In communicating with management, data scientists need to go beyond formulas, numbers, definitions of terms, and the magic of algorithms. Data scientists convert the results of predictive models into simple, straightforward language that others can understand.

Data scientists are knowledge workers par excellence. They are communicators playing a critical role in today's data-intensive world. Data scientists turn data into models and models into plans for action.

The approach we have taken in this and other books in the *Modeling Techniques* series has been to employ both classical and Bayesian methods. And sometimes we dispense with traditional statistics entirely and rely on machine learning algorithms.

Within the statistical literature, Seymour Geisser introduced an approach best described as *Bayesian predictive inference* (Geisser 1993). In emphasizing the success of predictions in data science, we are in agreement with Geisser. But our approach is purely empirical and in no way dependent on classical or Bayesian thinking. We do what works, following a simple premise:

The value of a model lies in the quality of its predictions.

We learn from statistics that we should quantify our uncertainty. On the one hand, we have confidence intervals,

point estimates with associated standard errors, significance tests, and p -values—that is the classical way. On the other hand, we have posterior probability distributions, probability intervals, prediction intervals, Bayes factors, and subjective (perhaps diffuse) priors—the path of Bayesian statistics.

The role of data science in business has been discussed by many ([Davenport and Harris 2007](#); [Laursen and Thorlund 2010](#); [Davenport, Harris, and Morison 2010](#); [Franks 2012](#); [Siegel 2013](#); [Maisel and Cokins 2014](#); [Provost and Fawcett 2014](#)). In-depth reviews of methods include those of [Izenman \(2008\)](#), [Hastie, Tibshirani, and Friedman \(2009\)](#), and [Murphy \(2012\)](#).

Doing data science means implementing flexible, scalable, extensible systems for data preparation, analysis, visualization, and modeling. We are empowered by the growth of open source. Whatever the modeling technique or application, there is likely a relevant package, module, or library that someone has written or is thinking of writing. Doing data science with open-source tools is discussed in [Conway and White \(2012\)](#), [Putler and Krider \(2012\)](#), [James *et al.* \(2013\)](#), [Kuhn and Johnson \(2013\)](#), [Lantz \(2013\)](#), and [Ledoit \(2013\)](#). Additional discussion of data science, predictive modeling techniques, and open-source tools is provided in other books in the *Modeling Techniques* series ([Miller 2015a](#), [2015b](#), [2015c](#), and [2015d](#)).

This appendix provides an overview of data science methods, citing relevant sources for further reading. Topics include mathematical programming, classical and Bayesian statistics, regression and classification, machine learning, data visualization, text analytics, and time series analysis. The final

section shows how data science relates to other disciplines.

A.1 MATHEMATICAL PROGRAMMING

We use the term *mathematical programming* to refer to problems that involve constrained optimization. The word “programming” in this context means “planning,” as in “resource planning.” It does not mean computer programming, although we certainly use computer programs to solve constrained optimization problems. We specify constrained optimization in what is known as *standard form*. The objective or goal is to maximize the quantity z , which is a sum of n non-negative decision variables x_j :

$$\text{Maximize } z = \sum_j c_j x_j \quad (j = 1, 2, \dots, n)$$

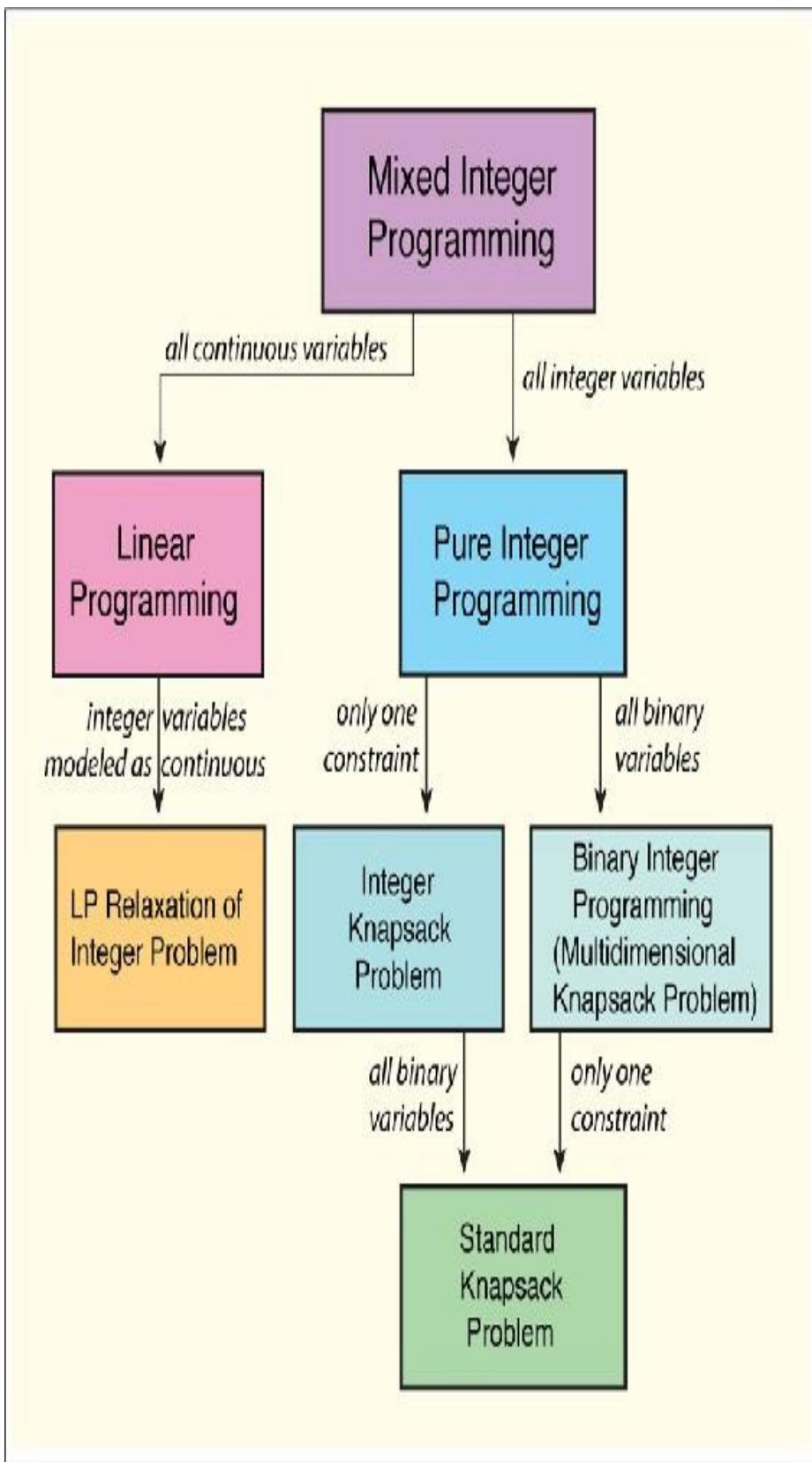
$$\text{subject to } \sum_j a_{ij} x_j \leq b_i \quad (i = 1, 2, \dots, m)$$

$$\text{where } x_j \geq 0$$

There is one and only one objective in this standard form, and the equation is linear in the parameters c_j . In standard form, we maximize the objective. But it is easy enough to set minimization as the objective by maximizing the negative of z . Many problems involve minimizing costs, which explains why we use the letter c for the fixed parameters in the objective function. Using standard form, we write less-than-or-equal-to inequality constraints on the decision variables. The form of the m constraint inequalities is linear with fixed parameters a_{ij} and b_i .

Figure A.1 provides an overview of mathematical programming modeling methods. The nature of the decision variables and the number of constraints define the type of

mathematical programming problem. When all of the decision variables are continuous, we have a linear programming problem. When all of the decision variables are integer, we say we have a pure integer programming problem. And when there is a mixture of continuous and integer decision variables, we have a mixed integer programming problem. We sometimes solve integer programming problems by pretending that the decision variables are continuous and using linear programming. This provides what is known as “relaxation” of linear programming. Most mathematical programming problems involve many constraints. A knapsack or backpack problem is a special type of integer programming problem involving only one constraint.



Adapted from. Chen, Batson, and Dang (2010).

Figure A.1. Mathematical Programming Modeling Methods

The solution to a mathematical programming problem is the set of decision variable values x_j . Some problems have a single, exact solution, satisfying the objective, and most problems have approximately optimal or very good solutions.

There are myriad applications of mathematical programming in sports. We need to pick players for teams and determine when and where to use players. Choices are subject to constraints such as salary caps, the number of players on a rosters, and the number of players in the lineup. Many problems in sports management analytics involve allocating scarce resources, maximizing revenue, or minimizing costs subject to constraints.

Mathematical programming models are deterministic, with known, fixed parameters in the objective function and constraints. But for practical purposes, it is unreasonable to assume that parameters are known and fixed. Nonetheless, here are four ways of using constrained optimization methods when we have uncertainty about cost and constraint parameters:

- **Guided Decision-Making.** Employ mathematical programming as a method for understanding the logic or nature of a decision problem and treat the solution as one of many possible answers to the problem.
- **Best-case/Worst-case Approach.** Set best-case and worst-case values for the parameters and obtain multiple solutions.
- **Sensitivity Analysis.** Let parameter values vary in a systematic way, and observe the degree to which the

solution changes.

- **Stochastic Programming.** Estimate probability distributions for the parameters and employ stochastic programming to find solutions.

For an introduction to the mathematics of linear programming, see Hadley (1962), Chvátal (1983), or Walker (2012). Stokey and Zeckhauser (1978) provide a non-mathematical introduction for decision makers. Chen, Batson, and Dang (2010) present an overview of mixed integer programming. Bradley, Hax, and Magnanti (1977) and Williams (2013) review business applications of mathematical programming. Wright (2009) reviews applications in sport. Braun and Murdoch (2007) show how to use R to solve linear programming problems. Hart, Laird, Watson, and Woodruff (2012), Sirona (2014), and Hart and Woodruff (2015) review a Python-based algebraic modeling system for mathematical programming.

A.2 CLASSICAL AND BAYESIAN STATISTICS

How shall we draw inferences from data? Formal scientific method suggests that we construct theories and test those theories with sample data. The process involves drawing statistical inferences as point estimates, interval estimates, or tests of hypotheses about the population. Whatever the form of inference, we need sample data relating to questions of interest. For valid use of statistical methods we desire a random sample from the population.

Which statistics do we trust? Statistics are functions of sample data, and we have more faith in statistics when samples are

representative of the population. Large random samples, small standard errors, and narrow confidence intervals are preferred.

Classical and Bayesian statistics represent alternative approaches to inference, alternative ways of measuring uncertainty about the world. Classical hypothesis testing involves making null hypotheses about population parameters and then rejecting or not rejecting those hypotheses based on sample data. Typical null hypotheses (as the word *null* would imply) state that there is no difference between proportions or group means, or no relationship between variables. Null hypotheses may also refer to parameters in models involving many variables.

To test a null hypothesis, we compute a special statistic called a test statistic along with its associated *p*-value. Assuming that the null hypothesis is true, we can derive the theoretical distribution of the test statistic. We obtain a *p*-value by referring the sample test statistic to this theoretical distribution. The *p*-value, itself a sample statistic, gives the probability of rejecting the null hypothesis under the assumption that it is true.

Let us assume that the conditions for valid inference have been satisfied. Then, when we observe a very low *p*-value (0.05, 0.01, or 0.001, for instance), we know that one of two things must be true: either (1) an event of very low probability has occurred under the assumption that the null hypothesis is true or (2) the null hypothesis is false. A low *p*-value leads us to reject the null hypothesis, and we say the research results are statistically significant. Some results are statistically significant and meaningful. Others are statistically significant and picayune.

For applied research in the classical tradition, we look for statistics with low p -values. We define null hypotheses as straw men with the intention of rejecting them. When looking for differences between groups, we set up a null hypothesis that there are no differences between groups. In studying relationships between variables, we create null hypotheses of independence between variables and then collect data to reject those hypotheses. When we collect sufficient data, testing procedures have statistical power.

Variability is both our enemy and our friend. It is our enemy when it arises from unexplained sources or from sampling variability—the values of statistics vary from one sample to the next. But variability is also our friend because, without variability, we would be unable to see relationships between variables.²

² To see the importance of variability in the discovery of relationships, we can begin with a scatter plot of two variables with a high correlation. Then we restrict the range of one of the variables. More often than not, the resulting scatter plot within the window of the restricted range will exhibit a lower correlation.

While the classical approach treats parameters as fixed, unknown quantities to be estimated, the Bayesian approach treats parameters as random variables. In other words, we can think of parameters as having probability distributions representing our uncertainty about the world.

The Bayesian approach takes its name from Bayes' theorem, a famous theorem in statistics. In addition to making assumptions about population distributions, random samples, and sampling distributions, we can make assumptions about population parameters. In taking a Bayesian approach, our job is first to express our degree of uncertainty about the world in the form of a probability distribution and then to reduce that

uncertainty by collecting relevant sample data.

How do we express our uncertainty about parameters? We specify prior probability distributions for those parameters. Then we use sample data and Bayes' theorem to derive posterior probability distributions for those same parameters. The Bayesian obtains conditional probability estimates from posterior distributions.

Many argue that Bayesian statistics provides a logically consistent approach to empirical research. Forget the null hypothesis and focus on the research question of interest—the scientific hypothesis. There is no need to talk about confidence intervals when we can describe uncertainty with a probability interval. There is no need to make decisions about null hypotheses when we can view all scientific and business problems from a decision-theoretic point of view (Robert 2007). A Bayesian probabilistic perspective can be applied to machine learning and traditional statistical models (Murphy 2012).

It may be a challenge to derive mathematical formulas for posterior probability distributions. Indeed, for many research problems, it is impossible to derive formulas for posterior distributions. This does not stop us from using Bayesian methods, however, because computer programs can generate or estimate posterior distributions. Markov chain Monte Carlo simulation is at the heart of Bayesian practice (Tanner 1996; Albert 2009; Robert and Casella 2009; Suess and Trumbo 2010).

Bayesian statistics is alive and well today because it helps us solve real-world problems (McGrayne 2011; Flam 2014). In the popular press, Silver (2012) makes a strong argument for

taking a Bayesian approach to predictive models. As Efron (1986) points out, however, there are good reasons why everyone is not a Bayesian.

There are many works from which to learn about classical inference (Fisher 1970; Fisher 1971; Snedecor and Cochran 1989; Hinkley, Reid, and Snell 1991; Stuart, Ord, and Arnold 2010; O'Hagan 2010; Wasserman 2010). There are also many good sources for learning about Bayesian methods (Geisser 1993; Gelman, Carlin, Stern, and Rubin 1995; Carlin and Louis 1996; Robert 2007).

When asked if the difference between two groups could have arisen by chance, we might prefer a classical approach. We estimate a p -value as a conditional probability, given a null hypothesis of no difference between the groups. But when asked to estimate the probability that the share price of Apple stock will be above \$100 at the beginning of the next calendar year, we may prefer a Bayesian approach. Which is better, classical or Bayesian? It does not matter. We need both. Which is better, Python or R? It does not matter. We need both.

A.3 REGRESSION AND CLASSIFICATION

Much of the work of data science involves a search for meaningful relationships between variables. We look for relationships between pairs of continuous variables using scatter plots and correlation coefficients. We look for relationships between categorical variables using contingency tables and the methods of categorical data analysis. We use multivariate methods and multi-way

contingency tables to examine relationships among many variables. And we build predictive models.

There are two main types of predictive models: *regression* and *classification*. Regression is prediction of a response of meaningful magnitude. Classification involves prediction of a class or category. In the language of machine learning, these are methods of supervised learning.

The most common form of regression is *least-squares regression*, also called ordinary least-squares regression, linear regression, or multiple regression. When we use ordinary least-squares regression, we estimate regression coefficients so that they minimize the sum of the squared residuals, where residuals are differences between the observed and predicted response values. For regression problems, we think of the response as taking any value along the real number line, although in practice the response may take a limited number of distinct values. The important thing for regression is that the response values have meaningful magnitude.

Poisson regression is useful for counts. The response has meaningful magnitude but takes discrete (whole number) values with a minimum value of zero. Log-linear models for frequencies, grouped frequencies, and contingency tables for cross-classified observations fall within this domain.

For models of events, duration, and survival, as in *survival analysis*, we must often accommodate censoring, in which some observations are measured precisely and others are not. With left censoring, all we know about imprecisely measured observations is that they are less than some value. With right censoring, all we know about imprecisely measured observations is that they are greater than some value.

A good example of a duration or survival model in marketing is customer lifetime estimation. We know the lifetime or tenure of a customer only after that person stops being our customer. For current customers, lifetime is imprecisely measured—it is right censored.

Most traditional modeling techniques involve *linear models* or linear equations. The response or transformed response is on the left-hand side of the linear model. The *linear predictor* is on the right-hand side. The linear predictor involves explanatory variables and is linear in its parameters. That is, it involves the addition of coefficients or the multiplication of coefficients by the explanatory variables. The coefficients we fit to linear models represent estimates of population parameters.

Generalized linear models, as their name would imply, are generalizations of the classical linear regression model. They include models for choices and counts, including logistic regression, multinomial logit models, log-linear models, ordinal logistic models, Poisson regression, and survival data models. To introduce the theory behind these important models, we begin by reviewing the classical linear regression model.

We can write the classical linear regression model in matrix notation as

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e}$$

\mathbf{y} is an $n \times 1$ vector of responses. \mathbf{X} is an $n \times p$ matrix, with p being the number of parameters being estimated. Often the first column of \mathbf{X} is a column of ones for the constant or intercept term in the model; additional columns are for

parameters associated with explanatory variables. \mathbf{b} is a $p \times 1$ vector of parameter estimates. That is to say that $\mathbf{X}\mathbf{b}$ is linear predictor in matrix notation. The error vector \mathbf{e} represents independent and identically distributed errors; it is common to assume a Gaussian or normal distribution with mean zero.

The assumptions of the classical linear regression model give rise to classical methods of statistical inference, including tests of regression parameters and analyses of variance. These methods apply equally well to observational and experimental studies. Parameters in the classical linear regression model are estimated by ordinary least squares. There are many variations on the theme, including generalized least squares and a variety of econometric models for time-series regression, panel (longitudinal) data models, and hierarchical models in general. There are also Bayesian alternatives for most classical models. For now, we focus on classical inference and the simplest error structure—*independent, identically distributed (iid)* errors.

Let y be one element from the response vector, corresponding to one observation from the sample, and let \mathbf{x} be its corresponding row from the matrix \mathbf{X} . Because the mean or expected value of the errors is zero, we observe that

$$E[y] = \mu = \mathbf{x}\mathbf{b}$$

That is, μ , the mean of the response, is equal to the linear predictor. Generalized linear models build from this fact. If we were to write in functional notation $g(\mu) = \mathbf{x}\mathbf{b}$, then, for the Gaussian distribution of classical linear regression, g is the identity function: $g(\mu) = \mu$.

Suppose $g(\mu)$ were the logit transformation. We would have the logistic regression model:

$$g(\mu) = \log\left(\frac{\mu}{(1-\mu)}\right) = \mathbf{x}\mathbf{b}$$

Knowing that the exponential function is the inverse of the natural logarithm, we solve for μ as

$$\mu = \frac{e^{\mathbf{x}\mathbf{b}}}{1 + e^{\mathbf{x}\mathbf{b}}}$$

For every observation in the sample, the expected value of the binary response is a proportion. It represents the probability that one of two events will occur. It has a value between zero and one. In generalized linear model parlance, g is called the link function. It links the mean of the response to the linear predictor.

In most of the choice studies in this book, each observation is a binary response. Customers choose to stay with their current telephone service or switch to another service. Commuters choose to drive their cars or take the train. Probability theorists think of binary responses as Bernoulli trials with the proportion or probability μ representing the mean of the response. For n observations in a sample, we have mean $n\mu$ and variance $\frac{\mu(1-\mu)}{n}$. The distribution is binomial.³

³ In many statistical treatments of this subject, π , rather than μ is used to represent the mean of the response. We use μ here to provide a consistent symbol across the class of generalized linear models.

Table A.1 provides an overview of the most important generalized linear models for work in business and economics. Classical linear regression has an identity link. Logistic regression uses the logit link. Poisson regression and log-linear models use a log link. We work Gaussian (normal), binomial, and Poisson distributions, which are in the exponential family of distributions. Generalized linear models have linear

predictors of the form $\mathbf{X}\mathbf{b}$. This is what makes them linear models—they involve functions of explanatory variables that are linear in their parameters.

<i>Distribution</i>	<i>Link Function</i>	<i>Variance Function</i>
Gaussian (normal)	μ	1
Poisson	$\log(\mu)$	μ
Binomial	$\log\left(\frac{\mu}{(1-\mu)}\right)$	$\frac{\mu(1-\mu)}{n}$

Table A.1. Three Generalized Linear Models

Generalized linear models help us model what are obvious nonlinear relationships between explanatory variables and responses. Except for the special case of the Gaussian or normal model, which has an identity link, the link function is nonlinear. Also, unlike the normal model, there is often a relationship between the mean and variance of the underlying distribution.

The binomial distribution builds on individual binary responses. Customers order or do not order, respond to a direct marketing mailing or not. Customers choose to stay with their current telephone service or switch to another service. This type of problem lends itself to logistic regression and the use of the logit link. Note that the multinomial logit model is a natural extension of logistic regression. Multinomial logit models are useful in the analysis of multinomial response variables. A customer chooses Coke, Pepsi, or RC Cola. A commuter drives, takes the train or bus, or walks to work.

When we record choices over a period of time or across a group of individuals, we get counts or frequencies. Counts arranged in multi-way contingency tables comprise the raw data of categorical data analysis. We also use the Poisson distribution and the log link for categorical data analysis and log-linear modeling. As we have seen from our discussion of the logit transformation, the log function converts a variable defined on the domain of positive real numbers into a variable defined on the range of all real numbers. This is why it works for counts.

The Poisson distribution is discrete on the domain of non-negative integers. It is used for modeling counts, as in Poisson regression. The insurance company counts the number of claims over the past year. A retailer counts the number of customers responding to a sales promotion or the number of stock units sold. A nurse counts the number of days a patient stays in the hospital. An auto dealer counts the number of days a car stays on the lot before it sells.

Linear regression is a special generalized linear model. It has normally distributed responses and an identity link relating the expected value of responses to the linear predictor. Linear regression coefficients may be estimated by ordinary least squares. For other members of the family of generalized linear models we use maximum likelihood estimation. With the classical linear model we have analysis of variance and F-tests. With generalized linear models we have analysis of deviance and likelihood ratio tests, which are asymptotic chi-square tests.

There are close connections among generalized linear models for the analysis of choices and counts. Alternative

formulations often yield comparable results. The multinomial model looks at the distribution of counts across response categories with a fixed sum of counts (the sample size). For the Poisson model, counts are random variables, being associated with individual cases, and the sum of counts is not known until observed in the sample. But we can use the Poisson distribution for the analysis of multinomial data. Log-linear models make no explicit distinction between response and explanatory variables. Instead, frequency counts act as responses in the model. But, if we focus on appropriate subsets of linear predictors, treating response variables as distinct from explanatory variables, log-linear models yield results comparable to logistic regression. The Poisson distribution and the log link are used for log-linear models.

When communicating with managers, we often use R-squared or the coefficient of determination as an index of goodness of fit. This is a quantity that is easy to explain to management as the proportion of response variance accounted for by the model. An alternative index that many statisticians prefer is the *root-mean-square error* (RMSE), which is an index of badness or lack of fit. Other indices of badness of fit, such as the percentage error in prediction, are sometimes preferred by managers.

The method of *logistic regression*, although called “regression,” is actually a classification method. It involves the prediction of a binary response. Ordinal and multinomial logit models extend logistic regression to problems involving more than two classes. Linear discriminant analysis is another classification method from the domain of traditional statistics. The benchmark study of text classification in the chapter on

sentiment analysis employed logistic regression and a number of machine learning algorithms for classification.

Evaluating classifier performance presents a challenge because many problems are low base rate problems. Fewer than five percent of customers may respond to a direct mail campaign. Disease rates, loan default, and fraud are often low base rate events. When evaluating classifiers in the context of low base rates, we must look beyond the percentage of events correctly predicted. Based on the four-fold table known as the *confusion matrix*, [figure A.2](#) provides an overview of various indices available for evaluating binary classifiers.

		Actual Binary Response		
		YES	NO	
Predicted Binary Response	YES	True Positive a	False Positive b	$a + b$
	NO	False Negative c	True Negative d	$c + d$
		$a + c$	$b + d$	$n = a + b + c + d$

Expected accuracy by chance based upon marginal totals of the confusion matrix

$$E = \left[\frac{(a+b)(a+c) + (b+d)(c+d)}{n^2} \right]$$

$$\text{True Positive Rate} = \frac{a}{a+c}$$

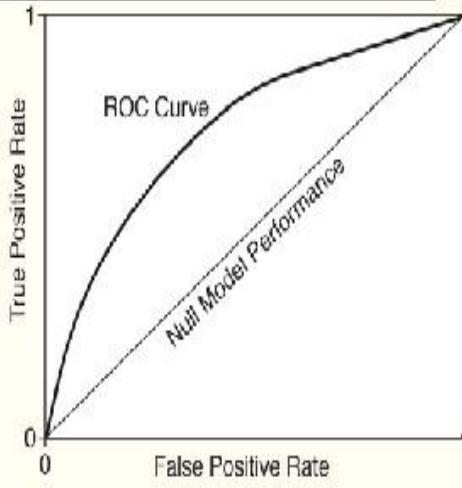
The true positive rate, also called sensitivity, shows the proportion of positives (YES responses) that are correctly identified as positive. We want this to be high.

Observed predictive accuracy = proportion correctly classified
 $O = (a+d) / (a+b+c+d)$

Precision = proportion of positive predictions that are true positives
 $= a / (a+b)$

Specificity = true negative rate
= proportion of negatives that are correctly classified as negatives
 $= d / (b+d)$
 $= 1 - (\text{false positive rate})$

Kappa = accuracy relative to accuracy expected by chance
 $= (O - E) / (1 - E)$



The false positive rate shows the proportion of negatives (NO responses) incorrectly identified as positive. We want this to be low.

The receiver operating characteristic (ROC) curve shows the performance of a binary classifier across the full range of decision criteria or cutoffs. Perfect performance would correspond to the point (0,1), that is, a false positive rate of 0 and a true positive rate of 1. The area under the curve provides a general index of classification performance.

Figure A.2. Evaluating the Predictive Accuracy of a Binary Classifier

Summary statistics such as Kappa (Cohen 1960) and the area under the receiver operating characteristic (ROC) curve are sometimes used to evaluate classifiers. Kappa depends on the probability cut-off used in classification. The area under the ROC curve does not.

The area under the ROC curve is a preferred index of classification performance for low-base-rate problems. The ROC curve is a plot of the true positive rate against the false positive rate. It shows the tradeoff between sensitivity and specificity and measures how well the model separates positive from negative cases.

The area under the ROC curve provides an index of predictive accuracy independent of the probability cut-off that is being used to classify cases. Perfect prediction corresponds to an area of 1.0 (curve that touches the top-left corner). An area of 0.5 depicts random (null-model) predictive accuracy.

Useful references for linear regression include Draper and Smith (1998), Harrell (2001), Chatterjee and Hadi (2012), and Fox and Weisberg (2011). Data-adaptive regression methods and machine learning algorithms are reviewed in Berk (2008), Izenman (2008), and Hastie, Tibshirani, and Friedman (2009). For traditional nonlinear models, see Bates and Watts (2007).

Of special concern to data scientists is the structure of the regression model. Under what conditions should we transform the response or selected explanatory variables? Should interaction effects be included in the model? Regression diagnostics are data visualizations and indices we use to check on the adequacy of regression models and to suggest variable

transformations. Discussion may be found in Belsley, Kuh, and Welsch (1980) and Cook (1998). The base R system provides many diagnostics, and Fox and Weisberg (2011) provide additional diagnostics. Diagnostics may suggest that transformations of the response or explanatory variables are needed in order to meet model assumptions or improve predictive performance. A theory of power transformations is provided in Box and Cox (1964) and reviewed by Fox and Weisberg (2011).

When defining parametric models, we would like to include the right set of explanatory variables in the right form. Having too few variables or omitting key explanatory variables can result in biased predictions. Having too many variables, on the other hand, may lead to over-fitting and high out-of-sample prediction error. This bias-variance tradeoff, as it is sometimes called, is a statistical fact of life.

Shrinkage and regularized regression methods provide mechanisms for tuning, smoothing, or adjusting model complexity (Tibshirani 1996; Hoerl and Kennard 2000). Alternatively, we can select subsets of explanatory variables to go into predictive models. Special methods are called into play when the number of parameters being estimated is large, perhaps exceeding the number of observations (Bühlmann and van de Geer 2011). For additional discussion of the bias-variance tradeoff, regularized regression, and subset selection, see Izenman (2008) and Hastie, Tibshirani, and Friedman (2009).

Graybill (1961, 2000) and Rencher and Schaalje (2008) review linear models. Generalized linear models are discussed in McCullagh and Nelder (1989) and Firth (1991). Kutner,

Nachtsheim, Neter, and Li (2004) provide a comprehensive review of linear and generalized linear models, including discussion of their application in experimental design. R methods for the estimation of linear and generalized linear models are reviewed in Chambers and Hastie (1992) and Venables and Ripley (2002).

The standard reference for generalized linear models is McCullagh and Nelder (1989). Firth (1991) provides additional review of the underlying theory. Hastie (1992) and Venables and Ripley (2002) give modeling examples with S/SPlus, most which are easily duplicated in R. Lindsey (1997) discusses a wide range of application examples. See Christensen (1997), Le (1998), and Hosmer, Lemeshow, and Sturdivant (2013) for discussion of logistic regression. Lloyd (1999) provides an overview of categorical data analysis. See Fawcett (2003) and Sing *et al.* (2005) for further discussion of the ROC curve. Discussion of alternative methods for evaluating classifiers is provided in Hand (1997) and Kuhn and Johnson (2013).

For Poisson regression and the analysis of multi-way contingency tables, useful references include Bishop, Fienberg, and Holland (1975), Cameron and Trivedi (1998), Fienberg (2007), Tang, He, and Tu (2012), and Agresti (2013). Reviews of survival data analysis have been provided by Andersen, Borgan, Gill, and Keiding (1993), Le (1997), Therneau and Grambsch (2000), Harrell (2001), Nelson (2003), Hosmer, Lemeshow, and May (2013), and Allison (2010), with programming solutions provided by Therneau (2014) and Therneau and Crowson (2014). Wassertheil-Smoller (1990) provides an elementary introduction to

classification procedures and the evaluation of binary classifiers. For a more advanced treatment, see Hand (1997). Burnham and Anderson (2002) review model selection methods, particularly those using the Akaike information criterion or AIC (Akaike 1973).

We sometimes consider robust regression methods when there are influential outliers or extreme observations. Robust methods represent an active area of research using statistical simulation tools (Fox 2002; Koller and Stahel 2011; Maronna, Martin, and Yohai 2006; Maechler 2014b; Koller 2014). Huet *et al.* (2004) and Bates and Watts (2007) review nonlinear regression, and Harrell (2001) discusses spline functions for regression problems.

A.4 DATA MINING AND MACHINE LEARNING

Recommender systems, collaborative filtering, association rules, optimization methods based on heuristics, as well as a myriad of methods for regression, classification, and clustering fall under the rubric of machine learning.

We use the term “machine learning” to refer to the methods or algorithms that we use as an alternative to traditional statistical methods. When we apply these methods in the analysis of data, we use the term “data mining.” Rajaraman and Ullman (2012) describe data mining as the “discovery of models for data.” Regarding the data themselves, we are often referring to massive data sets.

With traditional statistics, we define the model specification prior to working with the data. With traditional statistics, we often make assumptions about the population distributions

from which the data have been drawn. Machine learning, on the other hand, is data-adaptive. The model specification is defined by applying algorithms to the data. With machine learning, few assumptions are made about the underlying distributions of the data.

Machine learning methods often perform better than traditional linear or logistic regression methods, but explaining why they work is not easy. Machine learning models are sometimes called *black box models* for a reason. The underlying algorithms can yield thousands of formulas or nodal splits fit to the training data.

Extensive discussion of machine learning algorithms may be found in Duda, Hart, and Stork (2001), Izenman (2008), Hastie, Tibshirani, and Friedman (2009), Kuhn and Johnson (2013), Tan, Steinbach, and Kumar (2006), and Murphy (2012). Bacon (2002) describes their application in marketing.

Hothorn *et al.* (2005) review principles of benchmark study design, and Schauerhuber *et al.* (2008) show a benchmark study of classification methods. Alfons (2014a) provides cross-validation tools for benchmark studies. Benchmark studies, also known as statistical simulations or statistical experiments, may be conducted with programming packages designed for this type of research (Alfons 2014b; Alfons, Templ, and Filzmoser 2014).

Duda, Hart, and Stork (2001), Tan, Steinbach, and Kumar (2006), Hastie, Tibshirani, and Friedman (2009), and Rajaraman and Ullman (2012) introduce clustering from a machine learning perspective. Everitt, Landau, Leese, and Stahl (2011), Kaufman and Rousseeuw (1990) review traditional clustering methods. Izenman (2008) provides a

review of traditional clustering, self-organizing maps, fuzzy clustering, model-based clustering, and biclustering (block clustering).

Within the machine learning literature, cluster analysis is referred to as *unsupervised learning* to distinguish it from classification, which is *supervised learning*, guided by known, coded values of a response variable or class. Association rules modeling, frequent itemsets, social network analysis, link analysis, recommender systems, and many multivariate methods as we employ them in data science represent *unsupervised learning* methods.

An important multivariate method, *principal component analysis*, draws on linear algebra and gives us a way to reduce the number of measures or quantitative features we use to describe domains of interest. Long a staple of measurement experts and a prerequisite of factor analysis, principal component analysis has seen recent applications in latent semantic analysis, a technology for identifying important topics across a document corpus (Blei, Ng, and Jordan 2003; Murphy 2012; Ingersoll, Morton, and Farris 2013).

When some observations in the training set have coded responses and others do not, we employ a *semi-supervised learning* approach. The set of coded observations for the supervised component can be small relative to the set of uncoded observations for the unsupervised component (Liu 2011).

Leisch and Gruen (2014) describe programming packages for various clustering algorithms. Methods developed by Kaufman and Rousseeuw (1990) have been implemented in R programs by Maechler (2014a), including silhouette modeling and

visualization techniques for determining the number of clusters. Silhouettes were introduced by Rousseeuw (1987), with additional documentation and examples provided in Kaufman and Rousseeuw (1990) and Izenman (2008).

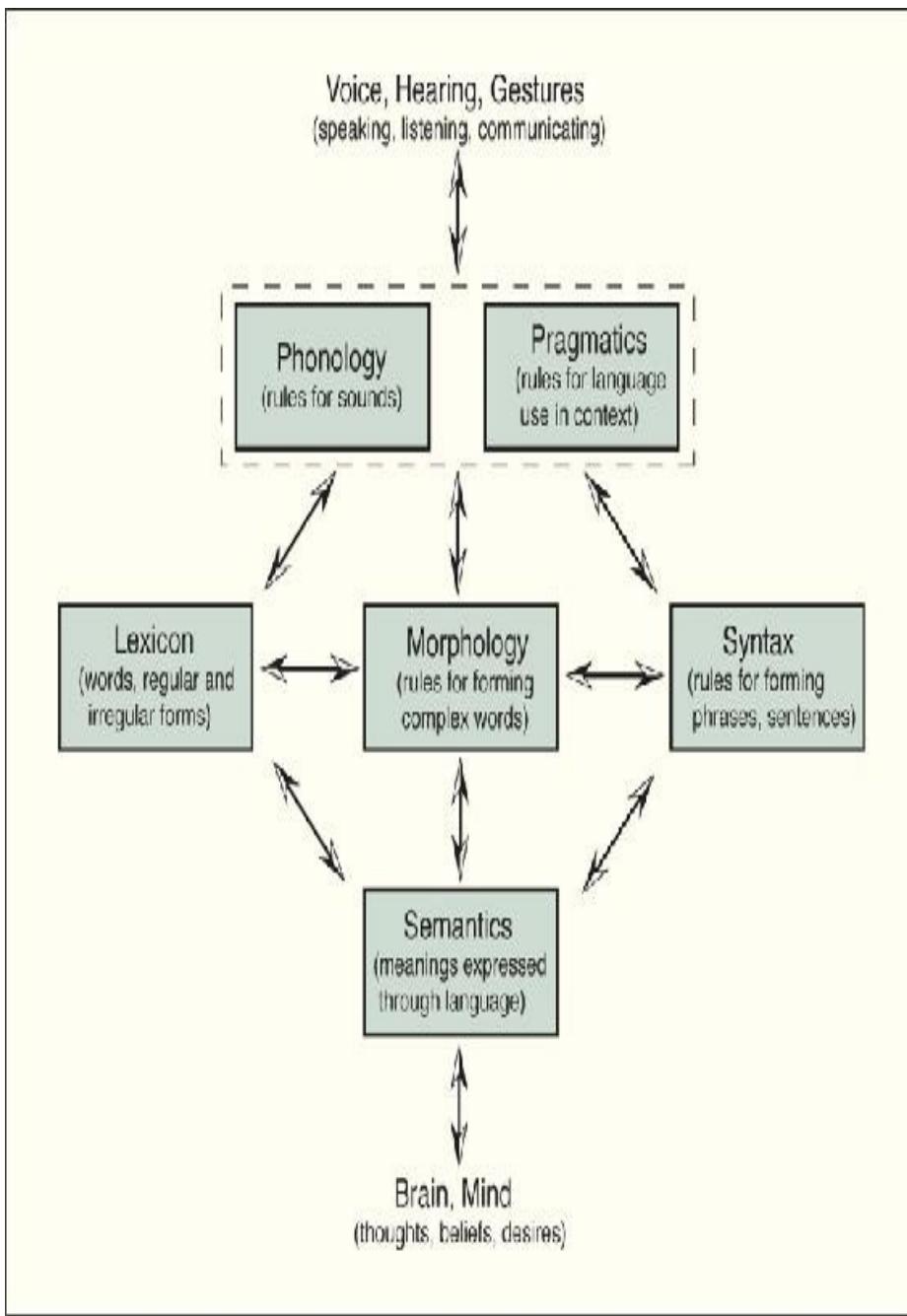
Thinking more broadly about machine learning, we see it as a subfield of artificial intelligence (Luger 2008; Russell and Norvig 2009). Machine learning encompasses biologically-inspired methods, genetic algorithms, and heuristics, which may be used to address complex optimization, scheduling, and systems design problems. (Mitchell 1996; Engelbrecht 2007; Michalawicz and Fogel 2004; Brownlee 2011).

A.5 TEXT AND SENTIMENT ANALYSIS

Text analytics draws from a variety of disciplines, including linguistics, communication and language arts, experimental psychology, political discourse analysis, journalism, computer science, and statistics. And, given the amount of text being gathered and stored by organizations, text analytics is an important and growing area of predictive analytics.

We have discussed web crawling, scraping, and parsing. The output from these processes is a document collection or text corpus. This document collection or corpus is in the natural language. The two primary ways of analyzing a text corpus are the *bag of words* approach and *natural language processing*. We parse the corpus further, creating commonly formatted expressions, indices, keys, and matrices that are more easily analyzed by computer. This additional parsing is sometimes referred to as text annotation. We extract features from the text and then use those features in subsequent analyses.

Natural language is what we speak and write every day. Natural language processing is more than a matter of collecting individual words. Natural language conveys meaning. Natural language documents contain paragraphs, paragraphs contain sentences, and sentences contain words. There are grammatical rules, with many ways to convey the same idea, along with exceptions to rules and rules about exceptions. Words used in combination and the rules of grammar comprise the linguistic foundations of text analytics as shown in figure A.3.



Source: Adapted from Pinker (1999).

Figure A.3. Linguistic Foundations of Text Analytics

Linguists study natural language, the words and the rules that we use to form meaningful utterances. “Generative grammar” is a general term for the rules; “morphology,” “syntax,” and “semantics” are more specific terms. Computer programs for

natural language processing use linguistic rules to mimic human communication and convert natural language into structured text for further analysis.

Natural language processing is a broad area of academic study itself, and an important area of computational linguistics. The location of words in sentences is a key to understanding text. Words follow a sequence, with earlier words often more important than later words, and with early sentences and paragraphs often more important than later sentences and paragraphs.

Words in the title of a document are especially important to understanding the meaning of a document. Some words occur with high frequency and help to define the meaning of a document. Other words, such as the definite article “the” and the indefinite articles “a” and “an,” as well as many prepositions and pronouns, occur with high frequency but have little to do with the meaning of a document. These *stop words* are dropped from the analysis.

The features or attributes of text are often associated with terms—collections of words that mean something special. There are collections of words relating to the same concept or word stem. The words “marketer,” “marketeer,” and “marketing” build on the common word stem “market.” There are syntactic structures to consider, such as adjectives followed by nouns and nouns followed by nouns. Most important to text analytics are sequences of words that form terms. The words “New” and “York” have special meaning when combined to form the term “New York.” The words “financial” and “analysis” have special meaning when combined to form the term “financial analysis.” We often

employ *stemming*, which is the identification of word stems, dropping suffixes (and sometimes prefixes) from words. More generally, we are parsing natural language text to arrive at structured text.

In English, it is customary to place the subject before the verb and the object after the verb. In English, verb tense is important. The sentence “Daniel carries the Apple computer,” can have the same meaning as the sentence “The Apple computer is carried by Daniel.” “Apple computer,” the object of the active verb “carry” is the subject of the passive verb “is carried.” Understanding that the two sentences mean the same thing is an important part of building intelligent text applications.

A key step in text analysis is the creation of a terms-by-documents matrix (sometimes called a lexical table). The rows of this data matrix correspond to words or word stems from the document collection, and the columns correspond to documents in the collection. The entry in each cell of a terms-by-documents matrix could be a binary indicator for the presence or absence of a term in a document, a frequency count of the number of times a term is used in a document, or a weighted frequency indicating the importance of a term in a document.

Figure A.4 illustrates the process of creating a terms-by-documents matrix. The first document comes from Steven Pinker’s *Words and Rules* (1999, p. 4), the second from Richard K. Belew’s *Finding Out About* (2000, p. 73). Terms correspond to words or word stems that appear in the documents. In this example, each matrix entry represents the number of times a term appears in a document. We treat

nouns, verbs, and adjectives similarly in the definition of stems. The stem “combine” represents both the verb “combine” and the noun “combination.” Likewise, “function” represents the verb, noun, and adjective form “functional.” An alternative system might distinguish among parts of speech, permitting more sophisticated syntactic searches across documents. After being created, the terms-by-documents matrix is like an index, a mapping of document identifiers to terms (keywords or stems) and vice versa. For information retrieval systems or search engines we might also retain information regarding the specific location of terms within documents.

Pinker (1999)

People do not just blurt out isolated words, but rather combine them into phrases and sentences, in which the meaning of the combination can be inferred from the meanings of words and the way they are arranged. We talk not merely of roses, but of the red rose, proud rose, sad rose of all my days. We can express our feelings about bread and roses, guns and roses, the War of Roses, or days of wine and roses. We can say that lovely is the rose, roses are red, or a rose is a rose is a rose. When we combine words their arrangement is crucial: *Violets are red, roses are blue*, though containing all the ingredients of the familiar verse, means something very different.

Belew (2000)

The most frequently occurring words are not really about anything. Words like NOT, OF, THE, OR, TO, BUT, and BE obviously play an important functional role, as part of the syntactic structure of sentences, but it is hard to imagine users asking for documents about OF or about BUT. Define function words to be those that have only a syntactic function, for example, OF, THE, BUT, and distinguish them from content words, which are descriptive in the sense that we're interested in them for the indexing task.

Term	Document	
	Pinker (1999)	Belew (2000)
combine	3	0
document	0	1
function	0	3
mean	3	0
rose	14	0
sentence	1	1
word	3	4

Source: Adapted from Miller (2005).

Figure A.4. Creating a Terms-by-Documents Matrix

Typical text analytics applications have many more terms than documents, resulting in sparse rectangular terms-by-documents matrices. To obtain meaningful results for text analytics applications, analysts examine the distribution of terms across the document collection. Very low frequency terms, those used in few documents, are dropped from the terms-by-documents matrix, reducing the number of rows in the matrix.

Unsupervised text analytics problems are those for which there is no response or class to be predicted. Rather, the task is to identify common patterns or trends in the data. As part of the task, we may define text measures describing the documents in the corpus.

A fundamental unsupervised application of text analytics is information retrieval. Text documents are indexed, making them searchable. A popular search algorithm depends on term frequency-inverse document frequency (TF-IDF) measurement. We note the frequency of terms in each document, relative to the frequency of those terms across the entire document collection. A vector of TF-IDF values represents each document. A user searching for information enters a text query, which is itself a document represented by a TF-IDF vector. Each document in the collection is matched to the query, obtaining a relevance score. The result is a ranked list of search results.

For supervised text analytics problems, there is a response or class of documents to be predicted. We build a model on a training set and test it on a test set. Text classification problems are common. Spam filtering has long been a subject of interest as a classification problem, and many e-mail users have benefitted from the efficient algorithms that have evolved in this area. In the context of information retrieval, search engines classify documents as being relevant to the search or not. Useful modeling techniques for text classification include logistic regression, linear discriminant function analysis, classification trees, and support vector machines. Various ensemble or committee methods may be employed.

Automatic text summarization is an area of research and

development that can help with information management. Imagine a text processing program with the ability to read each document in a collection and summarize it in a sentence or two, perhaps quoting from the document itself. Today's search engines are providing partial analysis of documents prior to their being displayed. They create automated summaries for fast information retrieval. They recognize common text strings associated with user requests. These applications of text analysis comprise tools of information search that we take for granted as part of our daily lives.

Programs with syntactic processing capabilities, such as IBM's Watson, provide a glimpse of what intelligent agents for text analytics are becoming. These programs perform grammatical parsing with an understanding of the roles of subject, verb, object, and modifier. They know parts of speech (nouns, verbs, adjective, adverbs). And, using identified entities representing people, places, things, and organizations, they perform relationship searches.

Sentiment analysis is measurement-focused text analysis. Sometimes called opinion mining, one approach to sentiment analysis is to draw on positive and negative word sets (lexicons, dictionaries) that convey human emotion or feeling. These word sets are specific to the language being spoken and the context of application. Another approach to sentiment analysis is to work directly with text samples and human ratings of those samples, developing text scoring methods specific to the task at hand.

A semi-supervised machine learning regimen can be especially useful in sentiment analysis. We work two sets of text samples. One sample, often a small sample (because it is

expensive and time-consuming to obtain human ratings of text), associates a rating or score with each text document. Another much larger sample is unrated but comes from the same content domain. We learn the direction of scoring from the first sample, and we learn about the text domain (including term frequencies in context) from both samples.

The objective of sentiment analysis is to score text for affect, feelings, attitudes, or opinions. Sentiment analysis and text measurement in general hold promise as technologies for understanding consumer opinion and markets. Just as political researchers can learn from the words of the public, press, and politicians, business researchers can learn from the words of customers and competitors. There are customer service logs, telephone transcripts, and sales call reports, along with user group, listserv, and blog postings. And we have ubiquitous social media from which to build document collections for text and sentiment analysis.

Precursors to sentiment analysis may be found in content analysis, thematic, semantic, and network text analysis (Roberts 1997; Popping 2000; West 2001; Leetaru 2011; Krippendorff 2012). These methods have seen a wide range of applications within the social sciences, including analysis of political discourse. An early computer implementation of content analysis is found in the General Inquirer program (Stone *et al.* 1966; Stone 1997). Buvač and Stone (2001) describe a version of the program that provides text measures based on word counts across numerous semantic categories.

Text measures flow from a measurement model (algorithms for scoring) and a dictionary, both defined by the researcher or analyst. A dictionary in this context is not a traditional

dictionary; it is not an alphabetized list of words and their definitions. Rather, the dictionary used to construct text measures is a repository of word lists, such as synonyms and antonyms, positive and negative words, strong and weak sounding words, bipolar adjectives, parts of speech, and so on. The lists come from expert judgments about the meaning of words. A text measure assigns numbers to documents according to rules, with the rules being defined by the word lists, scoring algorithms, and modeling techniques in predictive analytics.

Sentiment analysis and text measurement in general hold promise as technologies for understanding consumer opinion and markets. Just as political researchers can learn from the words of politicians, the press, and the public, data scientists can learn from the words of customers and competitors. For the data scientist interested in understanding consumer opinions about brands and products, there are substantial sources from which to draw samples. We have customer service logs, telephone transcripts, and sales call reports, along with user group, listserv, and blog postings. And we have ubiquitous social media from which to build document collections for text and sentiment analysis.

The measurement story behind opinion and sentiment analysis is an important story that needs to be told. Sentiment analysis, like all measurement, is the assignment of numbers to attributes according to rules. But what do the numbers mean? To what extent are text measures reliable or valid? Face validity (or content validity) involves showing that the content of the text measure relates to the attribute being measured. We examine word sets, and we try to gain agreement (among

subject matter experts, perhaps) that they measure a particular attribute or trait. Sentiment research often involves testing word sets within specific contexts and, when possible, testing against external criteria. To demonstrate predictive validity, we show that a text measure can be used for prediction.

Numerous studies have demonstrated the utility of Twitter-based measures in predictive models. Some researchers have predicted the success of movies prior to their being distributed to theaters nationwide (Sharda and Delen 2006; Delen, Sharda, and Kumar 2007). Most telling is work completed at HP Labs that used Twitter chat as a predictor of movie revenues (Asur and Huberman 2010). Bollen, Mao, and Zeng (2011) predicted stock market movements from Twitter sentiment measures.

Taddy's (2013b, 2014) sentiment analysis work built on the inverse regression methods of Cook (1998, 2007). Taddy (2013a) also used Twitter data to examine political sentiment.

Some have voiced concerns about unidimensional measures of sentiment. There have been attempts to develop more extensive sentiment word sets, as well as multidimensional measures (Turney 2002; Asur and Huberman 2010). Recent developments in machine learning and quantitative linguistics point to sentiment measurement methods that employ natural language processing rather than relying on positive and negative word sets (Socher et al. 2011). Among the more popular measurement schemes from the psychometric literature is Charles Osgood's semantic differential (Osgood, Suci, and Tannenbaum 1957; Osgood 1962). Exemplary bipolar dimensions include the positive–negative, strong–weak, and active–passive dimensions. Schemes like Osgood's set the stage for multidimensional measures of sentiment. We

expect sentiment analysis to be an active area of research for many years. Reviews of sentiment analysis methods have been provided by Liu (2010, 2011, 2012) and Feldman (2013).

Other books in the *Modeling Techniques* series provide examples of sentiment analysis using document collections of movie reviews Miller (2015c, 2015b, 2015d).

Ingersoll, Morton, and Farris (2013) give an introduction to the domain of text analytics for the working data scientist. Those interested in reading further can refer to Feldman and Sanger (2007), Jurafsky and Martin (2009), Weiss, Indurkhya, and Zhang (2010), and the edited volume by Srivastava and Sahami (2009).

Salton, Wong, and Yang (1975), Belew (2000), Meadow, Boyce, and Kraft (2000), and Baeza-Yates and Ribeiro-Neto (2011) review document indexing and search theory. Open-source search solutions include Elasticsearch and Solr, which build on Apache Lucene. Gromley and Tong (2015) introduce the Elasticsearch implementation of indexing and search.

Hausser (2001) gives an account of generative grammar and computational linguistics. Statistical language learning and natural language processing are discussed by Charniak (1993), Manning and Schütze (1999), and Indurkhya and Damerau (2010). Many of Steven Pinker's works (1994, 1997, 1999) provide insight into grammar and psycholinguistics. Maybury (1997) reviews data preparation for text analytics and the related tasks of source detection, translation and conversion, information extraction, and information exploitation.

Authorship identification, a problem addressed a number of years ago in the statistical literature by Mosteller and Wallace (1984), continues to be an active area of research (Joula 2008).

Merkel (2002) provides discussion of clustering techniques, which explore similarities between documents and the grouping of documents into classes. Dumais (2004) reviews latent semantic analysis and statistical approaches to extracting relationships among terms in a document collection.

A.6 TIME SERIES, SALES FORECASTING, AND MARKET RESPONSE MODELS

Sales forecasts are a critical component of business planning and a first step in the budgeting process. Models and methods that provide accurate forecasts can greatly benefit management. They help managers understand what determines sales, including promotions, pricing, advertising, and distribution. They reveal competitive position and market share.

There are many approaches to forecasting. Some rely on expert opinions or consensus. There are top-down and bottom-up forecasts, and various techniques for combining expert opinions. Other approaches depend on the analysis of past sales data.

A sales forecasting model can and should be organized by time periods useful to management. These may be days, weeks, months, or whatever intervals make sense for the problem at hand. Time dependencies can be noted in the same manner as in traditional time-series models. Autoregressive terms are useful in many contexts. Time-construed covariates, such as day of the week or month of the year, may be added to provide additional predictive power. And we may include promotion, pricing, and advertising variables organized in

time.

An analyst can work with time series data, using past sales to predict future sales, noting overall trends and cyclical patterns in the data. Exponential smoothing, moving averages, and various regression and econometric methods may be used with time series data.

Forecasting by location provides detail needed for management action. And organizing data by location contributes to a model's predictive power. Location may itself be used as a factor in models. In addition, we can search for explanatory variables tied to location. With geographic regions, for example, we might include consumer and business demographic variables known to relate to sales.

Sales dollars per time period is the typical response variable of interest in sales forecasting studies. Alternative response variables include sales volume and time-to-sale. Related studies of market share require information about other firms' sales in the same product category.

Forecasting is a large application area deserving its own professional conferences and journals. An overview of business forecasting methods is provided by Armstrong (2001). Time-series, financial, and econometric modeling methods are especially relevant to this application area (Judge *et al.* 1985; Hamilton 1994; Zivot and Wang 2003).

Longitudinal data analysis or panel data analysis is an example of a hierarchical, multi-level, or mixed data method with data organized by cross-sectional units and time. Frees and Miller (2004) describe a sales forecasting method that utilizes a mixed data modeling approach, working with sales data organized by time and location.

Autoregressive integrated moving average (ARIMA) models or what are often called Box-Jenkins models (Box, Jenkins, and Reinsel 2008) represent an important class of time series models. These have been used extensively in sales forecasting and in the analysis of financial and economic time series.

When working with time series models for sales, we try to select the very best model in terms of the Akaike Information Criterion (AIC) or some other measure that combines goodness-of-fit and parsimony. We use the selected model to generate forecasts and error bands around those forecasts. Hyndman *et al.* (2014) provide programs to search across large sets of candidate models, including autoregressive, moving-average, and seasonal components.

Forecasting uncertainty is estimated around the forecasted values. There is always uncertainty about the future, and the further we look into the future, the greater our uncertainty. The value of a model lies in the quality of its predictions, and sales forecasting presents challenging problems for data scientists.

When working with multiple time series, we might fit a multivariate time series or vector autoregressive (VAR) model to data. Alternatively, we could explore dynamic linear models, regressing one time series on another. We can utilize ARIMA transfer function models or state space models with regression components. The possibilities are as many as the modeling issues to be addressed.

There is a subtle but important distinction to be made between time series models and time series regression models. When we use the term *time series regression*, we are referring to regression analysis in which the organizing unit of analysis is time. We look at relationships among economic measures

organized in time. Much economic analysis concerns time series regression. Special care must be taken to avoid what might be called spurious relationships, as many economic time series are correlated with one another because they depend on common underlying factors, such as population growth or seasonality.

In time series regression, we check the residuals from our regression model to ensure that they are not correlated in time. If they are correlated in time (autocorrelated), then we use a method such as generalized least squares as an alternative to ordinary least squares. That is, we incorporate an error data model as part of our modeling process.

When we use the term *time series analysis*, we are not talking about time series regression. We are talking about methods that start by focusing on one economic measure at a time and its pattern across time. We look for trends, seasonality, and cycles in that individual time series. Then, after working with that single time series, we look at possible relationships with other time series. If we are concerned with forecasting or predicting the future, as we often are in data science, then we use time series analysis. Recently, there has been considerable interest in state space models for time series, which provide a convenient mechanism for incorporating regression components into dynamic time series models (Commandeur and Koopman 2007; Hyndman, Koehler, Ord, and Snyder 2008; Durbin and Koopman 2012).

There are myriad applications of time series analysis in sports business marketing, including marketing mix models and advertising research models. Along with sales forecasting, these fall under the general class of market response models,

as reviewed by Hanssens, Parsons, and Schultz (2001).

Marketing mix models look at the effects of price, promotion, and product placement in retail establishments. These are multiple time series problems.

Advertising research looks for cumulative effectiveness of advertising on brand and product awareness, as well as on sales. Exemplary reviews of advertising research methods and findings have been provided by Berndt (1991) and Lodish *et al.* (1995). Much of this research employs measures such as “advertising stock,” which is an accumulation of advertising impressions or rating points across a moving time series window of months or years. The thinking is that messages are most influential immediately after being received, decline in influence with time, but do not decline completely until many units in time later. Viewers or listeners remember advertisements long after initial exposure. Due to carry-over effects from one time period to the next, assessing advertising effectiveness presents a special challenge for data scientists.

Similar to other data with which we work, sales and marketing data are organized by observational unit, time, and space. The observational unit is typically an economic agent (individual or firm) or a group of such agents as in an aggregate analysis. It is common to use geographical areas as a basis for aggregation. Alternatively, space (longitude and latitude) can be used directly in spatial data analyses. Time considerations are especially important in macroeconomic analysis, which focuses on nationwide economic measures.

Baumohl (2008) provides a review of economic measures that are commonly thought of as leading indicators. Kennedy (2008) provides an introduction to the terminology of

econometrics. Key references in the area of econometrics include Judge et al.(1985), Berndt (1991), Enders (2010), and Greene (2012). Reviews of time series modeling and forecasting methods are provided by Holden, Peel, and Thompson (1990) and in the edited volume by Armstrong (2001).

More detailed discussion of time series methods is provided by Hamilton (1994), Makridakis, Wheelwright, and Hyndman (2005), Box, Jenkins, and Reinsel (2008), Hyndman *et al.* (2008), Durbin and Koopman (2012), and Hyndman and Athanasopoulos (2014). Time-series, panel (longitudinal) data, financial, and econometric modeling methods are especially relevant in demand and sales forecasting. Hierarchical and grouped time series methods are reviewed by Athanasopoulos, Ahmed, and Hyndman (2009) and Hyndman *et al.* (2011).

For gathering economic data with R, see Ryan (2014). Useful for programming with dates are R functions provided by Grolemund and Wickham (2011, 2014). Econometric and time series programs are listed in Kleiber and Zeileis (2008), Hothorn *et al.* (2014), Cowpertwait and Metcalfe (2009), Petris, Petrone, and Campagnoli (2009), Tsay (2013), Hyndman *et al.*(2014), Petris (2010), Petris and Gilks (2014), and Szymanski (2014).

A.7 SOCIAL NETWORK ANALYSIS

Teams, apart from competing on the fields and courts, interact as businesses and cooperate in leagues. There are player trades between teams and many communications among teams. Professional sports present a potentially rich domain for social network research, thinking of teams as

economic agents. Another way of using social network analysis in sports would be to consider patterns of interaction among players and coaches of a team.

A social network consists of nodes and links (vertices and edges in the language of graph theory). For example, we could think of each team as a node in a social network and each player trade as a link in that network. Then we could use social network analysis to understand better a log of trades across one or a number of seasons. To date, social network analysis is a relatively unexplored area of sports analytics.

Social network analysis has been an active area of research in psychology, sociology, anthropology, and political science for many years. The invention of the sociogram and concepts of social structure may be traced back more than eighty years (Moreno 1934; Radcliffe-Brown 1940). Research topics include isolation and popularity, prestige, power, and influence, social cohesion, subgroups and cliques, status and roles within organizations, balance and reciprocity, marketplace relationships, and measures of centrality and connectedness.

In years past, social network data collection was a painstaking process. Not so today. Social media implemented through the web have rekindled interest in social network analysis.

Technologies of the web provide a record of what people do, where they come from, where they go, with whom they communicate, and sometimes transcripts of what they say. There are online friends and followers, tweets and retweets, text messages, e-mail, and blog postings. The data are plentiful. The challenge is to find our way to useful data and make sense of them.

Recognizing growth in the use of electronic social networks and intelligent mobile devices, organizations see opportunities for communicating with and selling to friends of friends.

Businesses, nonprofits, and governmental organizations (not to mention political campaigns) are interested in learning from the data of social media. Network data sets are large but accessible. The possibilities are many.

Social networks are useful for making predictions. We can expect that our attitudes and behavior are affected by the people we know. People closest to us in “network space,” as it were, may be most influential in affecting our attitudes and behavior.

Various network measures come from mathematicians working with graph theory and from sociologists working with social networks. Network measures describe relationships among nodes and structural characteristics of networks.

We use mathematical models of networks in making predictions about network phenomena and in studying relationships across network measures. Three types of models are of special interest: random graph, preferential attachment, and small-world network models.

The first mathematical model of interest, developed by Paul Erdős and Alfred Rényi (1959, 1960, 1961), laid the foundation for models to follow. A *random graph* is a set of nodes connected by links in a purely random fashion. The *small-world network* is especially important in representing social networks (Watts and Strogatz 1998). It defines a structure in which many nodes or actors are connected to nearby neighbors and some nodes are linked to nodes that are not nearby. Small-world networks have been studied

extensively in social psychology, sociology, and network science (Milgram 1967; Travers and Milgram 1969; Watts 1999; Schnettler 2009).

Measures of node importance are useful for finding sources of power, influence, or importance. We might want to characterize the importance of a politician to his party, thinking of measures of centrality as indicators of political power.

We can begin by counting the links connected to the node. And for a directed network, we can count the number of links in and out—we measure the *degree* of each node, both *in-degree* and *out-degree*. The *degree distribution* of a network is one way to characterize the structure of a network, and the average degree or *degree centrality* is a way to summarize the connectedness of an entire network.

While the degree centrality of a node may be computed by looking only at that single node, other measures of centrality depend on the structure of the entire network. One nodal measure is *closeness centrality*, which characterizes how close a node is to all other nodes in the network, where closeness is the number of hops or links between nodes. Computing closeness centrality can be problematic in disconnected networks, so it is used less often to measure centrality than other measures (Borgatti, Everett, and Johnson 2013). When measuring *betweenness centrality*, we consider the proportion of times a node lies on the shortest path between other pairs of nodes. Betweenness centrality reflects the degree to which a node is a broker or go-between node, affecting the traffic or flow of information across the network.

What if we consider a node as important if it is close to other

nodes of importance, which are in turn close to other nodes of importance, and so on. This thinking moves us in the direction of the notion of *eigenvector centrality*. Much as the first principle component characterizes common variability in a set of variables, eigenvector centrality characterizes the degree to which a node is central to the set of nodes comprising the network. Eigenvector centrality is a measure of overall connectedness or importance. A node acquires higher eigenvector centrality by being connected to other well-connected nodes. Eigenvector centrality depends on the entire network and is a good summary measure of node importance.

There are power brokers within professional sports, as documented for Major League Baseball by Pessah (2015). If data scientists working for an MLB team were to trace trades among teams from the beginning of December 2014 through the end of July 2015, for example, they would learn much about the social network of teams or, more specifically, the social network of team general managers. And what data scientists learn about the social network of teams could inform future team behavior within that network. Information provides strategic advantage.

Major League Baseball posts a complete list of team transactions online, showing player injuries, transfers, assignments, and trades. Transactions from one day in December 2014 are shown in table A.2. Trades are important to sports teams, and much can be learned about teams by reviewing trades among teams.

Kansas City Royals designated 2B Johnny Giavotella for assignment.

Oakland Athletics designated Fernando Rodriguez for assignment.

Cleveland Indians sent RHP Bryan Price outright to Columbus Clippers.

Los Angeles Dodgers traded RF Matt Kemp, C Tim Federowicz and cash to San Diego Padres for C Yasmani Grandal, RHP Joe Wieland and RHP Zach Eflin.

Los Angeles Angels traded CF Matt Long to Los Angeles Dodgers.

Milwaukee Brewers traded C Shawn Zarraga to Los Angeles Dodgers for CF Matt Long and LHP Jarret Martin.

Oakland Athletics traded C Derek Norris and RHP Seth Strehl to San Diego Padres for RHP Jesse Hahn and RHP R.J. Alvarez.

Arizona Diamondbacks traded LHP Eury De La Rosa to Oakland Athletics for cash.

Toronto Blue Jays signed free agent RHP Rafael Cova to a minor league contract.

Texas Rangers signed free agent 1B Mike McDade to a minor league contract.

Chicago Cubs signed free agent RHP Jesus Camargo to a minor league contract.

Detroit Tigers signed free agent 2B Brandon Douglas to a minor league contract.
Kansas City Royals signed free agent RHP Kris Medlen.
Pittsburgh Pirates signed free agent RHP Adam Miller to a minor league contract.
Minnesota Twins signed free agent 1B Brock Peterson to a minor league contract.
Seattle Mariners signed free agent RHP Mark Lowe to a minor league contract and invited him to spring training.
Colorado Rockies signed free agent LF Roger Bernadina to a minor league contract and invited him to spring training.
Cleveland Indians signed free agent RHP Jeff Manship to a minor league contract and invited him to spring training.
Detroit Tigers signed free agent C Miguel Gonzalez to a minor league contract and invited him to spring training.
Chicago Cubs signed free agent 2B Carlos Sepulveda to a minor league contract.

Source. Major League Baseball (2015b).

Table A.2. Social Network Data: MLB Player Transactions, December 18, 2014

A.8 DATA VISUALIZATION

Data visualization is critical to the work of data science.

Examples in this book demonstrate the importance of data visualization in discovery, diagnostics, and design. We employ tools of exploratory data analysis (discovery) and statistical modeling (diagnostics). In communicating results to management, we use presentation graphics (design).

Statistical summaries fail to tell the story of data. To understand data, we must look beyond data tables, regression coefficients, and the results of statistical tests. Visualization tools help us learn from data. We explore data, discover patterns in data, identify groups of observations that go

together and unusual observations or outliers. We note relationships among variables, sometimes detecting underlying dimensions in the data.

There is no more telling example of the importance of data visualization than a demonstration that is affectionately known as the Anscombe Quartet. Consider the data sets in [figure A.5](#), developed by [Anscombe \(1973\)](#). Looking at the tabulated data, the casual reader will note that the fourth data set is clearly different from the others. But what about the first three data sets? Are there obvious differences in patterns of relationship between x and y ?

Set I		Set II		Set III		Set IV	
x1	y1	x2	y2	x3	y3	x4	y4
10	8.04	10	9.14	10	7.46	8	6.58
8	6.95	8	8.14	8	6.77	8	5.76
13	7.58	13	8.74	13	12.74	8	7.71
9	8.81	9	8.77	9	7.11	8	8.84
11	8.33	11	9.26	11	7.81	8	8.47
14	9.96	14	8.10	14	8.84	8	7.04
6	7.24	6	6.13	6	6.08	8	5.25
4	4.26	4	3.10	4	5.39	19	12.50
12	10.84	12	9.13	12	8.15	8	5.56
7	4.82	7	7.26	7	6.42	8	7.91
5	5.68	5	4.74	5	5.73	8	6.89

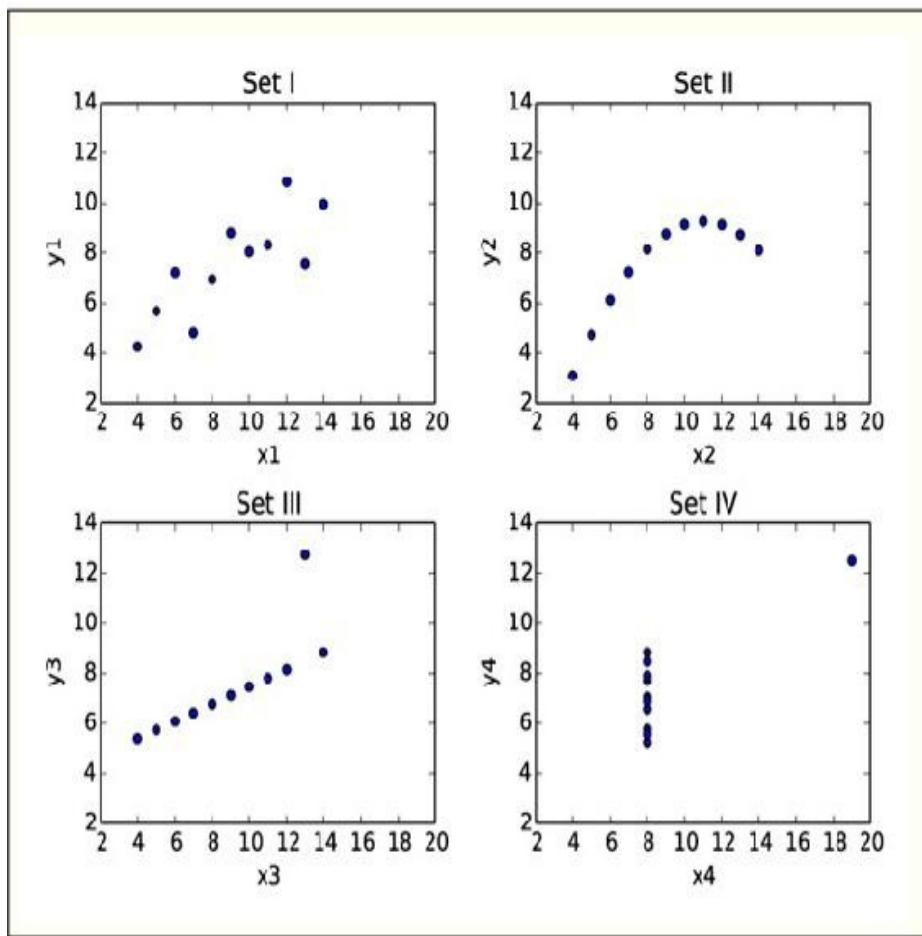


Figure A.5. Data and Plots for the Anscombe Quartet

When we regress y on x for the data sets, we see that the models provide similar statistical summaries. The mean of the

response y is 7.5, the mean of the explanatory variable x is 9. The regression analyses for the four data sets are virtually identical. The fitted regression equation for each of the four sets is $\hat{y} = 3 + 0.5x$. The proportion of response variance accounted for is 0.67 for each of the four models. Python and R programs for the Anscombe Quartet are provided in exhibits A.1 and A.2 (pages 242 and 244, respectively).

We see many examples of the utility of visualization in sports analytics and data science. Events in games are revealed by play-by-play and spatial data plots. We can view team performance across entire seasons. On the sports business side, we can see relationships among products and services and explore consumer preferences in perceptual maps. We can explore relationships among continuous performance measures in scatter plots and relationships among categorical variables in mosaic plots.

Displays of events across an entire season or a portion of a season are demonstrated by differential runs plots for baseball, introduced by Miller (2008). To make a differential runs plot, we work with one team at a time. The plot shows positive or negative differential runs on the vertical axis and is indexed on the horizontal axis by games. It is often convenient to show the abbreviation for the opposing team along the horizontal axis. Indices of competitive power may be computed from these data. With solid or open circles to show home versus away games, a differential runs plot provides a picture of a team's performance across many games.

Figure A.6 shows the differential runs plot for the Arizona Diamondbacks from April 1 through August 31, 2007. The plot shows a team with a good win/loss record but low

competitive power—they win many one-run games but lose many other games by five runs or more. Streaks of winning games, clearly visible in this Diamondbacks’ differential runs plot, are seen as consecutive vertical lines above the zero-line. Exhibit A.3 (page 245) shows R code for generating differential runs plots.

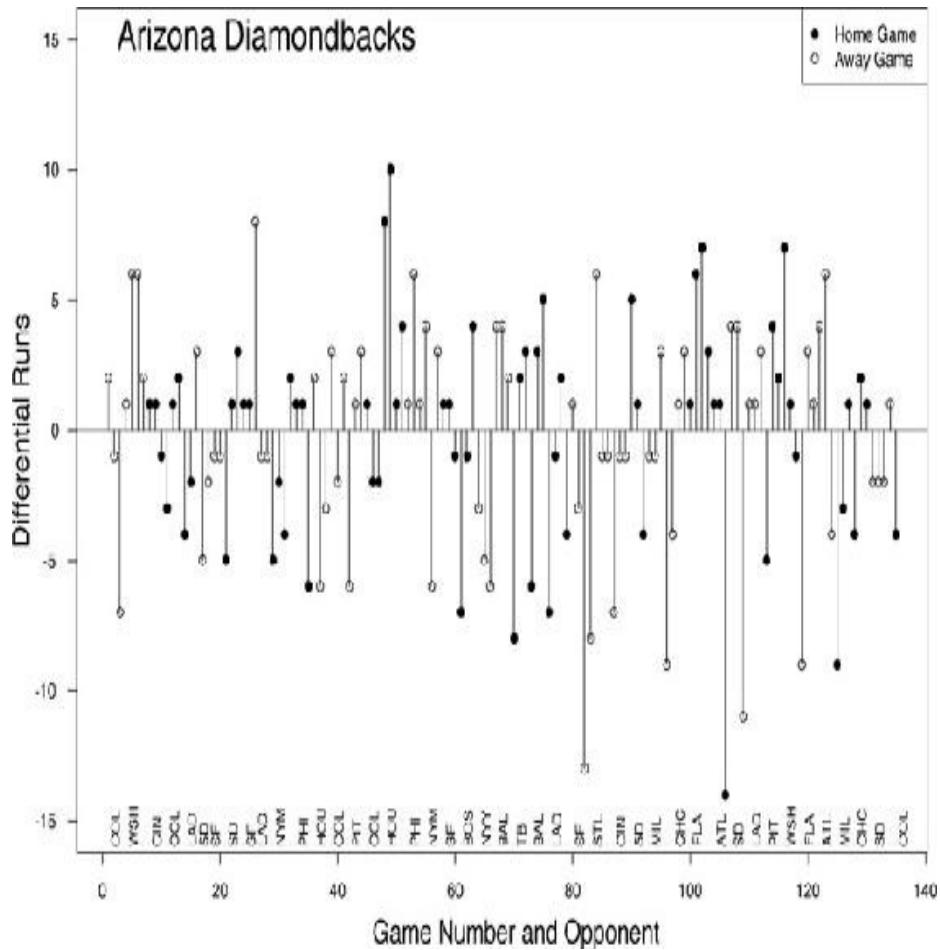
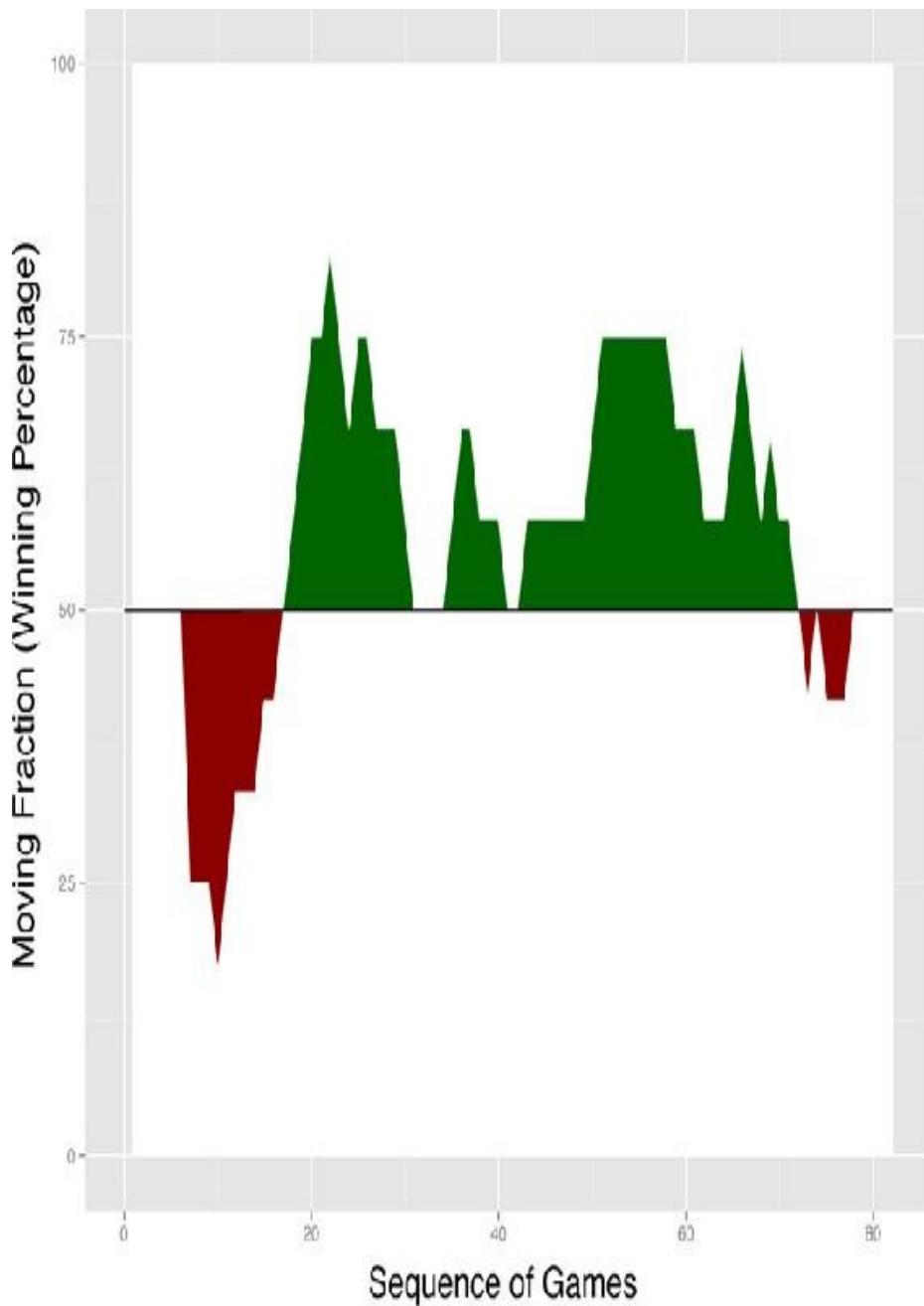


Figure A.6. Visualizing Many Games Across a Season:
Differential Runs Plot

Displays of winning and losing streaks across a season are illustrated by the moving fraction plot, introduced by Albert and Bennett (2001). To make a moving fraction plot, we organize box score data in temporal order, noting game outcomes with a binary variable. We then employ a moving

window across the game series, computing the proportion or percentage of wins within the widow. A ribbon plot may be used to color-code winning streaks above the 50-percent line and losing streaks below the 50-percent line.

Figure A.7 shows the moving fraction plot for the Oklahoma City Thunder of the NBA. The plot shows streaks of winning and losing across the 2014– 2015 season. Exhibit A.4 (page 246) shows an R program for making this moving fraction plot. This program draws on graphics software developed by Wickham and Chang (2014) and documented by Chang (2013).

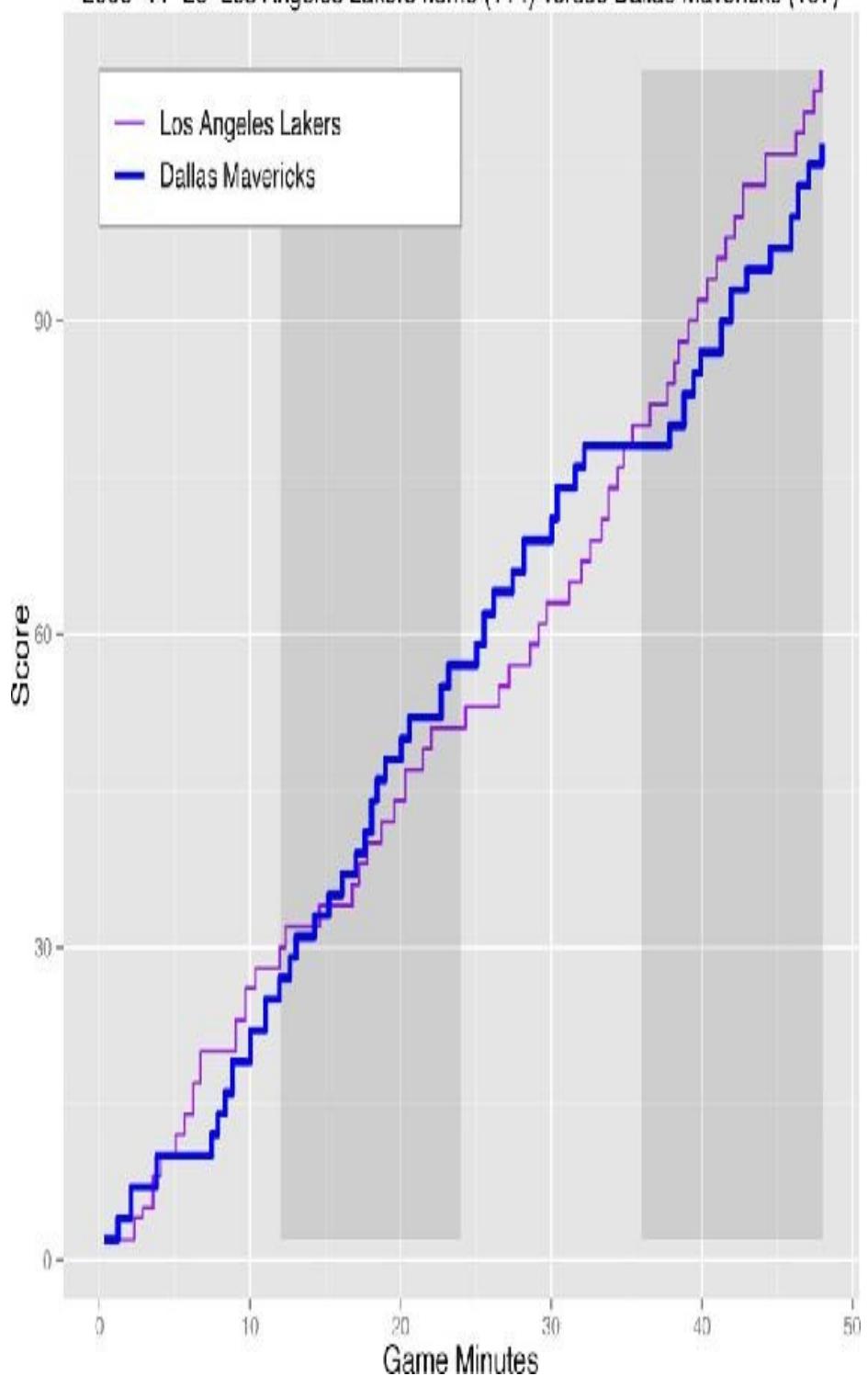


**Figure A.7. Moving Fraction Plot for Basketball:
Oklahoma City Thunder (2014–2015 Season)**

Data visualization of play-by-play data reveals the effects of team scoring streaks. Long no-scoring streaks go hand-in-hand with lead changes, as we can see in figure A.8. In the Lakers-Mavericks game, we see no-scoring streaks by both teams

associated with lead changes across the game. In the Lakers-Thunder game, we see one no-scoring streak in the second period, setting the stage for the Lakers taking and keeping the lead.

2008-11-28 Los Angeles Lakers home (114) versus Dallas Mavericks (107)



2009-02-10 Los Angeles Lakers home (105) versus Oklahoma City Thunder (98)

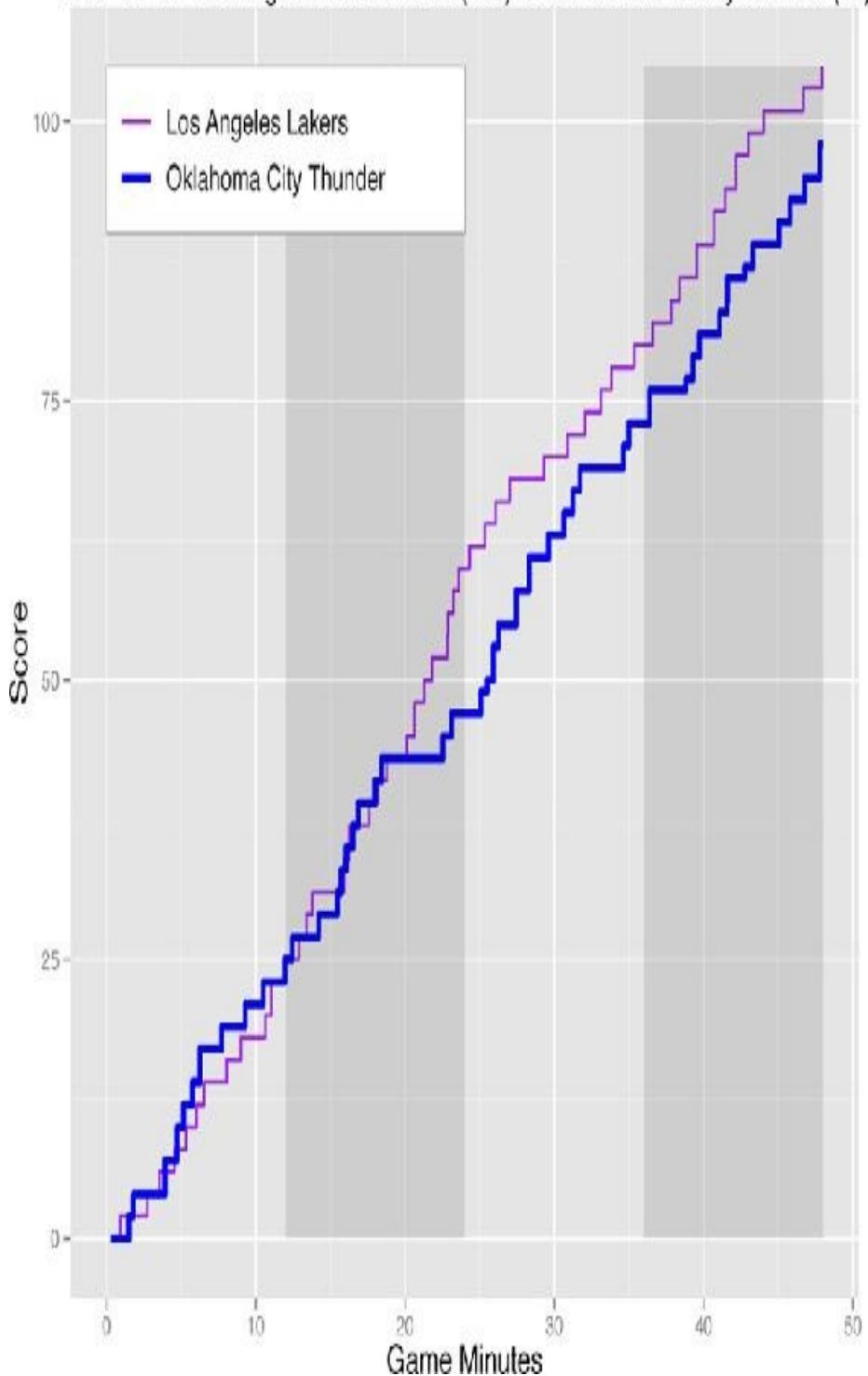


Figure A.8. Visualizing Basketball Play-by-Play Data

Exhibit A.5 (page 248) shows an R program that uses basketball play-by-play data to create game visualizations. It

generates the complete set of cumulative frequency distributions for the Los Angeles Lakers’ 2008–2009 season. The program utilizes play-by-play data provided by Parker (2010) and code developed by Grolemund and Wickham (2011, 2014), Wickham and Chang (2014), and Wickham (2014).

Heer, Bostock, and Ogievetsky (2010) demonstrate contemporary visualization techniques for web distribution. When working with very large data sets, special methods may be needed, such as partial transparency and hexbin plots (Unwin, Theus, and Hofmann 2006; Carr, Lewin-Koh, and Maechler 2014; Lewin-Koh 2014).

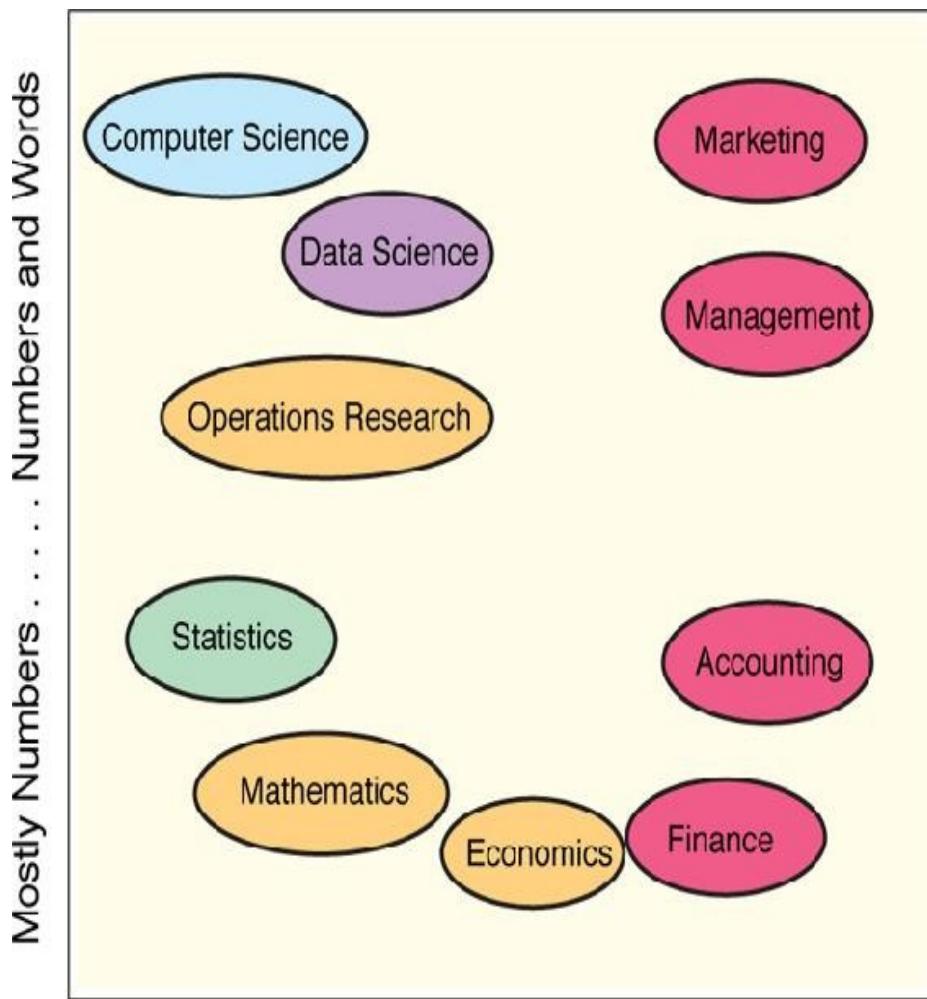
R is particularly strong for data visualization. An R graphics overview is provided by Murrell (2011). R lattice graphics, discussed by Sarkar (2008, 2014), build on the conceptual structure of an earlier system called S-Plus TrellisTM (Cleveland 1993; Becker and Cleveland 1996). Wilkinson’s (2005) “grammar of graphics” approach has been implemented in the Python ggplot package (Lamp 2014) and in the R ggplot2 package (Wickham and Chang 2014), with R programming examples provided by Chang (2013). Cairo (2013) and Zeileis, Hornik, and Murrell (2009, 2014) provide advice about colors for statistical graphics. Ihaka *et al.* (2014) show how to specify colors in R by hue, chroma, and luminance.

Graphics for exploratory data analysis are reviewed in classic references by Tukey (1977) and Tukey and Mosteller (1977). Regression graphics are covered by Cook (1998), Cook and Weisberg (1999), and Fox and Weisberg (2011). Statistical graphics and data visualization are illustrated in the works of

Tufte (1990, 1997, 2004, 2006), Few (2009), and Yau (2011, 2013). Wilkinson (2005) presents a review of human perception and graphics, as well as a conceptual structure for understanding statistical graphics. Cairo (2013) provides a general review of information graphics.

A.9 DATA SCIENCE: THE ECLECTIC DISCIPLINE

Data science draws from many academic disciplines, as we have seen in this book. To show how data science relates to other disciplines, I constructed a similarity ranking task for myself. The task comprised forty-five items—all possible pairs of ten discipline names. [Figure A.9](#) shows my personal perceptual map of these disciplines.



Methods Applications

Figure A.9. Data Science: The Eclectic Discipline

Information technology is about gathering data, distributing data, taking care of data, and providing secured access to data. Information retrieval and selection are important, as are data indexing and storage. Increasingly, organizations are turning to distributed solutions, storing data and documents across many computers. Data preparation is a big part of any sports analytics project. Data must be cleaned and organized prior to analysis.

Measurement is about getting to the right data, measures that

are reliable and valid. It is about understanding the meaning of measures, noting that the validity of a measure is defined by that measure's relationships with other measures. We check data quality, validate, transform, and annotate data. We make data from data, defining new measures as combinations of other measures. To do a good job of measurement, we need to understand the research context, be it sport or business.

Statistics is about doing the right things with data. We sample from populations. We draw inferences from samples. We explore, looking at data visualizations or statistical graphics. We compute summary statistics, reducing data to their essence. We build models from data for making inferences and predictions. We learn from data.

Data science is closely related to information technology and computer science. Like operations research, data science is concerned with methods and models for making informed, sometimes optimal, business decisions. Context is important, whether working on the performance or management sides of team sports. Data scientists need to be multilingual, understanding the languages of information systems, statistics, sport, and business.

Exhibit A.6 (page 252) shows the R program for making the map of the ten disciplines, showing their relationships. The program reads the matrix of similarity judgments and uses multidimensional scaling to define the map.

Exhibit A.1. Programming the Anscombe Quartet (Python)

[Click here to view code image](#)

```
# The Anscombe Quartet (Python)
```

```
# demonstration data from
# Anscombe, F. J. 1973, February. Graphs in
# statistical analysis.
# The American Statistician 27: 1721.

# prepare for Python version 3x features and
functions
from __future__ import division,
print_function

# import packages for Anscombe Quartet
demonstration
import pandas as pd # data frame operations
import numpy as np # arrays and math
functions
import statsmodels.api as sm # statistical
models (including regression)
import matplotlib.pyplot as plt # 2D plotting

# define the anscombe data frame using
dictionary of equal-length lists
anscombe = pd.DataFrame({'x1' : [10, 8, 13, 9,
11, 14, 6, 4, 12, 7, 5],
'x2' : [10, 8, 13, 9, 11, 14, 6, 4, 12, 7,
5],
'x3' : [10, 8, 13, 9, 11, 14, 6, 4, 12, 7,
5],
'x4' : [8, 8, 8, 8, 8, 8, 19, 8, 8, 8],
'y1' : [8.04, 6.95, 7.58, 8.81, 8.33,
9.96, 7.24, 4.26, 10.84, 4.82, 5.68],
'y2' : [9.14, 8.14, 8.74, 8.77, 9.26,
8.1, 6.13, 3.1, 9.13, 7.26, 4.74],
'y3' : [7.46, 6.77, 12.74, 7.11, 7.81,
8.84, 6.08, 5.39, 8.15, 6.42, 5.73],
'y4' : [6.58, 5.76, 7.71, 8.84, 8.47,
7.04, 5.25, 12.5, 5.56, 7.91, 6.89]})

# fit linear regression models by ordinary
least squares
set_I_design_matrix =
```

```
sm.add_constant(anscombe['x1'])
set_I_model = sm.OLS(anscombe['y1'],
set_I_design_matrix)
print(set_I_model.fit().summary())

set_II_design_matrix =
sm.add_constant(anscombe['x2'])
set_II_model = sm.OLS(anscombe['y2'],
set_II_design_matrix)
print(set_II_model.fit().summary())

set_III_design_matrix =
sm.add_constant(anscombe['x3'])
set_III_model = sm.OLS(anscombe['y3'],
set_III_design_matrix)
print(set_III_model.fit().summary())

set_IV_design_matrix =
sm.add_constant(anscombe['x4'])
set_IV_model = sm.OLS(anscombe['y4'],
set_IV_design_matrix)
print(set_IV_model.fit().summary())

# create scatter plots
fig = plt.figure()
set_I = fig.add_subplot(2, 2, 1)
set_I.scatter(anscombe['x1'],anscombe['y1'])
set_I.set_title('Set I')
set_I.set_xlabel('x1')
set_I.set_ylabel('y1')
set_I.set_xlim(2, 20)
set_I.set_ylim(2, 14)

set_II = fig.add_subplot(2, 2, 2)
set_II.scatter(anscombe['x2'],anscombe['y2'])
set_II.set_title('Set II')
set_II.set_xlabel('x2')
set_II.set_ylabel('y2')
set_II.set_xlim(2, 20)
set_II.set_ylim(2, 14)
```

```
set_III = fig.add_subplot(2, 2, 3)
set_III.scatter(anscombe['x3'],anscombe['y3'])
set_III.set_title('Set III')
set_III.set_xlabel('x3')
set_III.set_ylabel('y3')
set_III.set_xlim(2, 20)
set_III.set_ylim(2, 14)

set_IV = fig.add_subplot(2, 2, 4)
set_IV.scatter(anscombe['x4'],anscombe['y4'])
set_IV.set_title('Set IV')
set_IV.set_xlabel('x4')
set_IV.set_ylabel('y4')
set_IV.set_xlim(2, 20)
set_IV.set_ylim(2, 14)

plt.subplots_adjust(left=0.1, right=0.925,
top=0.925, bottom=0.1,
wspace = 0.3, hspace = 0.4)
plt.savefig('fig_anscombe_Python.pdf',
bbox_inches = 'tight', dpi=None,
facecolor='w', edgecolor='b',
orientation='portrait', papertype=None,
format=None, transparent=True,
pad_inches=0.25, frameon=None)

# Suggestions for the student:
# See if you can develop a quartet of your
own,
# or perhaps just a duet, two very different
data sets
# with the same fitted model.
```

Exhibit A.2. Programming the Anscombe Quartet (R)

[Click here to view code image](#)

```
# The Anscombe Quartet (R)

# demonstration data from
# Anscombe, F. J. 1973, February. Graphs in
# statistical analysis.
# The American Statistician 27: 1721.

# define the anscombe data frame
anscombe <- data.frame(
  x1 = c(10, 8, 13, 9, 11, 14, 6, 4, 12, 7,
5),
  x2 = c(10, 8, 13, 9, 11, 14, 6, 4, 12, 7,
5),
  x3 = c(10, 8, 13, 9, 11, 14, 6, 4, 12, 7,
5),
  x4 = c(8, 8, 8, 8, 8, 8, 19, 8, 8, 8),
  y1 = c(8.04, 6.95, 7.58, 8.81, 8.33,
9.96, 7.24, 4.26, 10.84, 4.82, 5.68),
  y2 = c(9.14, 8.14, 8.74, 8.77, 9.26, 8.1,
6.13, 3.1, 9.13, 7.26, 4.74),
  y3 = c(7.46, 6.77, 12.74, 7.11, 7.81,
8.84, 6.08, 5.39, 8.15, 6.42, 5.73),
  y4 = c(6.58, 5.76, 7.71, 8.84, 8.47,
7.04, 5.25, 12.5, 5.56, 7.91, 6.89))

# show results from four regression analyses
with(anscombe, print(summary(lm(y1 ~ x1, data
= anscombe))))
with(anscombe, print(summary(lm(y2 ~ x2, data
= anscombe))))
with(anscombe, print(summary(lm(y3 ~ x3, data
= anscombe))))
with(anscombe, print(summary(lm(y4 ~ x4, data
= anscombe)))))

# place four plots on one page using standard
# R graphics
# ensuring that all have the same scales
# for horizontal and vertical axes
pdf(file = "fig_anscombe_R.pdf", width = 8.5,
```

```

height = 8.5)
par(mfrow=c(2,2), mar=c(5.1, 4.1, 4.1, 2.1))
with(anscombe, plot(x1, y1, xlim=c(2,20),
ylim=c(2,14), pch = 19,
col = "darkblue", cex = 1.5, las = 1, xlab
= "x1", ylab = "y1"))
title("Set I")
with(anscombe, plot(x2, y2, xlim=c(2,20),
ylim=c(2,14), pch = 19,
col = "darkblue", cex = 1.5, las = 1, xlab
= "x2", ylab = "y2"))
title("Set II")
with(anscombe, plot(x3, y3, xlim=c(2,20),
ylim=c(2,14), pch = 19,
col = "darkblue", cex = 1.5, las = 1, xlab
= "x3", ylab = "y3"))
title("Set III")
with(anscombe, plot(x4, y4, xlim=c(2,20),
ylim=c(2,14), pch = 19,
col = "darkblue", cex = 1.5, las = 1, xlab
= "x4", ylab = "y4"))
title("Set IV")
dev.off()

# par(mfrow=c(1,1),mar=c(5.1, 4.1, 4.1,
2.1)) # return to plotting defaults

# Suggestions for the student:
# See if you can develop a quartet of your
own,
# or perhaps just a duet, two very different
data sets
# with the same fitted model.

```

Exhibit A.3. Making Differential Runs Plots for Baseball (R)

[Click here to view code image](#)

```

# Making Differential Runs Plots for Baseball
(R)

# data visualization with R standard graphics

excess.runs.data.frame.ARI <-
read.csv("MLB_2007_ARI_data_frame.csv")

pdf(file=paste("fig_differential_runs_ARI.pdf", sep
               width = 11, height = 8.5)
par(mfrow=c(1,1), xpd=NA, cex=1)

plot(excess.runs.data.frame.ARI$excess.runs,
     type="h",
     las=1, xlab="Game Number and Opponent",
     ylab="Differential Runs",
     ylim=c(-(max(abs(excess.runs.data.frame.ARI$ex
+ 1),
      (max(abs(excess.runs.data.frame.ARI$excess.run
+ 1))))
abline(h=0,lty="solid",xpd=FALSE,lwd=.5)

legend("topright", title=NULL,
       legend=c("Home Game ","Away Game "),
       pch=c(19,21))

# plot all as open circles first
points(excess.runs.data.frame.ARI$excess.runs,
       pch=21)

# then fill in the circles for the home games
for(i in
seq(along=excess.runs.data.frame.ARI$game.number))
  if(excess.runs.data.frame.ARI$home.away[i]=="hom
  points(excess.runs.data.frame.ARI$game.number[i]
          excess.runs.data.frame.ARI$excess.runs[i],
          pch=19)

for(i in
seq(along=excess.runs.data.frame.ARI$game.number))

```

```
text(excess.runs.data.frame.ARI$game.number[i], -  
      (max(abs(excess.runs.data.frame.ARI$excess.run  
      + 1)),  
       excess.runs.data.frame.ARI$other.team.label[i]  
       cex=.85, pos=4, srt=90)  
  
text(excess.runs.data.frame.ARI$game.number[1],  
      (max(abs(excess.runs.data.frame.ARI$excess.run  
      + 1)),  
       map.team.to.name(excess.runs.data.frame.ARI$th  
  
dev.off()
```

Exhibit A.4. Moving Fraction Plot: A Basketball Example (R)

[Click here to view code image](#)

```
# Moving Fraction Plot: A Basketball Example  
(R)  
  
library(grid) # graphics utilities needed for  
split-plotting  
library(ggplot2) # graphics package with  
ribbon plot  
  
# read game summary data for the Oklahoma City  
Thunder  
OKC_data <-  
read.csv("okc_data_2014_2015.csv", stringsAsFactor  
= FALSE)  
  
# check the structure of the data  
print(str(OKC_data))  
  
# create winning fraction plot to explore  
streakiness during the season  
# a window of twelve games is used following  
Albert and Bennett (2001)
```

```

# add game number to data frame
OKC_data$game_number <- seq(1:nrow(OKC_data))

# set binary indicator if this team has won
OKC_data$win_bin <- rep(0, length =
nrow(OKC_data))
for (i in seq(along = OKC_data$win_bin))
  if (OKC_data$win_team_code[i] == "OKC")
    OKC_data$win_bin[i] <- 1

# window across the season twelve games at a
time
OKC_data$win_window <- rep(NA, length =
nrow(OKC_data))
for (i in seq(along = OKC_data$win_window)) {
  if (i > 6)
    OKC_data$win_window[i] <-
      mean(OKC_data$win_bin[(i - 6):(i
+ 5)]) * 100
}

moving_fraction_plotting_frame <-
  OKC_data[, c("game_number", "win_window")]
moving_fraction_plotting_frame$ymax_topwhite
<-
  rep(NA, length =
nrow(moving_fraction_plotting_frame))
moving_fraction_plotting_frame$ymax_bottomwhite
<-
  rep(NA, length =
nrow(moving_fraction_plotting_frame))
moving_fraction_plotting_frame$ymin_topwhite
<-
  rep(NA, length =
nrow(moving_fraction_plotting_frame))
moving_fraction_plotting_frame$ymin_bottomwhite
<-
  rep(NA, length =

```

```

nrow(moving_fraction_plotting_frame))

for (i in seq(along =
moving_fraction_plotting_frame$win_window)) {
  if
(is.na(moving_fraction_plotting_frame$win_window[i
{
  moving_fraction_plotting_frame$ymax_topwhi
<- 100
  moving_fraction_plotting_frame$ymin_topwhi
<- 50.1
  moving_fraction_plotting_frame$ymax_bottom
<- 49.99
  moving_fraction_plotting_frame$ymin_bottom
<- 0
}
  if
(!is.na(moving_fraction_plotting_frame$win_window[
{
  if
(moving_fraction_plotting_frame$win_window[i]
> 50) {
    moving_fraction_plotting_frame$ymax_to
<- 100
    moving_fraction_plotting_frame$ymin_to
<-
    moving_fraction_plotting_frame$win
    moving_fraction_plotting_frame$ymax_bo
<- 49.99
    moving_fraction_plotting_frame$ymin_bo
<- 0
}
  if
(moving_fraction_plotting_frame$win_window[i]
<= 50) {
    moving_fraction_plotting_frame$ymax_to
<- 100
    moving_fraction_plotting_frame$ymin_to
<- 50.1
    moving_fraction_plotting_frame$ymax_bo

```

```

<-
  moving_fraction_plotting_frame$win_win
  moving_fraction_plotting_frame$ymin_bo
<- 0
}
}
}

greenmin <- 50.01
greenmax <- 100
redmin <- 0
redmax <- 49.99

pdf(file = "fig_moving_fraction_plot.pdf",
width = 8.8, height = 8.5)
ggobject <- ggplot() +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=greenmin,
  ymax=greenmax),
  stat="identity", colour="white", fill="darkgreen") +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=redmin,
  ymax=redmax),
  stat="identity", colour="white", fill="darkred") +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=ymin_topwhite,
  ymax=ymax_topwhite),
  stat="identity", colour="white", fill="white") +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number,
  ymin=ymin_bottomwhite, ymax=ymax_bottomwhite),
  stat="identity", colour="white", fill="white") +
  annotate("segment", x = 0, xend = 82, y =
  49.99, yend = 50.01) +
  ylab("Moving Fraction (Winning Percentage)") +
  xlab("Sequence of Games")

print(ggobject)

```

```
dev.off()
```

Exhibit A.5. Visualizing Basketball Games (R)

[Click here to view code image](#)

```
# Visualizing Basketball Games (R)

library(lubridate) # data/time functions and
lakers data frame
# lubridate imports plyr package
library(ggplot2) # statistical graphics, grid
graphics assumed

# functions used with grid graphics to split
the plotting region
# to set margins and to plot more than one
ggplot object on one page/screen
vplayout <- function(x, y)
viewport(layout.pos.row=x, layout.pos.col=y)

# user-defined function to plot a ggplot
object with margins
ggplot.print.with.margins <-
function(ggplot.object.name,
        left.margin.pct=10,
        right.margin.pct=10,top.margin.pct=10,bottom.
        { # begin function for printing ggplot
objects with margins
        # margins expressed as percentages of
total... use integers
        grid.newpage()
        pushViewport(viewport(layout=grid.layout(100,1
        print(ggplot.object.name,
        vp=vplayout((0 + top.margin.pct):(100 -
bottom.margin.pct),
        (0 + left.margin.pct):(100 -
```

```
right.margin.pct)))
} # end function for printing ggplot
objects with margins

# lakers data frame from lubridate includes
# play-by-play data for the Los Angeles Lakers
2008-2009 season
# original data from Parker (2010)
http://www.basketballgeek.com/data/
# date: year/month/day text to be converted to
date object
# opponent: three-character abbreviation for
other team
# game_type: home or away for Los Angeles
Lakers
# time: clock time remaining minutes:seconds
converted to duration
# period: period of play 1, 2, 3, or 4
# etype: one of ten event types: foul, free
throw, jump ball,
#      rebound, shot, sub, timeout,
turnover, violation
# team: three-character abbreviation for team
#      or OFF for neither team as with a jump
ball
# player: one of 371 players involved in Los
Angeles Lakers games
# result: missed or made
# points: points scored
# type: one of 74 event descriptions: hook,
off, layup,
#      shooting, personal, jump, pullup jump,
#      def, driving layup, driving finger
roll layup, regular,
#      offensive, 3pt, turnaround jump,
putback layup,
#      slam dunk, tip, dunk, defensive
goaltending,
#      hook bank, running layup, official,
driving slam dunk,
```



```
        "MIL", "MIN", "NJN", "NOH",
"NYK",
        "OKC", "ORL", "PHI", "PHX",
"POR",
        "SAC", "SAS", "TOR", "UTA",
"WAS")  
  
# Note. Charlotte Bobcats later became the  
Charlotte Hornets  
# New Orleans Hornets later became the New  
Orleans Pelicans  
team_name_2008 = c("Atlanta Hawks", "Boston  
Celtics",
                    "Charlotte Hornets",
"Chicago Bulls",
                    "Cleveland Cavaliers",
"Dallas Mavericks",
                    "Denver Nuggets", "Detroit
Pistons",
                    "Golden State Warriors",
"Houston Rockets",
                    "Indiana Pacers", "Los
Angeles Clippers",
                    "Los Angeles Lakers",
"Memphis Grizzlies",
                    "Miami Heat", "Milwaukee
Bucks",
                    "Minnesota Timberwolves",
>New Jersey Nets",
                    "New Orleans Hornets", "New
York Knicks",
                    "Oklahoma City Thunder",
"Orlando Magic",
                    "Philadelphia 76ers",
"Phoenix Suns",
                    "Portland Trail Blazers",
"Sacramento Kings",
                    "San Antonio Spurs",
>Toronto Raptors",
                    "Utah Jazz", "Washington
```

```
Wizards")
# Note. Charlotte Bobcats became the Charlotte
Hornets
# New Orleans Hornets became the New Orleans
Pelicans
# New Jersey Nets became the Brooklyn Nets

# lakers$date <- ymd(lakers$date) # code as
date variable
lakers$time <- ms(lakers$time) # code as
period object
lakers$time <- as.duration(lakers$time) #
code as durations

# convert time to gametime with periods of 12
minutes, overtime 5 minutes
# here we convert to minutes and fractions of
minutes
lakers$gametime <- dminutes(c(12, 24, 36, 48,
53)[lakers$period]) -
    as.duration(lakers$time)
lakers$minutes <-
as.numeric(seconds(lakers$gametime))/60
print(str(lakers)) # examine stucture of the
data frame
# get rid of observations with team OFF (jump
ball)
lakers_games <- lakers[(lakers$team != "OFF"),]

# route plots to external plotting device...
pdf file
pdf(file =
"plot_lakers_basketball_2008_2009.pdf",
width = 8.5, height = 8.5)
#
-----
# cycle through all Lakers games for the
season
gamedate <- unique(lakers_games$date)
```

```
for (igame in seq(along = gamedate)) { # begin
  for-loop for all games
    this_game <-
      lakers_games[lakers_games$date ==
        gamedate[igame],]

    # work with the current game, compute score as
    # cumulative sum for each team
    # using ddply function from the plyr package,
    # required for lubridate package
    this_game_scores <- ddply(this_game,
      "team", transform,
      score = cumsum(points))

    # identify team names and scores for this game
    first_team_name <-
      team_name_2008[which(team_code_2008 == "LAL")]
    second_team_name <-
      team_name_2008[which(team_code_2008 ==
        this_game_scores$opponent[1])]
    first_team_score <-
      max(this_game_scores[(this_game_scores$tea
        == "LAL"), "score"])
    second_team_score <-
      max(this_game_scores[(this_game_scores$tea
        == "LAC"), "score"])

    # summary for this game to be used in plot
    title
      this_game_summary_text <-
        paste(gsub(" UTC", "", " ",
          ymd(this_game_scores$date)[1]),
          " ", first_team_name, " ",
          this_game_scores$game_type[1],
          " (", first_team_score, ") versus ",
          second_team_name, " ", "(",
          second_team_score, ") ", sep = "")
```

```
# create visualization for this game with
annotation shading
# for the second and fourth 12-minute periods
in the game
# and with customized legend for the step
function score lines
ggplot_object <-
  ggplot(data = subset(this_game_scores,
team == "LAL"),
  aes(x = minutes, y = score)) +
  layer(geom = "step", size = 0.75,
colour = "purple") +
  layer(data = subset(this_game_scores,
team ==
this_game_scores$opponent[1]),
  geom = "step", size = 1.5, colour
= "blue") +
  annotate("rect", xmin = 12.01, xmax =
24.00,
  ymin =
min(this_game_scores$score),
  ymax =
max(this_game_scores$score),
  alpha = 0.1, fill = "black") +
  annotate("rect", xmin = 36.01, xmax =
48.00,
  ymin =
min(this_game_scores$score),
  ymax =
max(this_game_scores$score),
  alpha = 0.1, fill = "black") +
  annotate("rect", xmin = 0, xmax =
24.00,
  ymin = max(this_game_scores$score)
- 15,
  ymax =
max(this_game_scores$score),
  alpha = 1, colour = "darkgrey",
fill = "white") +
  geom_segment(x = 1.0, xend = 3.0,
```

```

      y = max(this_game_scores$score) -
5,
      yend = max(this_game_scores$score)
- 5,
      colour = "purple", size = 0.75) +
geom_segment(x = 1.0, xend = 3.0,
      y = max(this_game_scores$score) -
10,
      yend = max(this_game_scores$score)
- 10,
      colour = "blue", size = 1.5) +
annotate("text", x = 4, y =
max(this_game_scores$score) - 5,
      colour = "black", alpha = 0.9,
size = 4.5,
      label = first_team_name, hjust =
0) +
      annotate("text", x = 4, y =
max(this_game_scores$score) - 10,
      colour = "black", alpha = 0.9,
size = 4.5,
      label = second_team_name, hjust =
0) +
      xlab("Game Minutes") + ylab("Score") +
ggtitle(this_game_summary_text) +
theme(axis.title.x = element_text(size
= rel(1.25))) +
      theme(axis.title.y = element_text(size
= rel(1.25)))

ggplot.print.with.margins(ggplot_object,
      left.margin.pct = 5, right.margin.pct
= 5,
      top.margin.pct = 5, bottom.margin.pct
= 5)

} # end for-loop for all games
dev.off() # closer the pdf plotting device

```

Exhibit A.6. Seeing Data Science as an Eclectic Discipline (R)

[Click here to view code image](#)

```
# Seeing Data Science as an Eclectic
Discipline (R)

# Program for multidimensional scaling of ten
academic disciplines

library(MASS) # includes functions for
multidimensional scaling
library(wordcloud) # textplot utility to
avoid overlapping text

USE_METRIC_MDS <- FALSE # metric versus
nonmetric toggle

# define a utility function for converting a
distance structure
# to a distance matrix as required for some
routines and
# for printing of the complete matrix for
visual inspection.
make.distance.matrix <-
function(distance_structure)
{
  n <- attr(distance_structure, "Size")
  full <- matrix(0,n,n)
  full[lower.tri(full)] <-
distance_structure
  full+t(full)
}

# enter data into a distance structure as
required for various
# distance-based routines. That is, we enter
the upper triangle
# of the distance matrix as a single vector of
```

```
distances
distance_structure <-
  as.single(c(35, 45, 40, 43, 18, 32, 3,
20, 19,
8, 12, 9, 33, 30,
25, 7, 21,
1, 15, 16, 13, 41,
17, 10,
28, 36, 31,
42, 5, 26,
39, 38,
37, 4, 44,
2, 11,
34, 24,
14,
29, 6,
23,
22,
2
# provide a character vector of discipline
names
disciplines <- c("Marketing", "Operations
Research", "Mathematics",
"Statistics", "Computer Science",
"Accounting", "Finance",
"Management", "Data Science", "Economics")

attr(distance_structure, "Size") <-
length(disciplines) # set size attribute

# check to see that the distance structure has
been entered correctly
# by converting the distance structure to a
distance matrix
# using the utility function
make.distance.matrix, which we had defined
distance_matrix <-
unlist(make.distance.matrix(distance_structure))
cat("\n", "Distance Matrix of Academic
```

```

Disciplines","\n")
print(distance_matrix)

if (USE_METRIC_MDS)
{
  # apply the metric multidimensional
  scaling algorithm and plot the map
  mds_solution <-
  cmdscale(distance_structure, k=2, eig=T)
  First_Dimension <- mds_solution$points[,1]
  Second_Dimension <-
  mds_solution$points[,2]
}
# apply the nonmetric multidimensional scaling
algorithm
# this is more appropriate for rank-order data
# and provides a more satisfactory solution
here

if (!USE_METRIC_MDS)
{
  mds_solution <- isoMDS(distance_matrix, k
= 2, trace = FALSE)
}

pdf(file =
"plot_nonmetric_mds_data_science.pdf",
width=11, height=8.5) # opens pdf plotting
device
# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))

# original solution
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2]

# set up the plot but do not plot points...
use names for points
plot(First_Dimension, Second_Dimension, type =

```

```

"n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) #
first page of pdf plots
# We plot the city names in the locations
where points normally go.
text(First_Dimension, Second_Dimension, labels
= disciplines,
      offset = 0.0, cex = 1.5)
title("Academic Disciplines (initial
solution)")

# reflect the horizontal dimension
# multiply the first dimension by -1 to get
reflected image
First_Dimension <- mds_solution$points[,1] *
-1
Second_Dimension <- mds_solution$points[,2]
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) #
second page of pdf plots
text(First_Dimension, Second_Dimension, labels
= disciplines,
      offset = 0.0, cex = 1.5)
title("Academic Disciplines (horizontal
reflection)")

# reflect the vertical dimension
# multiply the section dimension by -1 to get
reflected image
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2] *
-1
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) #
third page of pdf plots
text(First_Dimension, Second_Dimension, labels
= disciplines,
      offset = 0.0, cex = 1.5)

```

```

title("Academic Disciplines (vertical
reflection)")
# multiply the first and second dimensions by
-1
# for reflection in both horizontal and
vertical directions
First_Dimension <- mds_solution$points[,1] *
-1
Second_Dimension <- mds_solution$points[,2] *
-1
plot(First_Dimension, Second_Dimension, type =
"n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) # fourth page of pdf plots
text(First_Dimension, Second_Dimension, labels
= disciplines,
      offset = 0.0, cex = 1.5)
title("Academic Disciplines (horizontal and
vertical reflection)")
dev.off() # closes the pdf plotting device

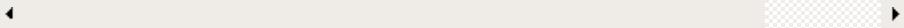
pdf(file =
"plot_pretty_original_mds_data_science.pdf",
      width=8.5, height=8.5) # opens pdf
plotting device
# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] # no reflection
Second_Dimension <-
mds_solution$points[,2] # no reflection
# wordcloud utility for plotting with no
overlapping text
textplot(x = First_Dimension,
      y = Second_Dimension,
      words = disciplines,
      show.lines = FALSE,
      xlim = c(-28, 28), # extent of horizontal
axis range

```

```
    ylim = c(-20, 20), # extent of vertical
axis range
    xaxt = "n", # suppress tick marks
    yaxt = "n", # suppress tick marks
    cex = 1.25, # size of text points
    mgp = c(0.85, 1, 0.85), # position of
axis labels
    cex.lab = 1.5, # magnification of axis
label text
    xlab = "Methods . . . . . . . . .
. Applications",
    ylab = "Mostly Numbers . . . . Numbers and
Words")
dev.off() # closes the pdf plotting device

pdf(file =
"plot_reflected_mds_data_science.pdf",
width=8.5, height=8.5) # opens pdf
plotting device
# use par(mar = c(bottom, left, top, right))
to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] *
-1 # reflect horizontal
Second_Dimension <- mds_solution$points[,2] *
-1 # reflect vertical
# wordcloud utility for plotting with no
overlapping text
textplot(x = First_Dimension,
        y = Second_Dimension,
        words = disciplines,
        show.lines = FALSE,
        xlim = c(-28, 28), # extent of horizontal
axis range
        ylim = c(-20, 20), # extent of vertical
axis range
        xaxt = "n", # suppress tick marks
        yaxt = "n", # suppress tick marks
        cex = 1.25, # size of text points
        mgp = c(0.85, 1, 0.85), # position of
```

```
axis labels
  cex.lab = 1.5, # magnification of axis
label text
  xlab = "Applications . . . . . . . .
. . . Methods",
  ylab = "Words and Numbers . . . Mostly
Numbers")
dev.off() # closes the pdf plotting device
```



B. Professional Leagues and Teams

Conference	Team Name	Abbreviation
Eastern	Atlanta Dream	ATL
	Chicago Sky	CHI
	Connecticut Sun	CON
	Indiana Fever	IND
	New York Liberty	NYL
	Washington Mystics	WAS
Western	Los Angeles Sparks	LAS
	Minnesota Lynx	MIN
	Phoenix Mercury	PHO
	San Antonio Stars	SAS
	Seattle Storm	SEA
	Tulsa Shock	TUL

Table B.1. *Women's National Basketball Association
(WNBA)*

League	Division	Team Name	Abbreviation
American	East	Baltimore Orioles	BAL
		Boston Red Sox	BOS
		New York Yankees	NYY
		Tampa Bay Rays	TBR
		Toronto Blue Jays	TOR
	Central	Chicago White Sox	CHW
		Cleveland Indians	CLE
		Detroit Tigers	DET
		Kansas City Royals	KCR
		Minnesota Twins	MIN
National	West	Houston Astros	HOU
		Los Angeles Angels	LAA
		Oakland A's	OAK
		Seattle Mariners	SEA
		Texas Rangers	TEX
	East	Atlanta Braves	ATL
		Miami Marlins	MAI
		New York Mets	NYM
		Philadelphia Phillies	PHI
		Washington Nationals	WSN
	Central	Chicago Cubs	CHC
		Cincinnati Reds	CIN
		Milwaukee Brewers	MIL
		Pittsburgh Pirates	PIT
		St. Louis Cardinals	STL
	West	Arizona Diamondbacks	ARI
		Colorado Rockies	COL
		Los Angeles Dodgers	LAD
		San Diego Padres	SDP
		San Francisco Giants	SFG

Table B.2. Major League Baseball (MLB)

Conference	Team Name	Abbreviation
Eastern	Chicago Fire	CHI
	Columbus Crew SC	CCR
	D.C. United	DCU
	Montreal Impact	MON
	New England Revolution	NER
	New York City FC	NYC
	New York Red Bulls	NYR
	Orlando City SC	ORL
	Philadelphia Union	PHI
	Toronto FC	WAS
Western	Colorado Rapids	COL
	FC Dallas	FTD
	Houston Dynamo	HOU
	LA Galaxy	LAG
	Portland Timbers	POR
	Real Salt Lake	RSL
	San Jose Earthquakes	SAN
	Seattle Sounders FC	SEA
	Sporting Kansas City	SKC
	Vancouver Whitecaps FC	VAN

Table B.3. Major League Soccer (MLS)

Conference	Division	Team Name	Abbreviation
Eastern	Atlantic	Boston Celtics	BOS
		Brooklyn Nets	BKN
		New York Knicks	NYK
		Philadelphia 76ers	PHI
		Toronto Raptors	TOR
	Central	Chicago Bulls	CHI
		Cleveland Cavaliers	CLE
		Detroit Pistons	DET
		Indiana Pacers	IND
		Milwaukee Bucks	MIL
Western	Southeast	Atlanta Hawks	ATL
		Charlotte Hornets	CHA
		Miami Heat	MIA
		Orlando Magic	ORL
		Washington Wizards	WAS
	Southwest	Dallas Mavericks	DAL
		Houston Rockets	HOU
		Memphis Grizzlies	MEM
		New Orleans Pelicans	NOP
		San Antonio Spurs	SAS
	Northwest	Denver Nuggets	DEN
		Minnesota Timberwolves	MIN
		Oklahoma City Thunder	OKC
		Portland Trail Blazers	POR
		Utah Jazz	UTA
Pacific	Pacific	Golden State Warriors	GSW
		Los Angeles Clippers	LAC
		Los Angeles Lakers	LAL
		Phoenix Suns	PHX
		Sacramento Kings	SAC

Table B.4. National Basketball Association (NBA)

Conference	Division	Team Name	Abbreviation
American	East	Buffalo Bills	BUF
		Miami Dolphins	MAI
		New England Patriots	NE
		New York Jets	NYJ
		Baltimore Ravens	BAL
	North	Cincinnati Bengals	CIN
		Cleveland Browns	CLE
		Pittsburgh Steelers	PIT
		Houston Texans	HOU
		Indianapolis Colts	IND
National	South	Jacksonville Jaguars	JAC
		Tennessee Titans	TEN
		Denver Broncos	DEN
		Kansas City Chiefs	KC
		Oakland Raiders	OAK
	West	San Diego Chargers	SD
		Dallas Cowboys	DAL
		New York Giants	NYG
		Philadelphia Eagles	PHI
		Washington Redskins	WAS
American	North	Chicago Bears	CHI
		Detroit Lions	DET
		Green Bay Packers	GB
		Minnesota Vikings	MIN
	South	Atlanta Falcons	ATL
		Carolina Panthers	CAR
		New Orleans Saints	NO
		Tampa Bay Buccaneers	TB
		Arizona Cardinals	ARI
National	West	San Francisco 49ers	SF
		Seattle Seahawks	SEA
		St. Louis Rams	STL

Table B.5. National Football League (NFL)

Conference	Division	Team Name	Abbreviation
Eastern	Atlantic	Boston Bruins	BOS
		Buffalo Sabres	BUF
		Detroit Red Wings	DET
		Florida Panthers	FLA
		Montreal Canadiens	MON
		Ottawa Senators	OTW
		Tampa Bay Lightning	TBL
	Metropolitan	Toronto Maple Leafs	TOR
		Carolina Hurricanes	CAR
		Columbus Blue Jackets	CBJ
Western	Central	New Jersey Devils	NJD
		New York Islanders	NYI
		New York Rangers	NYR
		Philadelphia Flyers	PHI
		Pittsburgh Penguins	PIT
		Washington Capitals	WAS
		Chicago Blackhawks	CHI
	Pacific	Colorado Avalanche	COL
		Dallas Stars	DAL
		Minnesota Wild	MIN
		Nashville Predators	NAS
		St. Louis Blues	STL
		Winnipeg Jets	WIN
		Anaheim Ducks	ANA
		Arizona Coyotes	ARI
		Calgary Flames	CAL
		Edmonton Oilers	EDM
		Los Angeles Kings	LAK
		San Jose Sharks	SJS
		Vancouver Canucks	VAN

Table B.6. National Hockey League (NHL)

Data Science Glossary

adjacency matrix Consider a network with nodes and links between nodes. We let $x_{ij} = 1$ if node i is linked to node j and let $x_{ij} = 0$ otherwise. Undirected networks yield symmetric matrices. Directed networks yield asymmetric matrices.

agent Also called an “artificial agent.” A program that automatically performs duties for a computer user or a program that behaves like a person within a limited domain of operation.

agent-based modeling Simulation method that can build on discrete event simulation. In agent-based modeling of networks, we acknowledge the fact that individual nodes may differ from one another in roles, motives, behavior, or interactions. Agent-based techniques provide a facility for modeling networks that change with time (dynamic networks).

algebraic modeling system Computer implementation of mathematical programming that separates the structure (algebra) of constrained optimization problems from input data defining the parameters of those problems. Especially useful in sensitivity testing and stochastic programming.

Apache Software Foundation Established in 1999, an open-source community for software development and maintenance. Provides essential tools for operating on the World Wide Web.

application programming interface (API) Method for

processing data requests between systems. Client-side programs interact with server-side applications to obtain data within a defined structure, such as JavaScript Object Notation (JSON) or extensible markup language (XML). Often utilizes a representational state transfer (REST) for data interchange.

This glossary defines technical terms and abbreviations used in data science. Readers desiring more complete dictionaries of computer and Internet terms can refer to Hale and Scanlon (1999) and Downing, Covington, Covington, Barrett, and Covington (2012). Moran and Hunt (2009) review terms relevant to website performance in search. Hyslop (2010), Casciano (2011), and Gasston (2011; 2013) review current HTML and CSS concepts. Robbins (2003) provides an introductory treatment for web designers. Amor (2002) and Tannenbaum and Wetherall (2010) review data communications, the Internet, and World Wide Web. See Miller (2015d) for additional discussion of terminology from web and network data science.

ARPANET Military-sponsored wide-area network, precursor to the Internet.

ASP Abbreviation for an Application Services Provider, a firm that provides outsourced application services. Also, in Microsoft systems, Active Server Pages.

association rules Machine learning models for identifying most frequently occurring item-sets or combinations of binary variables. Useful in market basket analysis and recommender systems.

bandwidth An expression that relates to the speed at which

information travels over a network. The words “low bandwidth” mean slower speed and “high bandwidth” mean faster speed.

Bayesian statistics Inferential statistics built on probability theory, in particular, Bayes Theorem. Data are used to update a prior probability distribution, which represents researcher beliefs or assumptions about the world. The result of the updating process is a posterior probability distribution.

best-case/worst-case approach In decision analysis, the process of exploring extreme outcomes described as best case and worst case. A third situation commonly included in a best-case/worst-case approach is the expected outcome between the two extremes.

betweenness centrality Usually computed for nodes, this is an index that reflects the degree to which a node lies on the shortest path between other pairs of nodes. An index of node importance. For a network link, betweenness is similarly computed and represents the degree to which a link lies on the shortest path between pairs of nodes.

big data Term initiated by information technology firms to refer to the glut of data being collected and stored. Data are arriving at a faster pace and in larger quantities than ever before. Distributed, scalable database systems may be needed to accommodate the volume, velocity, and variety of data. NoSQL systems are often used in this context.

binary variable A variable that takes only two values, often coded as 0 and 1.

blog Shortened form of “weblog.” An asynchronous method of

communication on the web. Often a personal column or diary, composed of short entries, perhaps relating to a specific topic. A blog can also involve multiple users interacting with a host or moderator, in which case it is a message board or asynchronous focus group with the discussion organized around themes or threads.

bootstrap sampling Resampling technique that involves repeated sampling with replacement from a sample. Provides method for estimating sampling distributions and the properties of statistics under random sampling. Working with a sample of N observations, each bootstrap sample is a random sample of size N with replacement. Like cross-validation, bootstrap sampling can be used to evaluate predictive modeling techniques.

boundary (of a network) The extent of the network with its included nodes and links. Defines where the network begins and ends—the population of interest.

bps Bits per second. Measure of speed across a communications link. Sometimes referred to as the “baud rate.”

brute force approach Process of exploring many alternatives (thousands or millions of options), taking advantage of computing power.

C, C++, C# Variations on the popular computer language C. Programs written in these languages require compilation prior to execution. C is a procedural language. C++ and C# (pronounced “C sharp”) are object-oriented extensions of C.

CART Abbreviation for “classification and regression trees.” Recursive partitioning method for tree-structured models

used for both classification and regression problems. A machine learning method.

cascading style sheet (CSS) Set of HTML rules for a website.

Current version is CSS3.

chat room A virtual meeting room on the Internet where participants gather and speak to each other in real-time. May be public (open to anybody, space permitting) or private (requiring passwords to get in). Also called a “real-time focus group” or “synchronous focus group.” An online focus group with the moderator and participants present at the same time.

classical statistics Most commonly associated with classical hypothesis testing in which a null (no-relationship or no-differences) hypothesis is set up in order to be rejected in favor of an alternative hypothesis. When we reject the null hypothesis, we say we have a statistically significant result. Bayesian statistics presents an alternative to classical statistics.

classification Group of supervised learning methods designed to predict the value of a categorical variable or class. Distinct from *regression*.

client A computer through which users interact with the network or Internet. Web clients communicate with web servers. “Client” is also used to refer to the person or organization for which research is conducted, the user of research and information services.

client-server application A computer application implemented with client and server computers communicating over the Internet.

closeness centrality A measure of node importance that characterizes how close a node is to all other nodes in the network, where closeness is the number of hops or links between nodes.

cluster analysis Multivariate statistical method for identifying groups in data. One primary use in business is in market segmentation.

comma-delimited text (csv) File format and extension commonly used as input to spreadsheet programs and systems for statistical analysis.

compile cycle Phrase used to describe the program development process with languages like C/C++/C and Java. We program, then compile, then execute the compiled code. This is in contrast with programming languages that are interpreted and have no formal compilation step.

conjoint analysis Better described as conjoint measurement. Statistical methods applied to the analysis of conjoint surveys, in an effort to measure consumer attitudes about a number of product attributes simultaneously.

content analysis The analysis of the meaning of text from qualitative research. Sometimes called thematic, semantic, and network text analysis, content analysis often starts with simple word counts. Also see text measures.

cookie Data stored on a research user's workstation that may be used to identify the participant and to keep information about past transactions.

corpus A document collection (often a collection of transcripts) used in text analysis and text mining. The plural is "corpora."

crawler (web crawler) A computer program for automated data acquisition from the web. Also called a spider, bot, or robot.

cross-sectional data Units of study are organized in ways that assume independence from one unit to the next. Data items are not adjacent in time or geography, so they are not dependent or related to one another.

cross-validation Resampling technique designed to test predictive modeling techniques. A sample is split repeatedly into training and test subsamples. Each predictive model is fit to the training subsample and evaluated on the test subsample. Multi-fold cross validation splits the sample into K subsamples, and on each iteration, one of the K subsamples is used for testing and the other subsamples are used for fitting the model. Results for the K subsamples are averaged to provide an estimate of out-of-sample predictive accuracy. Leave-one-out cross-validation sets K equal to the size of the sample, so that only one sample observation is used as the test subsample on each iteration. Cross-validation is sometimes referred to as *internal cross-validation* because it is internal to the sample.

data mining Machine learning with large data sets. See [machine learning](#).

data visualization General term for statistical graphics and information design graphics. Used for discovery (exploratory data analysis), diagnostics (model development and checking), and display (presentation, visual communication).

declarative language A computer program in which the statements provide an assertion about a desired result, rather

than a sequence of steps for achieving a result. Examples are Prolog and SQL.

degree (of a network) Each node in an undirected network has a degree or number of links to other nodes. An index of node importance.

degree centrality For a node, another term for degree. The average degree across all nodes in a network. A summary statistic for networks.

degree distribution The distribution or set of degree values for all nodes in a network.

density (of a network) The proportion of actual links out of the set of all possible links. A completely unconnected or disconnected set of nodes has zero links, while a completely connected network of n nodes or clique has $\frac{n(n-1)}{2}$ links. If l is the number of links in a network, then network density is given by the formula $\frac{l}{n(n-1)/2}$.

descriptive statistics Traditional statistical methods that are not intended to draw inferences from samples to populations. Statistical summaries of data that are neither explanatory nor predictive. Distinct from methods described as *inferential statistics*. Also called *descriptives*.

Document Object Model (DOM) Rules that define the proper arrangement or hierarchy of nodes (tree structure) for HTML on web pages.

eigenvector centrality For a node, we consider a node as important if it is close to other nodes of importance, which are in turn close to other nodes of importance, and so on. Much as the first principle component characterizes common variability in a set of variables, eigenvector

centrality characterizes the degree to which a node is central to the set of nodes comprising the network.

Elasticsearch Open-source solution for document indexing, storage, search, and selection. Like Solr, it builds on Apache Lucene and Java-based algorithms for text analytics.
Developed for distributed document stores.

e-mail Store-and-forward method for sending messages over the Internet. Sometimes used for interviewing and for surveys.

emoticon Use of characters on a keyboard to indicate emotions in online communication. Capital letters can be used for emphasis or to indicate a raised voice. Some of the more popular examples with special characters include :-)

for a smile or happy response, :-)

for a wink or “just kidding,” and :-()

for an unhappy or disapproving response. Also called “emoji.”

ethnography Observational or field study of social and consumer behavior in real life settings.

expected value A weighted average of values, with probabilities defining the weights. For observations, the expected value is the arithmetic mean.

experimental research Research setting in which the environment or aspects of the environment can be manipulated by the researcher. Distinct versions of the environment represent experimental treatments. There is randomization through random assignment of subjects to treatments. As much as possible, there is control of factors that can have an effect on subject response. This type of research setting is contrasted with an *observational research*

setting, in which no aspect of the environment can be manipulated by the researcher.

explanatory model A model relating one or more explanatory variables to one or more response variables. Understanding the underlying mechanism relating explanatory and response variables is an important aspect of the explanatory model, in contrast with a model that is purely predictive.

explanatory variable Variable used to predict or explain another variable (the response variable).

factor analysis Multivariate statistical procedure based on linear algebra. Useful in indentifying underlying dimensions in data. Builds on the foundation of principal components analysis. Distinct from principal components in that it focuses on common variance, not total variance. May be used to reduce the number of explanatory variables needed for a predictive model.

frame An element of web design. Some web pages are constructed using frames, blocks of a page that a user can navigate within while all other frames remain static. For example, some online facilities utilize frames, with one frame for the chat stream, one for the text entry area, and another for stimuli to be displayed.

ftp File transfer protocol, one of a number of networking standards used over the Internet.

functional language Computer language that treats programming as an evaluation of mathematical functions, mapping sets of input values into sets of output values. A declarative programming paradigm. Examples are Scheme, Lisp, and Erlang. Programming in a functional style can be

done within languages that are described as procedural or object-oriented.

game theory Specialization within mathematics and economics dealing with players, their knowledge, objectives, and strategies for achieving objectives. There are both competitive and cooperative games. As it is typically used, the term has no special reference to sports.

genetic algorithm Heuristic method guided by gene and mutation ideas from biology and population genetics. Used to solve problems that are not easily solved by traditional optimization or constrained optimization methods.

generalized linear model Class of predictive models (supervised learning models) from traditional statistics. These include linear regression, logistic regression, and Poisson regression, among other methods. The right-hand side of the prediction formula is a linear combination of explanatory variables.

generative grammar Linguistic term referring to the rules that we use to form meaningful utterances. A general term for morphology, syntax, and semantics.

Go (Golang) Open-source computer language originally developed at Google. Like C, C++, and C#, Go requires a compile cycle. Employs a concurrency model designed for fast execution in a distributed processing environment.

graph theory Branch of mathematics introduced by Euler in the eighteenth century. Deals with mathematical structures that represent objects (nodes, vertices) and their relations (links, edges). Mathematical foundation of network science or social network analysis.

grounded theory A methodology for analyzing and interpreting qualitative data. Theory is derived from the data. The analysis involves identifying categories and coding data by categories. Common themes and concepts emerge from observed relationships among categories.

Hadoop An open-source distributed file system on which database systems can be built. Not itself a database system. Often associated with discussions of *big data*.

heuristic Rule in complex modeling situations. Useful in optimization problems that cannot be easily addressed by mathematical programming.

hierarchical modeling Also called multi-level modeling. Traditional statistical modeling employing levels of parameters that must be estimated. Most commonly associated with Bayesian statistics.

HTML Hypertext markup language. Standard codes or tags in text files used to format web pages. The implementation of the *DOM*. Current version is HTML5. See cascading style sheet (CSS).

HTTP Hypertext transfer protocol. The dominant method for client-server communication over the web.

indexing Process of coding text documents or database records so they may be easily located in the future.

inferential statistics Traditional statistical methods that draw inferences from samples to populations. The fitted models may be explanatory or predictive. Distinct from methods described as descriptive statistics.

information retrieval (search) A fundamental unsupervised application of text analytics. Text documents are indexed,

making them searchable. A user searching for information enters a text query, which is matched up with the index for the text documents.

integer programming Methods of constrained optimization in which decision variables take integer values only. A special case is integer programming with binary decision variables.

integrated development environment (IDE) An integrated development environment fosters efficient code development by providing a graphical user interface, a programming-language-aware editor, debugging utilities, and a convenient mechanism for compiling and executing code.

Internet A public data communications network consisting of many interconnected networks worldwide.

Internet of Things (IoT) A network of physical objects with embedded electronics for communicating over a network.

intranet A data communications network internal to an organization.

IRC Internet Relay Chat, a form of chat room software.

ISP Internet Services Provider. An organization that provides connection services to the Internet. These organizations usually provide software applications that run over the Internet, such as browsers and search engines.

IT Abbreviation for “information technology.”

Java A general-purpose programming language introduced into the public domain by Sun Microsystems, with updates and support from Oracle. Especially useful in the development of online applications. Java programs require compilation prior to execution (a compile cycle).

JavaScript An object-oriented scripting language for the web.

Preeminent language for client-side event handling.

Embedded in web browsers. Implemented as a stand-alone language in Node.js, which can be used on the server side.

JavaScript and Java are distinct languages.

JavaScript Object Notation (JSON) JavaScript data structure

with unordered name/value pairs. Provides storage for arrays, strings, numbers, logical (boolean true/false), and the special value *null*. May be utilized directly in JavaScript code used in web pages.

JPEG Graphical bit-mapped file format for images.

knapsack problem A special case of integer programming.

Gets its name from a knapsack or backpack that can hold only a certain total weight or volume of items. Each item to be placed in the knapsack has a certain weight or volume as well as a value to the person who is stuffing the knapsack. The objective is to maximize the value while satisfying the weight or volume constraint. The *standard knapsack problem* has binary decision variables (one item of each type which may or may not be placed in the knapsack) and one constraint, such as weight or volume capacity for the knapsack, but not both. The knapsack problem may be generalized to the case for which multiple items of each type may be placed inside, yielding an *integer knapsack problem*. It may also be generalized to the case of multiple constraints, such as restricting both the weight and a volume of the knapsack. With multiple constraints, we have the *multidimensional knapsack problem (MKP)*.

kbps Kilobits per second. Measure of speed across a communications link.

keyword One word or a group of words used in search queries and indexed by search engine providers.

LAMP A public-domain suite for the development of online applications, this refers to the Linux operating system, Apache web server, MySQL database, and Perl, PHP, or Python programming languages.

levels of measurement Dating back to the work of S. S. Stevens (1946), the distinction among nominal (categorical), ordinal (ranks), interval, and ratio measures.

linear regression Also called ordinary least-squared regression or just regression. A form of generalized linear model using a linear combination of explanatory variables to predict a continuous response variable or variable having meaningful magnitude.

linear programming Methods of constrained optimization in which decision variables take continuous values only.

listserv An automated or moderated e-mail group (MEG). E-mail addresses of all participants (respondents and moderator) are programmed into a listserv. Moderator initiates the thread of discussion. Any response to that message automatically goes to anybody on the listserv.

logistic regression A form of generalized linear model using a linear combination of explanatory variables to predict a discrete binary response variable.

longitudinal data Multiple observations for person or units of study across time periods, but also organized by cross-sections. Methods of analysis are called multiple time series, panel, or longitudinal data analysis. Adjacent units in time can be expected to be more highly related to one

another than units more distant in time.

Lucene Open-source algorithms for information retrieval (search). Foundation for Solr and Elasticsearch search applications.

machine learning Algorithms for analyzing data that use the data themselves to specify the form of the model. Machine learning methods are data-adaptive methods rather than traditional (classical or Bayesian) statistical methods, in which model specification and distributional assumptions are made in advance of analyzing the data. Common machine learning methods include tree-structured modeling, support vector machines, neural networks, and nave Bayes models. Sometimes called *statistical learning* and *data mining*.

Markov chain A stochastic process, finite/discrete or continuous, having the Markov property that the future is independent of the past. This is easiest to illustrate with a finite Markov chain. There are distinct states, and for any two states i and j , the probability of going from i to j is P_{ij} regardless of the previous states of the process. The P_{ij} values are called transition probabilities.

mathematical programming Methods of constrained optimization. General term for linear programming, quadratic programming, and integer programming.

measurement The assignment of numbers to attributes according to rules.

mixed integer programming Methods of constrained optimization in which decision variables take integer or continuous values.

model Representation of the world in a way that can be captured by computer code. Examples are stochastic (probability) and statistical models, mathematical programming, and simulation models.

modem Modulator demodulator. Communications equipment that encodes and decodes data for transmission over a communications network.

MongoDB Open-source document-oriented database system, often used for storing text data from the World Wide Web, social media, and information system processing logs. MongoDB is an example of a NoSQL database system.

morphology A branch of linguistics. Rules for forming complex words. Also see generative grammar.

multi-level categorical variable Categorical variable that takes more than two values, as opposed to a binary variable. Also called a multinomial variable.

multiple imputation The preferred method of handling missing data is multiple imputation. That is, we create a number of alternative data sets using existing data to predict what the missing data values may be, and then we analyze each of the resulting data sets.

MySQL A popular open-source, relational database system. Now distributed by Oracle Corporation.

näive Bayes model Machine learning model for classification that builds on the repeated application of Bayes theorem. It is called näive because it assumes that individual explanatory variables act independently in determining the response.

natural language processing (NLP) Using grammatical rules

to mimic human communication and convert natural language into structured text for further analysis. Natural language refers to the words and the rules that we use to form meaningful utterances.

netiquette Social behavior appropriate for working on the Internet. For example, typing USING ALL CAPITAL LETTERS is considered poor netiquette, as it is considered to be the online equivalent of shouting.

network Interconnected objects—nodes and the links connecting them. A data communications network contains computer system nodes and data communication links. The links are electronic circuits, wired or wireless. Following the mathematics of graph theory, nodes are also called vertices and links are called edges.

network science Interdisciplinary study of networks, drawing on the mathematics of graph theory, physics, and the social sciences.

neural network Biologically inspired machine learning method that implicitly represents interactions among explanatory variables and non-linear relationships among explanatory variables and the response. Connections among explanatory variables are set up in a network structure leading to ultimate connections to the response. May be used for classification or regression.

NoSQL A database system that is distinct from a relational (SQL) system. Common NoSQL databases include key-value pairs, document, column-based, and graph-based systems.

object-oriented language Computer language that builds on

classes of objects, which provide a structure for embedding both data and methods for processing data. Examples are Java, Ruby, Python, and R.

observational research Distinct from *experimental research*.

The researcher is unable to control the environment or manipulate environmental conditions. Rather, the researcher observes events as they occur in the environment, making measures as things happen. Most business research is observational research.

offline Any methodology that does not make use of the Internet or World Wide Web, such as in-person focus groups or telephone focus groups.

online Any methodology that makes use of the Internet or World Wide Web, such as real-time chats, bulletin boards (message boards), and listservs (MEGs).

online community An aggregation of individuals that emerges through communication over the Internet. That is, a group of individuals that interacts online over an extended period of time. Also called virtual community. When used for primary research, an online community may consist of a group of fifty to two hundred people who have been recruited to participate in research about related topics over a period of three to twelve months. The research may incorporate a variety of methods, including online surveys, real-time focus groups, and bulletin boards.

operations research General term for technologies such as mathematical programming, queuing theory, and simulation methods.

organic search Natural search in which the search engine

finds the most relevant matches to the user's query. By far the most important component of relevance is the links a web page receives from other web pages. Organic search is distinct from paid search, in which search results are advertisements and links paid for by advertisers.

outlier Observed value of a variable that is outside the range of most other observed values.

PageRank Algorithm for computing the importance, credibility, or relevance of links in a web search. Analogous to eigenvector centrality.

paid search Search results that are advertisements and links paid for by advertisers. Distinct from organic or natural search results.

panel A group of respondents that has been recruited to participate in many studies over a period of time.

panel data Another term for longitudinal data. Not necessarily related to a panel as a group of respondents.

parameter Quantitative characteristic of a population distribution.

parser (text parser) A computer program that prepares text for subsequent analysis, replacing one character string sequence with another. Often implemented with regular expressions.

Perl "Practical Extraction and Report Language." Text processing and scripting language used extensively in web applications.

PHP Personal home page. A scripting language alternative to Perl, used extensively in web applications.

Poisson distribution Probability distribution for a discrete

variable taking non-negative integer values. Useful in modeling counts, including runs scored and points scored in sports. Gets its name from the French mathematician Siméon Denis Poisson.

Poisson regression A form of generalized linear model using a linear combination of explanatory variables to predict a discrete response taking non-negative integer values (counts). Like the Poisson distribution, it gets its name from the French mathematician Siméon Denis Poisson.

population Complete set of observations of interest to the researcher. Samples are selected from a *sampling frame* identifying members of the population.

population distribution The distribution of interest to the researcher. Characterized by population parameters.

post (noun) Posting made by a participant (moderator, respondent, or observer) in a blog. A statement or question made by a participant in a real-time online group. (verb) To send a message.

posterior distribution Probability distribution of a parameter. In Bayesian statistics, the posterior distribution is obtained by using Bayes theorem along with a prior distribution and the likelihood function for the sample data. It is “posterior” in the sense that it is the end product of a Bayesian data analysis, whereas the prior distribution begins the process. The Bayesian statistician uses Bayes theorem to revise his beliefs about population parameters, beginning with a prior distribution and ending with a posterior distribution. A “belief” in this context refers to a mathematical representation of the population or the unknown.

PostgreSQL A popular open-source, relational database system, written as “PostgreSQL” and often referred to simply as “Postgres.”

predictive model A model designed to predict as yet unobserved values of a response variable using observed values of explanatory variables. The objective is for the model to predict the response accurately, whether or not the underlying mechanism relating explanatory and response variables is understood.

principal component analysis Multivariate statistical procedure based on linear algebra. Useful in identifying underlying dimensions in data. Foundation for factor analysis. May be used to reduce the number of explanatory variables needed for a predictive model.

prior distribution Probability distribution of a parameter at the beginning of a Bayesian data analysis. The prior distribution (also referred to as the “prior”) is in place before any data are collected or analyzed. It represents the researcher’s prior belief about the population parameter.

procedural language Computer language that describes an algorithm or sequence of steps to achieve a certain result. Examples include Basic, Fortran and C. Python and R are often used as though they were procedural languages, ignoring their object-oriented foundations.

psychographics Demographics relating to psychological factors.

Python Open-source, object-oriented, general-purpose computer language. Widely used for text and data processing and web programming, as well as for traditional

statistics and machine learning. Python programs do not require a compile cycle, so they are more quickly developed than programs in C/C++/C or Java.

queueing theory Q Mathematical models for understanding queueing or wait-time processes. Especially useful in the analysis of scheduling and traffic problems.

R Open-source, object-oriented language for programming with data. Widely used for traditional statistics, machine learning, and data visualization. R programs do not require a compile cycle, so they are more quickly developed than programs in C/C++/C or Java.

random forests Machine learning method that uses a community or ensemble of tree-structured models. The method of random forests is an exemplary ensemble method. When building a random forest, we combine the predictions of thousands of tree-structured models. Prior to building each tree, we sample across explanatory variables without replacement, and we sample across observations with replacement (bootstrap sampling). After thousands of trees have been constructed, we average their predictions. See *bootstrap sampling*.

random network (random graph) A set of nodes connected by links in a purely random fashion.

random variable Real-valued quantities of interest. Continuous random variables can take any value along a segment of the real number line. Discrete random variables take integer values along the real number line.,

regression Group of supervised learning methods designed to predict the value of a variable with meaningful magnitude.

Distinct from *classification*.

regular expressions Specialized syntax for identifying character strings, as needed for efficient search and text processing. Implemented in operating system shell search tools and various host languages, including Perl, Java, JavaScript, Python, and R.

relational database Database composed of tables with row-and-column structure. Rows are often referred to as records, columns as fields.

reliability (of a measure) The degree to which repeated measures of the same trait agree with one another, as in test-retest reliability or split-half reliability. For multi-item scales, reliability is reflected in indices of internal consistency. These measures must be taken at about the same time. Reliability is not to be confused with stability, which refers to the degree to which measures relate to one another over time.

resampling Statistical procedure that involves repeated sampling from a sample. See *bootstrap sampling* and *cross-validation*.

sampling Process by which a subset of the population (rather than the entire population) is used for analysis and modeling.

sampling distribution Statistical term for the distribution of a sample statistic across repeated samples from the population. Foundation for drawing inferences about the population using statistics from the sample.

sampling frame List of members of the population from which a sample is drawn.

Scala Open-source, object-oriented, general-purpose computer language that can utilize Java JVM libraries. Foundation language for Apache Spark.

scatter plot Statistical graphic for displaying the values of two continuous variables. Useful to examining relationships between variables.

scraper (web scraper) A computer program for identifying specific portions of data from the web. Utilized in the context of automated data acquisition from the web (web crawling) and often implemented with XPath syntax.

semantics A branch of linguistics. The study of meaning expressed through language. Also see generative grammar.

semantic web An idea for having web-based data defined and linked in a way that can be used by computers as well as by people. This requires wide acceptance of information coding standards such as XML.

semi-supervised learning Combination of supervised and unsupervised learning. Some observations have known response values, others do not. Common situation in text analytics.

sensitivity analysis Within mathematical programming, sensitivity analysis explores the degree to which the optimal solution is stable to variation in input data defining the parameters of the model. A method, model, or process is “sensitive” when small changes in input result in large changes in output.

simulated annealing Heuristic modeling method inspired by the annealing process in metallurgy. Useful in optimization problems that cannot be easily addressed by mathematical

programming.

simulation Capturing the essence of things in a computer program, so the computer program mimics or simulates behavior. Variations include discrete event simulation, process simulation, and agent-based modeling.

social network analysis Term used by social scientists when referring to applications of network science to the study of interactions between individuals, groups, or organizations.

Solr Open-source solution for document indexing, storage, search, and selection. Like Elasticsearch, it builds on Apache Lucene and Java-based algorithms for text analytics.

Spark Real-time analytics environment that builds on Scala. Part of the Apache Software Foundation.

spatial data Observations have geocodes or location data affixed. Common geocodes are longitude and latitude. Methods of analysis are called spatial data analysis. Adjacent units in space are expected to be more highly related to one another than units more distant in space.

SQL Structured query language. A declarative syntax for describing a desired result and getting it back from a database. Associated with relational database systems.

stemming (word stemming) In text analytics, stripping affixes or reducing a word to its base or stem.

stochastic process A collection of random variables, either discrete or continuous.

stochastic programming Mathematical programming with uncertain input data. Rather than specify parameters as fixed input data, we use probability distributions. More easily

implemented with algebraic modeling systems.

supervised learning In traditional statistics or machine learning, a model that recognizes the distinction between explanatory and response variables, with the explanatory variables being used to predict the response variables. Major types of supervised learning include classification and regression.

support vector machine Machine learning method for classification that utilizes decision boundaries defined across sets of explanatory variables dividing the sample observations into groups. These decision boundaries are known as support vectors.

syntax A branch of linguistics. Rules for forming phrases and sentences. Also see generative grammar.

system logging Computer systems keep records of what is happening. These records themselves can serve the purposes of consumer and business research. NoSQL databases are frequently employed for this purpose.

TCP/IP Transmission Control Protocol/Internet Protocol. Basic protocol for computer-to-computer communications over the Internet.

telnet Terminal-to-computer communication protocol used over the Internet.

text analysis General term for methods of analysis and interpretation of text, including content analysis, grounded theory construction, and text mining.

text measure Scores on attributes that describe text as in sentiment analysis. Text measures can be used to assess personality, consumer preferences, and political opinions,

just as survey instruments can. Text measures begin with unstructured text (documents, transcripts) as their input data, rather than forced-choice questionnaire responses.

text mining The automated or partially automated processing of text. It involves imposing structure upon text and extracting relevant information from text. Like data mining, it is often associated with the analysis of large databases or document collections (corpora).

TF-IDF The term frequency-inverse document frequency measurement model. We note the frequency of distinct terms in each document, relative to the frequency of those terms across the entire document collection. A vector of TF-IDF values represents each document. Utilized in information retrieval (search) and various text analytics methods.

thread A thread of discussion or conversation. The sequence of online messages relating to a particular topic. Questions and statements that lead to other questions and statements. A term more prevalent in asynchronous focus groups (bulletin boards or message boards) than in real-time focus groups.

time series Multiple observations for a person or unit of study across time periods. Methods of analysis are called time series analysis. Adjacent units in time can be expected to be more highly related to one another than units more distant in time.

traditional statistics General term for both classical and Bayesian methods, as opposed to machine learning methods. With traditional statistics, we specify the model in advance of analysis and make assumptions about underlying

population distributions prior to fitting the model to sample data.

transitivity A measure of connectedness of a network. Also known as the average clustering coefficient of a network. Usually computed from the adjacency matrix as the proportion of fully connected triples (triads). A triad or triple is a set of three nodes, and a closed triad is a set of three nodes with links between each pair of nodes.

tree-structured model A model for classification or regression that builds a tree structure in relating explanatory variables to the response. The most important or useful explanatory variable is at the top of the tree. Variables become less important going from the top to the bottom of the tree. Implicitly represents interactions among explanatory variables in predicting the response. See CART.

unsupervised learning In traditional statistics or machine learning, a model that does not make a distinction between explanatory and response variables. Examples are association rules, cluster analysis, and principal components analysis. See machine learning.

URL Uniform resource locator, address for Internet information resources.

Usenet Wide-area network precursor of the Internet.

validity (of a measure) The degree to which a measure measures what it is supposed to measure.

virtual community Another term for online community.
Aggregation of individuals that emerges through communication over the Internet

virtual facility Adjective used to describe online the analogue

of a real, physical research facility. For example, an online focus group software provides a virtual facility, analogous to a real in-person focus group.

virtual private network (VPN) Secure network connection for point-to-point communication over the Internet.

web presence The degree to which an organization, brand, website, or web landing page is getting recognized on the web. What is often called “search engine optimization” (SEO) is one part of web presence. One way to assess web presence is to see where a site falls on the list of organic search results.

Wiki An exercise in collective expression on the web.

Software that facilitates the process of generating a website that is editable by a group of contributors. Wiki web pages may be edited by anyone at any time, from any device with web browser access. Term comes from Hawaiian word “wiki,” meaning “quick.”

World Wide Web (web) Hypertext-linked data available over the Internet. These data are, for the most part, unstructured or semi-structured text that must be scraped and parsed prior to input to predictive models. WWW is an abbreviation for the World Wide Web.

XML Extensible markup language. Similar in structure and appearance to HTML, XML has a distinct purpose. It is a way of marking up data that describes the nature of the data.

XPath A specialized syntax for navigating across the nodes and attributes of the Document Object Model (DOM) and extracting relevant data.

Baseball Glossary

ahead in the count From the batter's point of view, a situation in which he has more balls than strikes. Opposite of *behind in the count*.

All Star Game In Major League Baseball, a game played toward the middle of the baseball season with players selected from various teams for their exceptional skills.

American League In Major League Baseball, one of two leagues composed of fifteen teams, five in each of three divisions. The American League allows the designated hitter. See *National League*.

around the horn Baseball slang for throwing the ball around the infield after completing a play with no one on base.

at bats (AB) Batting appearance in which the batter does not receive a base-on-balls (walk), is not hit by a pitch, and does not sacrifice (by bunt or sacrifice fly). The total at bats is used as the denominator in computing a player's batting average. A subset of plate appearances.

away team The team that is playing a game away from its home stadium. Also called *visiting team*.

bailing out Baseball slang for a batter who's foot closest to the pitcher moves away from home plate when swinging at the ball. May happen to a batter regardless of his batting stance (closed or open). Also, known as *stepping in the bucket*.

balk While pitching from the stretch with a runner on base, the pitcher must come to a complete stop before throwing to a position player covering the on-base runner. A balk is

failing to come to a complete stop, and the runner is awarded the next base as a penalty to the team on the field.

Developed in conjunction with Martin (2016), this is a guide to terms used by baseball players, managers, analysts, and fans. It provides an introduction to the language of the game as it has evolved over more than two hundred years of play in the United States. Official definitions of terms are provided by Major League Baseball (2015a). Hample (2007) reviews baseball slang. Ripken and Ripken (2004, 2007) explain baseball fundamentals for players and coaches. Additional useful resources are the [MLB.com](#) website and the Wikipedia (2015) *Glossary of Baseball*.

ball A pitch outside the strike zone and is not hit by the batter.

Any ball that touches the ground prior to reaching home plate.

ban on women Between 1952 and 1992, there was a ban on women in Major League Baseball.

base One of four points that a runner must touch to score a run: first base, second base, third base, and home base (home plate).

base on balls (BB) The batter is awarded first base without hitting the ball because he receives four balls before getting three strikes. Also called a *walk* because the batter typically walks rather than runs to first base.

base runner Offensive player on base (first, second, or third base).

bases loaded Baseball slang for having runners on first, second, and third base.

base coach Team member in uniform designated by the manager to help with advising runners on the base paths. The first base coach advises runners on first about running to second base. The third base coach advises runners about running to home base.

base hit A hit in which the batter reaches as far as first base.

See single (1B).

baserunner Offensive player who has reached one of the bases safely and remains on-base until the inning is over.

baserunning error An offensive player runs outside the base path or fails to touch a base. Counts as an out for the batting team, but is not recorded as an error in the box score for the team. Recorded errors are for the fielding team only.

batter Initial role of offensive player. Uses bat to hit the ball thrown by the pitcher. Stands in the batter's box. Also called hitter.

batter in the hole Offensive player in the dugout set to be the batter after the batter on deck.

batter on deck Offensive player due next in the batter's box. That is, the player set to follow the batter currently at bat. There is a special area called the *next batter's box* where the batter on deck is supposed to stand or kneel.

batter's box Area in which batter must stand while at bat.

battery The pitcher and catcher as a unit.

batting average (BA, AVG) Hits divided by at bats (with walks and sacrifices not counted in at bats):

$$BA = \frac{Hits}{At\ Bats}$$

A batting average above 0.300 is highly valued. Batting

below 0.200 (sometimes called the *Mendoza Line*) is tolerated only for pitchers and exceptionally good catchers and infielders. The average regular-season BA across the thirty MLB teams in the 2014 season was 0.251 ([Sports Reference LLC 2015a](#)). A rule of thumb in selecting batters is to look for BA values above 0.250—that is, we look for batters who get a hit about one in every four at bats. Hitters batting above 0.300 are rare.

batting stance A batter’s position in the batter’s box relative to home plate prior to the pitcher’s throwing the ball. Also see [*closed batting stance*](#) and [*open batting stance*](#).

batting (team) The team on offense, as opposed to the team on defense (on the field).

behind in the count From the batter’s point of view, a situation in which he has more strikes than balls. Also called *down in the count* or *in the hole*. Opposite of *ahead in the count*.

bench Seating area for a team. Also called the *dugout*. Players not in the line-up are referred to as *sitting on the bench*. A team with talented players not in the line-up is said to have *bench strength* or a *deep bench*. Idiomatic expression: when an organization has many talented employees, it is said to have bench strength.

big leagues Major League Baseball. Affectionately called “the bigs.”

BIP (BPIP) Ball in play or ball put into play. When counting balls and strikes for a pitcher, balls put into play by hitters are strikes.

bloop single Term used for a ball that falls for a hit between

the infielders and the outfielders. Also called a “Texas Leaguer.”

bunt Batting strategy that involves holding the bat in a fixed position, rather than swinging the bat. With two strikes, a bunted foul ball results in a third strike.

call As a verb, the umpire calls plays (calls a pitch a ball or strike, calls a hit ball fair or foul, and calls a runner safe or out). As a noun, the umpire makes calls.

called game A game canceled or postponed by the umpire. Prior to lighted stadiums, a game might have been called due to darkness. Currently, games are commonly called due to rain.

catcher Defensive player who crouches behind home plate and catches balls thrown by the pitcher. The only player on the fielding team positioned in foul territory. Position number 2.

caught looking The batter is called out on strikes without swinging the bat.

caught off base A base runner is permitted to step off base but may be tagged out by a defensive player when off base.

caught stealing (CS) On an attempted steal, a base runner is tagged out prior to reaching the next base.

center fielder Defensive player who plays in the middle of the outfield, between the left fielder and right fielder. Position number 8.

Championship Series At the end of the regular season, following the Wild Card Game, there are two Division Series playoffs in each league. The winners of these Division Series play each other in a best-of-seven series known as a Championship Series, one in the American

League and one in the National League.

changeup An off-speed pitch (MLB speeds 70–85 mph).

Speed, rather than movement, is what distinguishes this pitch from a fastball. Comes in various forms, including the circle changeup, three-finger changeup, and four-finger changeup (also known as a *palm ball*).

check swing A batter begins to swing at a pitched ball, but not far enough to be called a strike. Whether a batter's swing is a check swing or a strike swinging is a matter of judgment. The call can be made by the home plate umpire, with possible appeal to the first base umpire for a right-handed batter or to the third base umpire for a left-handed batter.

choking up Holding the bat with hands away from the bottom (bat handle end) of the bat. A technique used by singles hitters who take shorter swings and have more control of the bat.

chop single Base hit in which the batter intentionally hits the ball down toward the ground so that it bounces high and is hard to catch in time to throw the batter out at first base.

clean-up hitter The fourth hitter in the line-up, selected because he has a good chance to get a hit with other runners on base.

closed batting stance A batter's position in the batter's box in which the foot closest to the pitcher is closer to home plate than the foot farthest from the pitcher.

closer Relief pitcher who ends a game.

clutch hitter A hitter who does well with runners on base and/or when the game is coming to an end and his team needs runs to avoid a loss.

coach Team member in uniform designated by the manager to help with coaching duties such as acting as a base coach.

command A pitcher with command can throw strikes when he needs to throw strikes. Also see *control*.

control A pitcher with control can throw strikes much more often than balls. Also see *command*.

cover the bases A player on the fielding team covers a base by standing close to the base. By being close to the base the fielder is ready to execute a tag out or a force out. Having a defensive player on every base is covering the bases.

Idiomatic expression: when we prepare for every possibility, we cover the bases. When referring to a single base in baseball, it is common to use the phrase *cover the bag*, as a fielder would do for a pick-off play.

crowd the plate Batter stands in the batter's box very close to home plate.

curveball (curve) A breaking ball much slower than a fastball (MLB speed 65–80 mph). With a right-handed pitcher throwing to a right-handed hitter or with a left-handed pitcher throwing to a left-handed hitter, a curve ball breaks away from the hitter. The standard curveball is the overhand curveball, affectionately known as *Uncle Charlie* or *the hook*. Some pitchers throw what is known as a *knuckle curveball*.

cut fastball (cutter) Also called a *cut fastball*. A fastball with movement.

cut-off position A position on the field of play between an outfielder and home plate. With a runner rounding the bases and heading for home, the outfielder can throw a ball

toward home plate using a low trajectory, so an infielder in the cut-off position can intercept the ball and throw to another base, usually second base, to prevent another runner from advancing around the bases. Alternatively, the infielder cutting off the ball from the outfield can decide to throw the ball home enabling the catcher to tag the runner out. Because the infielder is closer to home base than the outfielder, the infielder's throw is likely to be more accurate than the outfielder's. Many outfielders do not have strong throws to home base, so throwing to the player in the cut-off position is essential.

dead ball Ball out of play, resulting in temporary suspension of play.

dead ball era Pre-1920 baseball. Period during which very few balls were used during games. There was no requirement to use a new white ball. Contrasted with *live ball era*.

dead red hitter A batter who can hit only when he gets the type of pitch he is looking for, such as a fastball. But when thrown another pitch, such as a curve, he has trouble hitting.

defense Team on the field, as opposed to the team that is up to bat (offense).

defensive indifference A runner on base attempts to take the next base, and the team on the field allows it. The situation occurs late in a game when the team on the field has a lead of more than one run, so an additional run by the hitting team will not win the game. When the official scorer rules defensive indifference, the runner's act of taking the next base is not recorded as a stolen base.

designated hitter (DH) A player who has a place in the

batting order but does not play a position in the field. This player hits in place of the weakest hitting player who takes the field (usually the pitcher). In Major League Baseball, the designated hitter role may be used in the American League, but not the National League.

diamond Another word for the baseball field of play. Gets its name from the square/diamond shape of the infield defined by the four bases.

dig in Batter uses spikes on shoes to make a hole in the batter's box.

Division Series After the end of the regular season, a best-of-five series of playoff games within a league between winners of the three divisions and/or the Wild Card Game. There are two division series within each league.

double (2B) A hit in which the batter reaches as far as second base. Also called a *two-bagger*.

double-header Two games played the same day. A standard double-header has two games with a short break between them. A *day-night double-header* has a number of hours between the games.

double play Getting two outs on one play. Various double plays are designated by player position numbers. For example, a 4-6-3 double play involves a throw from the second baseman to the shortstop, followed by a throw from the shortstop to the first baseman. An *unassisted double play* is a double play executed by one player alone.

double-switch Player substitution involving two players. Often used to substitute a position player along with a relief pitcher, with the position player (usually a better hitter)

placed in the line-up so that he hits before the pitcher (usually a poorer hitter).

dugout Seating area for a team. Also called the *bench*.

earned run average (ERA) Index of pitching performance.

The number of runs allowed (not due to errors) per nine innings of play. Also see WHIP.

expected runs Computed for any half-inning state or game situation, the number of runs the hitting team can expect to earn through the end of the half-inning. The computation is based on historical play-by-play data for the state or game situation. Expected runs are especially useful in the evaluation of alternative playing strategies, such as whether to use the sacrifice bunt or whether to attempt a stolen base.

extra-base hit Hit greater than a single. That is, a double, triple, or home run.

fair ball Ball hit inside the foul lines.

fair territory Region inside the foul lines.

fan Short for “fanatic” or “fancier.” Another earlier term was “crank.” A consumer of the sport, someone who loves the game and/or a particular team.

fantasy baseball Fans of baseball pretend to manage a Major League Baseball team by assembling a team of players. Enables fans to make decisions as if they were owners of teams. An early term for this activity was *rotisserie baseball*.

fastball Faster than any other type of pitch, has MLB speeds 80–100 mph. Comes in various forms defined by the pitcher’s grip on the ball, including the *four-seam fastball*, *two-seam fastball*, *cut fastball* (also called a *cutter*), and a

split-finger fastball (also called a *splitter*). The four-seam fastball is faster but has less movement than other fastballs.

fielder Player on defense.

fielder's choice Batter arrives at first base because a fielder elects to get a runner out in place of the batter. Batter is not awarded a hit.

fielding error A defensive play that should have resulted in an out, but does not. Due to catching, not throwing. Also see throwing error.

first base The base a batter/hitter must touch first in order to get a hit.

first baseman Defensive player and infielder who plays closest to first base. His position number is 3.

five-tool player A player with strong baseball skills for running, fielding, throwing, hitting, and hitting with power.

fly ball Ball hit in the air, as opposed to a ground ball, but not hit low in the air, as opposed to a line drive.

fly out Ball hit in the air and caught in the air, resulting in an out. May be in fair or foul territory. Also called a *fly ball out*. Different from a *ground out*.

forfeited game Game ended by the umpire-in-chief in favor of a team that is leading by a score of nine to nothing or for violation of the rules of the game.

force out As opposed to a *tag out*, an out recorded when a defensive player with ball in hand or glove touches the base that a runner must advance toward because previous bases are occupied or about to be occupied. Also called a *force play*. Also see *neighborhood play*.

foul ball Ball hit out of play, outside the foul lines.

foul territory Region outside the foul lines.

foul tip A foul ball that comes directly off the batter's bat and into the catcher's glove or hand. It counts as a strike, including a third strike.

frame (a pitch) Catcher's attempt to catch a ball thrown by a pitcher in a way that appears to be within the strike zone even if it does not fall within the strike zone.

free agent Player not under contract to any specific team and eligible to sign with any team.

full count Three balls and two strikes on the batter.

game (G) A baseball game or appearance in a game.

grand slam A home run with the bases loaded.

ground ball Ball hit on the ground, as opposed to a line drive or fly ball.

ground out Batter is thrown out at first base after hitting a ground ball to one of the fielders. As opposed to a *fly out*.

ground-rule double When a hit bounces over a fence, leaving the field of play, the batter is awarded second base (a double).

hit (H) Batter hits the ball and reaches a base safely without the aid of an error, fielder's choice, walk, or hit-by-pitch. Includes singles, doubles, triples, and home runs. Idiomatic expression: a popular item is called a hit.

hit-and-run Offensive strategy intended to give the runner a head-start in moving to the next base. The batter tries to hit the ball regardless of where it is thrown in order to protect the runner.

hit batsman (hit by pitch, HBP) When a pitched ball hits the

batter's body, the batter is awarded first base.

hit for the cycle A hitter gets at least one single, double, triple, and home run in a game. An unusual accomplishment.

hitter Another term for batter.

hitting slump A batter is having trouble hitting for a number of games in a row, either having no hits or having very few hits.

hitting streak A batter is doing exceptionally well at the plate.

Usually assessed by the number of games in a row with at least one hit. Joe DiMaggio holds the record, hitting safely in 56 consecutive games.

hitter's park Parks such as Coors Field in Denver and Fenway Park in Boston are known to be favorable to hitters, yielding higher expected runs across all in-game states. As opposed to a *pitcher's park*.

hitting for power Hitting home runs.

holding runner on base Activity of the first baseman, standing close to first base when a runner is at first base, so that the runner will stay closer to the base (take less of a lead) to avoid being tagged out if the pitcher were to throw the ball to the first baseman. This is done so that the runner is less likely to steal second base.

home plate Flat rubber marker with a square shape facing toward the pitcher and a triangular side facing away from the pitcher. Catchers work in a crouched position behind home plate. When batting, players stand beside home plate in the batter's box. And after running around the bases, they must touch home plate to score a run. Also called *home base*.

home run (HR) A hit in which the batter reaches as far as home plate. Also called a *homer* or the *long ball*. Usually obtained by hitting the ball over the fences. For an *inside-the-part home run*, the ball stays in the field of play while the batter runs around the bases, touching home plate safely. Idiomatic expression: when we do a good job on a project, we say we have hit a home run.

home team The team that is playing a game away in its home stadium. Also see [away team](#) or [visiting team](#).

illegal pitch A pitch delivered by the pitcher without his pivot foot on the pitcher's plate, a quick pitch, or a balk.

infielder Defensive player who plays close to the bases.

inning Unit of play in a baseball game. An inning consists of three outs. A full inning consists of the visiting team batting, followed by the home team batting. The top of the inning is the portion of the inning in which the visiting team is batting. The bottom of the inning is the portion of the inning with the home team batting.

intentional base on balls The pitcher gives a batter a base on balls without trying to throw strikes. Used as baseball strategy to avoid pitching to the hitter (who may be a good hitter) or to put a runner on first base so that force-outs are possible.

interference Offensive interference occurs when the batting team interferes with, obstructs, impedes, or confuses any fielder attempting to make a play. For example, a runner hit by the ball while running interferes with the fielder trying to catch the ball. This is referred to as being *soaked*, and the runner is called out. Defensive interference occurs when the

catcher's glove touches the bat, affecting the batter's swing or preventing him from hitting the ball. There can also be umpire interference and spectator/fan interference.

in the hole From the batter's point of view, a situation in which he has more strikes than balls. Also called *down in the count*.

keystone sack Another term for second base.

knuckleball A very slow pitch (MLB speeds 55–70 mph) with little spin that moves in an unpredictable manner. Also called a *knuckler*.

lead-off hitter The first hitter in the line-up, selected to begin the team's hitting because he has a good chance of getting on base.

leave the yard (go yard) Baseball slang for hitting a home run.

left fielder (LF) Defensive player who plays on the left side of the outfield where "left side" is defined by looking at the field from home plate. Position number 7. Idiomatic expression: crazy ideas come out of left field.

left on base (LOB) Player on base while the team reaches three outs, ending that batting team's half-inning.

lefty Player who bats or throws left-handed.

line drive Ball hit in the air, as opposed to a ground ball, but not hit high in the air, as opposed to a fly ball.

lineup The order of players/hitters for a team submitted to the home plate umpire at the beginning of the game, with changes reported to the umpire during the game.

live ball A ball that is in play, as opposed to a *dead ball*.

live ball era After 1920 there was a requirement to use a new white ball in Major League Baseball. Contrasted with *dead ball era*.

making the turn A runner running to first base is permitted to run beyond the base and toward foul territory. If instead of running toward foul territory, the runner begins on the path to second base, he is referred to as making the turn and may be tagged out by a fielder.

manager Person responsible for a team's actions on the field.
May be one of the players.

manufactured run Run generated by a “small ball” strategy of getting runners on base and advancing them one base at a time by singles, walks, and stolen bases, as opposed to extra-base hits.

Mendoza Line Named after Mario Mendoza, a big league shortstop who's lifetime batting average was .215. The Mendoza Line is a .200 batting average, thought to be a minimally acceptable batting average for a position player. A better term for this line might be the “Uecker Line,” after Bob Uecker, a strong defensive catcher whose career batting average with the Braves in Milwaukee and Atlanta, 1962–1967, was exactly .200.

men on base Batting team's runners on first, second, and/or third base.

middle reliever Pitcher who enters a game after the start of the game and before the end of the game. Distinct from the starting pitcher and the closer.

middle infielder Second baseman or shortstop. See up the middle.

MLB Major League Baseball. Professional baseball organization in the United States. Currently composed of thirty teams in two leagues, fifteen in the American League and fifteen in the National League.

National League In Major League Baseball, one of two leagues composed of fifteen teams, five in each of three divisions. The National League does not allow the designated hitter. See *American League*.

neighborhood play When there is a close play at second base with the runner sliding into the base, a force out may be recorded even if the second baseman does not keep his foot on the base. As long as the second baseman's foot is in the neighborhood of second base, the force out is recorded. This is to avoid injuries to the players who are wearing spikes on their shoes. See *force out*.

no hitter Pitcher pitches nine innings without allowing a hit, although batters may reach base through walks or errors.

Also see *perfect game*.

no-no Baseball slang for a no hitter. Think in terms of no hits and no runs scored by the other team. Also see *perfect game*.

obstruction A fielder impeding the progress of a runner while not in possession of the ball.

offense Team up to bat, as opposed to the team that is in the field (defense).

official scorer Person designated to keep records of the game and make decisions about hits versus fielding errors.

on the field (team) The team on the field is the team playing defense, as opposed to the batting team.

on-base percentage (OBP) Proportion of plate appearances in which a batter reaches base through a hit, walk, or hit by pitch. Does not include reaching base by a fielding error, fielder's choice, dropped/uncaught third strike, fielder's obstruction, or catcher's interference. Also called on-base average (OBA). Usually presented as a proportion, not a percentage. Popularized by the Lewis (2003) book *Moneyball* and the subsequent movie by the same name, OBP is seen as a better index of offensive performance than batting average. Computed as

$$OBP = \frac{(Hits + Walks + Hit by Pitch)}{(At Bats + Walks + Hit by Pitch + Sacrifices)}$$

Getting a walk or being hit by a pitch has the same effect as a base hit for the batter himself—they move the batter to first base without an out being recorded. The average regular-season OBP across the thirty MLB teams in the 2014 season was 0.314 (Sports Reference LLC 2015a). A rule of thumb in selecting good batters is to look for OBP values above 0.333—that is, we look for batters who get on base about one in every three plate appearances.

on-base percentage plus slugging (OPS) Measure of overall hitting prowess.

$$OPS = OBP + SLG$$

The average regular-season OPS across the thirty MLB teams in the 2014 season was 0.700 (Sports Reference LLC 2015a).

open batting stance A batter's position in the batter's box in which the foot closest to the pitcher is further away from home plate than the foot farthest from the pitcher.

out One of three units of play for the batting team. That is, there are three outs in a half-inning.

“out” Umpire call indicating that an out is to be recorded.

outfielder Defensive player who plays far away from the bases.

overslide At second or third base, a runner can, after touching the base, go beyond the base and lose contact with the base. When the player loses contact with the base, he may be tagged out by a fielder. An overslide can occur at second and third base. An overslide is not relevant at first base where running beyond the base and toward foul territory is permitted. An overslide is not relevant at home base, where running beyond home base after scoring a run completes the runner's path around the bases.

pace of play Also called *pace of game*. Major League Baseball has been experimenting with ways to speed up the game, focusing on unnecessary delays by pitchers or batters. This is controversial because the leisurely pace and lack of a time clock are attractions of baseball. The introduction of video replay affects pace of play, making games longer.

passed ball Pitch that gets past the catcher that is the fault of the catcher, as opposed to a wild pitch, which is the fault of the pitcher.

PECOTA A measurement and prediction system that uses player-comparable age curves as its base data. The name refers to Bill Pecota, a Kansas City Royals infielder from the 1980s.

perfect game Pitcher pitches nine innings without allowing a runner to reach base—no hits, no walks, no errors. Twenty-

seven batters come to the plate, and they all make outs.

pick off assignment Designation of the fielder responsible for covering the bag for a pick off play. Relevant to pick offs at second base, where either the second baseman or the shortstop could cover the bag.

pick off play A pitcher throws to first, second, or third base in an attempt to catch a runner off base. The fielder receiving the ball must tag the runner out before the runner touches a base. Most commonly executed with a runner at first base.

pinch hitter Player batting for another player, replacing the original player in the lineup. The original player may not return to the game.

pinch runner Player running for another player who happens to be on base, replacing the original player in the lineup. The original player may not return to the game.

pitch Ball delivered by the pitcher to the batter.

pitch count The number of pitches (balls plus strikes) thrown by a pitcher after his entry into a game. This number is watched carefully by managers. A starting pitcher's pitch count of one hundred is seen as a warning signal by many managers. With higher pitch counts, pitchers are more susceptible to injuries. At higher pitch counts, a pitcher's fastball velocity is reduced. The pitcher is less effective in getting outs. Pitch count also refers to the number of pitches that a batter has faced in a given at-bat. The offensive team may employ a high-pitch-count strategy, attempting to take more pitches (balls and strikes) to drive up the starting pitcher's pitch count, thus forcing that pitcher out of the game. This was one of the strategies popularized by the

Lewis (2003) book *Moneyball* and the subsequent movie by the same name.

pitcher (P) Defensive player responsible for throwing the ball to the batter. Position number 1.

pitcher's duel A game in which two dominant pitchers are pitching, with very few runs being scored.

pitcher's park Parks such as U.S. Cellular Field in Chicago and AT&T Park in San Francisco are known to be favorable to pitchers, yielding lower expected runs across all in-game states. As opposed to a *hitter's park*.

pitcher's plate A rectangular hard rubber platform near the middle of the pitcher's mound. The pitcher's pivot foot must be on the pitcher's plate prior to delivering the ball to the batter. Also called the *rubber*.

pitching from the stretch Instead of using a full windup, the pitcher uses a partial windup, coming to a full stop before throwing. Typically used when there is a runner on base.

pitching depth Number of pitchers a team has available for a game.

pitching mound Raised circular area from which the pitcher pitches.

pitching rotation The ordered set of four or five players that a team uses as its starting pitchers. Across games in the regular season, starting pitchers usually have a fixed sequence of appearance so that each pitcher has three or four days rest between starts.

pivot foot Pitcher's foot that must be in contact with the pitcher's plate when delivering the ball to the batter.

place hitter Hitter who controls the bat so well as to place the

ball where no fielders can get to it, while keeping the ball in the park.

plate Baseball slang verb for scoring a run. Comes from the term “home plate.” So “they plate two” means “they score two runs.”

plate appearance Sum of at bats, walks, hit by pitches and sacrifices.

$$\text{Plate Appearances} = \text{At Bats} + \text{Walks} + \text{Hit by Pitch} + \\ \text{Sacrifices}$$

platooning A managerial strategy in which different lineups are used when facing right-handed versus left-handed pitchers. For example, the lineup may include more right-handed hitters when the starting pitcher of the opposing team is left-handed and more left-handed hitters when the starting pitcher of the opposing team is right-handed. This is done because right-handed hitters are expected to be less effective against right-handed pitchers and left-handed hitters are expected to be less effective against left-handed pitchers. These differences in effectiveness are due to the fact that a curve ball breaks away from a like-handed hitter, making it harder to hit.

“play ball” Umpire-in-chief’s order to begin the game.

pop-up Ball hit in the air that does not travel far. Usually caught by an infielder, catcher, or pitcher.

position number Number associated with field position and used for official scoring: (1) pitcher, (2) catcher, (3) first baseman, (4) second baseman, (5) third baseman, (6) shortstop, (7) left fielder, (8) center fielder, and (9) right fielder.

position player Player other than the pitcher. That is, an infielder, outfielder, or catcher.

power hitter Baseball player who hits home runs.

productive at bat An at bat that advances a runner in some way, including a hit, sacrifice bunt, sacrifice fly, or hitting behind a runner to advance the runner. Also called a *productive out*, with the opposite being an *unproductive out*.

pull hitter A right-handed batter who usually hits the ball between second and third base or to left field. Alternatively, a left-handed batter who usually hits the ball between first and second base or to right field. In response to pull hitters, some fielding teams put on a shift, moving players away from their normal positions.

pull the string Baseball slang for fooling a batter with a pitch, usually a curveball.

quick pitch A pitch delivered by the pitcher without giving the batter sufficient time to get situated in the batter's box. Also called a *quick return*.

reaching for the fences Baseball player who tries to hit home runs.

regulation game Game that counts as an official completed game. Some games are canceled. Others are postponed without being complete and must be completed at a later time or date.

relief pitcher Pitcher who enters the game after the starting pitcher has been removed from the game.

replay review A subset of umpire calls is subject to video replay review by umpires located in a remote location. These *reviewable calls* include potential home run calls,

non-home-run boundary calls, specified fair/foul ball calls, force out/tag out calls, catches in the outfield, hit-by-pitch calls, collisions at home plate, tag-up calls, and various force-out/tag-out calls. Balls and strikes called by the home plate umpire are not reviewable.

retouch Runner returning to base after passing a base. Runner must be touching the base to avoid being tagged out by a fielder.

reverse curve A breaking ball that moves in the opposite direction from a curveball. Also called a *screwball* or *fall-away*.

right fielder (RF) Defensive player who plays on the right side of the outfield where “right side” is defined by looking at the field from home plate. Position number 9.

RISP Symbol for a runner in scoring position, which is a runner on second or third base because a single is usually enough to get that runner home.

rounding the bases Baseball slang for running around the bases, from first to second, to third, to home.

run Unit of scoring in baseball. Also called a *score*. Analogous to a point in football or basketball.

run batted in (RBI) When a runner scores as a result of a batter’s hit, sacrifice, or fielder’s choice out, the batter gets an RBI.

run down A runner caught between bases may be tagged out by a fielder. Often two or more fielders are involved in this activity, throwing the ball back and forth in an attempt to tag the runner out.

runner Offensive player no longer at bat, but on the base

paths.

sacrifice bunt A bunt that successfully advances a baserunner.

sacrifice fly A line drive or fly ball that successfully advances a baserunner. The baserunner must tag the current base and advance to the next base without being tagged out.

“safe” Umpire call, awarding a base to a runner.

scoring position A runner on second or third base is referred to as a runner in scoring position because a single is usually enough to get that runner home.

screwball A breaking ball that moves in the opposite direction from a curveball. Also called a *reverse curve* or *fall-away*.

season In Major League Baseball, a baseball season consists of 162 games, 81 away games and 81 home games.

second base Base in the middle of the infield, between first and third bases. Also called the *keystone sack*.

second baseman Defensive player, infielder who plays closest to second base. Position number 4.

secondary lead A baserunner takes his initial lead off the base while the pitcher is in his windup. But after the pitcher throws the ball, the baserunner moves further away from the base. This additional distance from the base is known as a secondary lead.

semi-intentional walk A walk in which the pitcher pitches to the batter with the intention of walking him but without the catcher signaling an intentional walk. The event is recorded as a walk, rather than an intentional walk.

set position Legal pitching position usually used when offensive players are on base. In the set position, the pitcher comes to a complete stop prior to throwing the ball. The

other legal position is the *windup position*. Also see *balk*.

shadow ball Baseball pantomime or playing with an imaginary ball to entertain fans. Popularized in the black baseball teams during the depression.

shift On-the-field players playing out of position to adjust to the particular strengths of the hitter. For a right-handed pull hitter, for example, a shift may involve the second baseman playing between second base and third base, instead of in his normal position between second base and first base. The use of the shift is controversial. Fielders playing away from their normal positions may be less proficient in making double plays, executing pick offs, and taking cut-off positions. The shift also affects the integrity of baseball data. Suppose a third baseman is in a shift, playing between first base and second base with a left-handed pull hitter at bat. When that third baseman fields a ground ball between first and second base, throwing the batter out at first, the play is recorded as a 5-3 put out, implying that the ball was hit toward third base.

shine ball The pitcher rubs the ball clean on one side using his uniform, while applying dirt or powder on the other side. Employed to affect the ball's trajectory.

shutout A complete nine-inning game thrown by one pitcher with no runs scored by the opposing team..

shortstop Defensive player, infielder who, like the third baseman, plays between second base and third base. But the shortstop is further from third base than the third baseman. Position number 6.

side-arm delivery Pitching delivery in between a common

overhand delivery and an underhand delivery (as in softball).

single (1B) A hit in which the batter reaches as far as first base.

Also called a *base hit*.

slider A ball gripped like a two-seam fastball, but slightly off-center. With MLB speeds 70–85 mph, it is not as fast as a fastball but breaks in the same direction as a curveball.

slugging percentage (SLG) Measure of hitting power. Babe Ruth's slugging percentage was 0.690, the MLB all-time record. The maximum value possible is 4.0, implying that a player hits a home run in every at bat. Computed as

$$SLG = \frac{(Singles + (2 \times Doubles) + (3 \times Triples) + (4 \times Home Runs))}{At\ Bats}$$

The average regular-season SLG across the thirty MLB teams in the 2014 season was 0.386 (Sports Reference LLC 2015a).

small ball Baseball strategy based on base hits, walks, and stolen bases. That is, runs are scored by advancing by one base at a time—manufactured runs. Also called an *inside game*, suggesting that the action is inside the base paths.

spin rate The rate of a pitched ball's spin on its way from the pitcher's hand to home plate. Fastballs with the same speed in miles per hour may have very different spin rates. Fastballs with low spin rates (fewer than 2,000 revolutions per minute) are sometimes called “heavy.” They appear to sink. Fastballs with high spin rates (more than 2,500 revolutions per minute) have more movement.

spitball Also called a *spitter*. The pitcher applies saliva or some foreign substance to the ball, which may affect the ball's trajectory. Spitballs are not allowed in Major League

Baseball, although some pitchers have been suspected of throwing spitballs.

squeeze play A batter bunts with a runner on third and fewer than two outs, trying to bring the runner home. With a *suicide squeeze*, the runner on third begins running home as the pitcher begins to throw the ball. With a *safety squeeze*, the runner on third begins running after the batter bunts the ball.

starting pitcher The pitcher who is in the starting line-up for a game, as opposed to a relief pitcher.

steal (stolen base, SB) Advancing from first to second base, second to third, or third to home without the assistance of a hit, passed ball, or wild pitch. The MLB success rate in stolen bases since 1990 is around 70 percent (Baumer and Zimbalist 2014).

stepping in the bucket Baseball slang for a batter whose foot closest to the pitcher moves away from home plate when swinging at the ball. May happen to a batter regardless of his batting stance (closed or open). Also, known as *bailing out*.

strike For a *called strike*, a pitched ball within the strike zone that the batter does not swing at on any strike. Or a ball that the batter hits foul on a first or second strike. For a *strike swinging*, a ball that the batter swings at but does not hit on any strike. Or, on a third strike, a ball that the batter swings at and hits foul directly into the catcher's glove or hand, which is also known as a *foul tip*. Or, on an attempted bunt on the third strike, a bunt that is hit outside the foul lines is a *strike swinging*. For keeping a complete record of balls and strikes, any ball that is put into play (regardless of its

position in the strike zone) is counted as a strike.

strikeout (K) A batter gets three strikes before getting four balls. Can be a *strikeout looking* or *called strike three* with the batter not swinging at the ball or a *strikeout swinging*. The symbol *K* represents a strikeout and may be thought of as coming from the word “knockout” or from the letter “k” in “strikeout.” The *K* is sometimes written backwards to indicate a strikeout looking. If first base is unoccupied by a runner and the catcher fails to catch the third strike, then the batter can be granted first base if he reaches that base before being forced out. The symbol *SO* is used to represent a strikeout or the number of strikeouts by a batter.

strike zone Rectangular region with its base width and depth defined by the area of home plate and its height defined by reference to the batter, extending from the batter’s kneecap to approximately three baseballs above the batter’s belt. Formerly, the upper range of the strike zone was the batter’s shoulders. The effective strike zone varies from one umpire to the next. A smaller or narrow strike zone favors the batter. A larger or wide strike zone favors the pitcher. Baseball analysts measure the size of the strike zone by area in square feet.

suspended game A called game to be completed at a later time/date.

sweep Baseball slang for winning all games in a short series (usually three or four games) with another team. Also known as a “clean sweep.”

switch hitter Batter who can hit either from the left or right side.

switch pitcher Pitcher who can throw with either hand.

Extremely rare player.

tagging up When there are fewer than three outs, a runner may keep his foot on a base, wait until a fly ball is caught, and then run to the next base. Upon tagging up in this way, the runner will be granted the next base if he is not tagged out by a fielder prior to touching the next base.

tag out As opposed to a *force out*, an out recorded for a runner by tagging him with the ball or glove containing the ball before the runner can touch the base. Also called a *tag play*.

take a lead (off base) A runner moves away from the base he is positioned at and in the direction of the next base in order to get a head start in running to or stealing the next base.

See *holding runner on base*.

take a pitcher deep Baseball slang for hitting a home run.

Texas Leaguer Term used for a ball that falls for a hit between the infielders and the outfielders. Also called a “bloop single.” The Texas League is a Double-A minor league baseball league, but there is no evidence that this type of hit is any more prevalent in that league than in any other league.

third base The base after first and second base that a runner must touch on his way to home plate to score a run.

third baseman Defensive player, infielder who plays closest to third base. Position number 5.

three-bagger A hit in which the batter reaches as far as third base. Also called a *triple*.

throw Any act of throwing the ball in play other than pitching to the batter.

throwing error A defensive play that should have resulted in an out, but does not. Due to throwing, not catching. Also see fielding error.

tie game Regulation game called with home and away teams having the same number of runs. Rare event in baseball.

“time” An umpire call, interrupting play. The ball is dead while time is called.

tip a pitch An action by the pitcher that reveals the type of pitch that is being thrown. This could be the pitcher’s position on the pitcher’s plate (rubber), arm angle, or release point. If any of these varies with the type of pitch, an astute batter may be able to make an inference about the type of pitch being thrown. Knowing the type of pitch that is being thrown gives the batter an advantage in hitting.

total bases (TB) The number of bases a player gains as a result of hits. That is, the number of singles, plus two times the number of doubles, three times the number of triples, and four times the number of home runs.

triple (3B) A hit in which the batter reaches as far as third base. Also called a *three-bagger*.

triple crown Batter across the regular season who leads the league in batting average, home runs, and runs batted in.

triple-play Getting three outs on one play. Various triple plays are designated by player position numbers. For example, a 5-4-3 triple-play involves a throw from the third baseman to the second baseman, followed by a throw from the second baseman to the first baseman. An *unassisted triple-play* is a triple-play performed by one player alone.

two-bagger A hit in which the batter reaches as far as second

base. Also called a *double* (*2B*).

umpire Person enforcing the official rules of Major League Baseball, calling safe or out on the base paths, fair or foul balls, and balls or strikes at home base. Four umpires are set for regular season games, associated with each of the bases and with the umpire at home base calling balls and strikes. In playoff games two additional umpires are added, one for the right field line and one for the left field line.

umpire-in-chief Lead umpire of the umpiring crew.

up the middle Sound defense up the middle refers to the play of the middle infielders (second baseman and shortstop) and the center fielder. Many teams put their best fielders in these positions because so many balls are hit toward the middle of the field (up the middle).

up to bat (team) The batting team or team on offense, as opposed to the fielding team.

visiting team The team that is playing a game away from its home stadium. Also called *away team*.

VORP A player's value over a replacement player of average ability at the same position. Expressed in units of runs per game. A summary performance measure similar to *WAR*.

walk Batter awarded first base because four balls are called before three strikes. Also called a *base on balls* (*BB*).

walk-off balk A pitcher's balk that brings a runner on third home to win a game in the bottom of the ninth inning. A very rare event. Could also be called a *balk-off win*.

walk-off hit A hit that scores a run for the home team in the bottom of the ninth inning, winning the game.

walk-off home run A home run for the home team in the

bottom of the ninth inning, winning the game.

WAR (WARP) Wins above replacement (player) across an entire season. A player's value over a replacement player of average ability at the same position. A WAR value of 5, say, means that the team will win five fewer runs across the entire season if it has to replace the player. Expressed in units of wins across the season, with ten runs being equivalent to one win. Various proprietary versions of this measure exist, one variation being wins above replacement player (WARP). WAR is a summary performance measure similar to *VORP*. The measure *openWAR* represents an open-source measure of WAR, which permits calculation from public-domain data sources.

WHIP Walks and hits allowed per inning pitched. Like earned run average (ERA), a summary measure of pitching performance.

Wild Card Game After the end of the regular season, a one-game playoff game between two wildcard teams, which are the teams with the best records without having won their divisions.

wild pitch Pitch that gets past the catcher that is the fault of the pitcher, as opposed to a passed ball, which is the fault of the catcher. A wild pitch is viewed as uncatchable.

windup position Legal pitching position usually used when there are no offensive players on base. In the windup position, the pitcher does not come to a complete stop prior to throwing the ball. The other legal position is the *set position*.

World Series Best-of-seven game playoff at the end of the

Major League Baseball season, featuring the winner of the American League Championship Series (pennant) against the winner of the National League Championship Series (pennant). Played every year since 1903, with the exception of 1904 and 1994.

Bibliography

- Abrams, R. I. 2010. *The Money Pitch: Baseball Free Agency and Salary Arbitration*. Philadelphia: Temple University Press.
- Adler, J. 2006. *Baseball Hacks: Tips & Tools for Analyzing and Winning with Statistics*. Sebastopol, Calif.: O'Reilly.
- Agresti, A. 2013. *Categorical Data Analysis* (third ed.). New York: Wiley.
- Akaike, H. 1973. Information theory and an extension of the maximum likelihood principle. In B. N. Petrov and F. Csaki (eds.), *Second International Symposium on Information Theory*, pp. 267–281. Budapest: Akademiai Kiado. 214
- Alamar, B. C. 2013. *Sports Analytics: A Guide for Coaches, Managers, and Other Decision Makers*. New York: Columbia University Press.
- Albert, J. 2009. *Bayesian Computation with R*. New York: Springer. 34, 205
- Albert, J. H. 2003. *Teaching Statistics Using Baseball*. Washington D.C.: The Mathematical Association of America.
- Albert, J. H. and J. Bennett 2001. *Curve Ball: Baseball, Statistics, and the Role of Chance in the Game*. New York: Springer. 34
- Albert, J. H., J. Bennett, and J. J. Cochran (eds.) 2005. *Anthology of Statistics in Sports*. Alexandria, Va.: ASA-SIAM.

- Albert, J. H. and R. H. Koning 2007. *Statistical Thinking in Sports*. New York: Chapman & Hall/CRC. in press.
- Alder, J. 2015. Super bowl attendance. Retrieved from the World Wide Web on August 5, 2015 at <http://football.about.com/od/histo2/a/SBattendance.htm/.9>
- Alfons, A. 2014a. *cvTools: Cross-Validation Tools for Regression Models*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/cvTools/cvTools.pdf>.
- Alfons, A. 2014b. *simFrame: Simulation Framework*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/simFrame/simFrame.pdf>. 215
- Alfons, A., M. Templ, and P. Filzmoser 2014. *An Object-Oriented Framework for Statistical Simulation: The R Package simFrame*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/simFrame/vignettes/simFrame-intro.pdf>. 215
- Allen, M. J. and W. M. Yen 2002. *Introduction to Measurement Theory*. Prospect Heights, Ill.: Waveland Press.
- Allison, P. D. 2010. *Survival Analysis Using SAS: A Practical Guide* (second ed.). Cary, N.C.: SAS Institute Inc.
- American Football Coaches Association 1999a. *Defensive Football Strategies*. Champaign, Ill.: Human Kinetics.

- American Football Coaches Association 1999b. *Offensive Football Strategies*. Champaign, Ill.: Human Kinetics.
- Amor, D. 2002. *The E-business (R)evolution: Living and Working in an Interconnected World* (second ed.). Upper Saddle River, N.J.: Prentice Hall.
- Anand, S. S. and A. G. Büchner 2002. Database marketing and web mining. In W. Klösgen and J. M. Zytkow (eds.), *Handbook of Data Mining and Knowledge Discovery*, Chapter 46.1, pp. 843–849. Oxford: Oxford University Press.
- Andersen, P. K., Ø. Borgan, R. D. Gill, and N. Keiding 1993. *Statistical Models Based on Counting Processes*. New York: Springer.
- Anderson, C. and D. Sally 2013. *The Numbers Game: Why Everything You Know About Soccer is Wrong*. New York: Penguin Press.
- Angell, R. 1972. *The Summer Game*. New York: Ballantine Books.
- Angell, R. 1977. *Five Seasons: A Baseball Companion*. New York: Simon & Schuster.
- Angell, R. 1982. *Late Innings: A Baseball Companion*. New York: Simon & Schuster.
- Angell, R. 2003. *Game Time: A Baseball Companion*. Orlando, Fla.: Harcourt.
- Anscombe, F. J. 1973, February. Graphs in statistical analysis. *The American Statistician* 27: 17–21.
- Appleton, D. R. 1995. May the best man win? *The Statistician* 44(4):529–538. 42
- Armstrong, J. S. (ed.) 2001. *Principles of Forecasting: A*

Handbook for Researchers and Practitioners. Boston:
Kluwer.

Asinof, E. 1963. *Eight Men Out: The Black Sox and the 1919 World Series.* New York: Holt, Rinehart & Winston.

Asur, S. and B. A. Huberman 2010. Predicting the future with social media. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '10, pp. 492–499. Washington, DC, USA: IEEE Computer Society.
<http://dx.doi.org/10.1109/WI-IAT.2010.63>. 224

Athanasopoulos, G., R. A. Ahmed, and R. J. Hyndman 2009. Hierarchical forecasts for Australian domestic tourism. *International Journal of Forecasting* 25:146–166.

Avery, C. and J. Chevalier 1999. Identifying investor sentiment from price paths: the case of football betting. *Journal of Business* 72(4):493–521.

Bacon, L. D. 2002. Marketing. In W. Klösgen and J. M. Z̄ytkow (eds.), *Handbook of Data Mining and Knowledge Discovery*, Chapter 34, pp. 715–725. Oxford: Oxford University Press.

Badenhausen, K., M. K. Ozanian, and C. Settimi 2014, August 20. Cowboys are the first U.S. team to top \$3b valuation. *Forbes*. Retrieved from the World Wide Web, August 3, 2015, at <http://www.forbes.com/nfl-valuations/>.

Badenhausen, K., M. K. Ozanian, and C. Settimi 2015a, January 21. Lakers top 2015 list of NBA's most valuable teams; average team worth record \$1.1 billion. *Forbes*. Retrieved from the World Wide Web, August 3, 2015, at

<http://www.forbes.com/nba-valuations/>. 140

Badenhausen, K., M. K. Ozanian, and C. Settimi 2015b, March 25. MLB worth \$36b as team values hit record \$1.2b average. *Forbes*. Retrieved from the World Wide Web, August 3, 2015, at <http://www.forbes.com/mlb-valuations/>.

Baeza-Yates, R. and B. Ribeiro-Neto 2011. *Modern Information Retrieval* (second ed.). New York: Addison-Wesley.

Bahga, A. and V. Madisetti 2014. *Cloud Computing: A Hands-On Approach*. Atlanta. Self-published with online information at www.cloudcomputingbook.info.

Baker, B. O., C. D. Hardyck, and L. F. Petrino 1966. Weak measurements vs. strong statistics: An empirical critique of S. S. Stevens' proscriptions on statistics. *Educational and Psychological Measurement* 26:291–309. Reprinted in Mehrens and Ebel (1967).

Bar-Eli, M., S. Avugos, and M. Raab 2006. Twenty years of “hot hand” research: Review and critique. *Journal of Sport and Exercise* 7:525–553.

Barlow, G. L. 2000. Capacity management in the football industry. In A. Ingold, U. McMahon-Beattie, and I. Yeoman (eds.), *Yield Management: Strategies for Service Industries* (second ed.), Chapter 20, pp. 303–314. London UK: Thomson Learning.

Barry, D. and J. A. Hartigan 1993, September. Choice models for predicting divisional winners in major league baseball. *Journal of the American Statistical Association* 88(423): 766–774.

- Bates, D. M. and D. G. Watts 2007. *Nonlinear Regression Analysis and Its Applications*. New York: Wiley.
- Baumer, B. and A. Zimbalist 2014. *The Sabermetric Revolution: Assessing the Growth of Analytics in Baseball*. Philadelphia: University of Pennsylvania Press. 295
- Baumer, B. S., S. T. Jensen, and G. J. Matthews 2015, June. openWAR: An open source system for evaluating overall player performance in Major League Baseball. *Journal of Quantitative Analysis in Sports* 11(2):69–84.
- Baumohl, B. 2008. *The Secrets of Economic Indicators: Hidden Clues to Future Economic Trends and Investment Opportunities* (second ed.). Upper Saddle River, N.J.: Pearson.
- Beazley, D. M. 2009. *Python Essential Reference* (fourth ed.). Upper Saddle River, N.J.: Pearson Education.
- Beazley, D. M. and B. K. Jones 2013. *Python Cookbook* (third ed.). Sebastopol, Calif.: O'Reilly.
- Becker, R. A. and W. S. Cleveland 1996. *S-Plus TrellisTM Graphics User's Manual*. Seattle: MathSoft, Inc. 238
- Belew, R. K. 2000. *Finding Out About: A Cognitive Perspective on Search Engine Technology and the WWW*. Cambridge: Cambridge University Press.
- Belichick, S. 2008. *Football Scouting Methods* (reissue ed.). Saratoga, Calif.: Liber Apertus Press. Written by Bill Belichick's father.
- Belsley, D. A., E. Kuh, and R. E. Welsch 1980. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: Wiley.
- Bennett, J. M. and J. A. Flueck 1983, February. An evaluation

- of major league baseball offensive performance measures.
The American Statistician 37(1):76–82.
- Berk, R. A. 2008. *Statistical Learning from a Regression Perspective*. New York: Springer.
- Berndt, E. R. 1991. *The Practice of Econometrics*. Reading, Mass.: Addison-Wesley.
- Berners-Lee, T. 2000. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. New York: HarperBusiness.
- Berri, D. J. and M. B. Schmidt 2010. *Stumbling on Wins: Two Economists Expose the Pitfalls on the Road to Victory in Professional Sports*. Upper Saddle River, N.J.: FT Press/Pearson. 67
- Betz, N. E. and D. J. Weiss 2001. Validity. In B. Bolton (ed.), *Handbook of Measurement and Evaluation in Rehabilitation* (third ed.), pp. 49–73. Gaithersburg, Md.: Aspen Publishers.
- Bird, S., E. Klein, and E. Loper 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. Sebastopol, Calif.: O'Reilly.
<http://www.nltk.org/book/>. 35
-
- Birge, J. R. and F. Louveaux 2011. *Introduction to Stochastic Programming*. New York: Springer.
- Bishop, Y. M. M., S. E. Fienberg, and P. W. Holland 1975. *Discrete Multivariate Analysis: Theory and Practice*. Cambridge: MIT Press.
- Blattberg, R. C., B.-D. Kim, and S. A. Neslin 2008. *Database Marketing: Analyzing and Managing Customers*. New York: Springer.

- Blei, D., A. Ng, and M. Jordan 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research* 3:993–1022. 216
- Blount, Jr., R. 1993. Yogi. In M. Mulvoy (ed.), *Baseball: Four Decades of Sports Illustrated's Finest Writing on America's Favorite Pastime*, pp. 171–190. Birmingham: Oxmoor House.
- Bodendieck, Z. and T. Gatto (eds.) 2006. *2007 Baseball Register*. Chesterfield, Mo.: Sporting News.
- Bollen, J., H. Mao, and X. Zeng 2011. Twitter mood predicts the stock market. *Journal of Computational Science* 2:1–8.
- Borg, I. and P. J. F. Groenen 2010. *Modern Multidimensional Scaling: Theory and Applications* (second ed.). New York: Springer. 81
- Borgatti, S. P., M. G. Everett, and J. C. Johnson 2013. *Analyzing Social Networks*. Thousand Oaks, Calif.: Sage. 232
- Boswell, T. 1984. *Why Time Begins on Opening Day*. Garden City, N.Y.: Doubleday.
- Boswell, T. 1989. *The Heart of the Order*. New York: Doubleday.
- Boswell, T. 1994. *Cracking the Show*. New York: Doubleday.
- Bouton, J. 1981. *Ball Four, Plus Ball Five: An Update, 1970–1980*. New York: Stein & Day.
- Box, G. E. P. and D. R. Cox 1964. An analysis of transformations. *Journal of the Royal Statistical Society, Series B (Methodological)* 26(2):211–252.
- Box, G. E. P., G. M. Jenkins, and G. C. Reinsel 2008. *Time Series Analysis: Forecasting and Control* (fourth ed.). New

York: Wiley. 227

Bradbury, J. C. 2007. *The Baseball Economist: The Real Game Exposed*. New York: Dutton. 13, 14

Bradley, R. A. 1976, June. Science, statistics, and paired comparisons. *Biometrics* 32(2): 213–239.

Bradley, R. A. and M. E. Terry 1952, December. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika* 39(3/4):324–345.

Bradley, S. P., A. C. Hax, and T. L. Magnanti 1977. *Applied Mathematical Programming*. Reading, Mass.: Addison-Wesley.

Braun, W. J. and D. J. Murdoch 2007. *A First Course in Statistical Programming with R*. Cambridge, UK: Cambridge University Press.

Brealey, R., S. Myers, and F. Allen 2013. *Principles of Corporate Finance*. New York: McGraw-Hill/Irwin.

Breiman, L. 2001. Statistical modeling: The two cultures. *Statistical Science* 16(3):199–215.

Briskorn 2008. *Sports Leagues Scheduling: Models, Combinatorial Properties, and Optimization Algorithms*. New York: Springer.

Broughton, D. 2012, November 12–18. Everybody loves bobbleheads. *Sports Business Journal*:9. Retrieved from the World Wide Web at
<http://www.sportsbusinessdaily.com/Journal/Issues/2012/11/12/Research-and-Ratings/Bobbleheads.aspx>. 111

Brown, C. B. 2015. *The Art of Smart Football*. San Bernardino, Calif. Self-published.

- Brown, F. G. 1976. *Principles of Educational and Psychological Testing* (Second ed.). New York: Holt, Rinehart, and Winston. 35
- Brownlee, J. 2011. *Clever Algorithms: Nature-Inspired Programming Recipes*. Melbourne, Australia: Creative Commons. <http://www.CleverAlgorithms.com>.
216
- Bühlmann, P. and S. van de Geer 2011. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. New York: Springer. 213
- Bukiet, B., R. Elliotte, and J. Palacios 1997. A markov chain approach to baseball. *Operations Research* 45:14–23.
- Burnham, K. P. and D. R. Anderson 2002. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach* (second ed.). New York: Springer-Verlag.
- Buvač, V. and P. J. Stone 2001, April 2. The General Inquirer user's guide. Software developed with the support of Harvard University and The Gallup Organization.
- Cairo, A. 2013. *The Functional Art: An Introduction to Information Graphics and Visualization*. Berkeley, Calif: New Riders.
- Cameron, A. C. and P. K. Trivedi 1998. *Regression Analysis of Count Data*. Cambridge: Cambridge University Press.
- Campbell, D. T. and D. W. Fiske 1959. Convergent validity and discriminant validity by the multitrait-multimethod matrix. *Psychological Bulletin* 56:81–105.
- Campbell, S. and S. Swigart 2014. *Going beyond Google: Gathering Internet Intelligence* (5 ed.). Oregon City, Oreg.: Cascade Insights.

<http://www.cascadeinsights.com/gbgdownload>.

- Carlin, B. P. 1996, February. Improved NCAA basketball tournament modeling via point spread and team strength information. *The American Statistician* 50(1):39–43. 42
- Carlin, B. P. and T. A. Louis 1996. *Bayes and Empirical Bayes Methods for Data Analysis*. London: Chapman & Hall. 205
- Carr, D., N. Lewin-Koh, and M. Maechler 2014. *hexbin: Hexagonal Binning Routines*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/hexbin/hexbin.pdf>. 238
- Carr, J. 2015, June 18. Final: Dodgers get balk-off win against Rangers, 1-0. *The Orange County Register*. Retrieved from the World Wide Web on September 12, 2015 at <http://ocregister.com/dodgers/writer-667515-dodgers-final.html/>. 196
- Carroll, B., P. Palmer, and J. Thorn 1988. *The Hidden Game of Football*. New York: Warner Books.
- Carroll, B., P. Palmer, J. Thorn, and D. Pietrusza (eds.) 1998. *The Hidden Game of Football: The Next Edition*. Kingston, N.Y.: Total Sports.
- Carroll, J. D. and P. E. Green 1997. Psychometric methods in marketing research: Part II, multidimensional scaling. *Journal of Marketing Research* 34:193–204. 81
- Carter, D. M. 2011. *Money Games Profiting from the Convergence of Sports and Entertainment*. Stanford, Calif.: Stanford University Press.
- Casciano, C. 2011. *The CSS Pocket Guide*. Upper Saddle

- River, N.J.: Pearson/Peachpit.
- Ceri, S., A. Bozzon, M. Brambilla, E. D. Valle, P. Fraternali, and S. Quarteroni 2013. *Web Information Retrieval*. New York: Springer.
- Cespedes, F. V., J. P. Dougherty, and B. S. Skinner, III 2013, Winter. How to identify the best customers for your business. *MIT Sloan Management Review* 54(2):53–59.
- Chakrabarti, S. 2003. *Mining the Web: Discovering Knowledge from Hypertext Data*. San Francisco: Morgan Kaufmann.
- Chambers, J. M. and T. J. Hastie (eds.) 1992. *Statistical Models in S*. Pacific Grove, Calif.: Wadsworth & Brooks/Cole. Champions of S, S-Plus, and R call this “the white book.” It introduced statistical modeling syntax using S3 classes.
- Chang, W. 2013. *R Graphics Cookbook*. Sebastopol, Calif.: O’Reilly.
- Charniak, E. 1993. *Statistical Language Learning*. Cambridge: MIT Press.
- Chatterjee, S. and A. S. Hadi 2012. *Regression Analysis by Example* (fifth ed.). New York: Wiley.
- Chen, D.-S., R. G. Batson, and Y. Dang 2010. *Applied Integer Programming: Modeling and Solution*. New York: Wiley.
- Chihara, L. and T. Hesterberg 2011. *Mathematical Statistics with Resampling and R*. New York: Wiley.
- Chiusano, P. and R. Bjarnason 2015. *Functional Programming in Scala*. Shelter Island, N.Y.: Manning.
- Chodorow, K. 2013. *MongoDB: The Definitive Guide* (second ed.). Sebastopol, Calif.: O’Reilly.

- Christensen, R. 1997. *Log-Linear Models and Logistic-Regression* (second ed.). New York: Springer.
- Chun, W. J. 2007. *Core Python Programming* (second ed.). Upper Saddle River, N.J.: Pearson Education.
- Chvátal, V. 1983. *Linear Programming*. New York: W. H. Freeman and Company.
- Cleveland, W. S. 1993. *Visualizing Data*. Murray Hill, N.J.: AT&T Bell Laboratories. Initial documentation for trellis graphics in S-Plus. 238
- Cohen, J. 1960, April. A coefficient of agreement for nominal data. *Educational and Psychological Measurement* 20(1):37–46. 211
- Commandeur, J. J. F. and S. J. Koopman (eds.) 2007. *An Introduction to State Space Time Series Analysis*. Oxford: Oxford University Press. 228
- Conejo, A. J., M. Carrión, and J. M. Morales 2010. *Decision Making Under Uncertainty in Electricity Markets*. New York: Springer. Chapter 2: Stochastic Programming Fundamentals.
- Connolly, T. and C. Begg 2015. *Database Systems: A Practical Approach to Design, Implementation, and Management* (sixth ed.). New York: Pearson.
- Conway, D. and J. M. White 2012. *Machine Learning for Hackers*. Sebastopol, Calif.: O'Reilly.
- Cook, E. 1966. *Percentage Baseball*. Cambridge, Mass.: MIT Press.
- Cook, R. D. 1998. *Regression Graphics: Ideas for Studying Regressions through Graphics*. New York: Wiley.
- Cook, R. D. 2007. Fisher lecture: Dimension reduction in

- regression. *Statistical Science* 22: 1–26.
- Cook, R. D. and S. Weisberg 1999. *Applied Regression Including Computing and Graphics*. New York: Wiley.
- Copeland, R. 2013. *MongoDB Applied Design Patterns* (second ed.). Sebastopol, Calif.: O'Reilly.
- Cover, T. M. 1978, January–February. Errata: An offensive earned-run average for baseball. *Operations Research* 27(1):207.
- Cover, T. M. and C. W. Keilers 1977, October. An offensive earned-run average for baseball. *Operations Research* 25(5):729–740. Also see Cover (1978).
- Cowpertwait, P. S. P. and A. V. Metcalfe 2009. *Introductory Time Series with R*. New York: Springer.
- Cox, T. F. and M. A. A. Cox 1994. *Multidimensional Scaling*. London: Chapman & Hall. 81
- Cronbach, L. J. 1951. Coefficient alpha and the internal structure of tests. *Psychometrika* 16:297–334. 35
- Davenport, T. H. and J. G. Harris 2007. *Competing on Analytics: The New Science of Winning*. Boston: Harvard Business School Press. 199
- Davenport, T. H., J. G. Harris, and R. Morison 2010. *Analytics at Work: Smarter Decisions, Better Results*. Boston: Harvard Business School Press. 199
- David, H. A. 1963. *The Method of Paired Comparisons*. London: Charles Griffin & Company Limited.
- Davidson, R. R. and P. H. Farquhar 1976, June. A bibliography on the method of paired comparisons. *Biometrics* 32(2):241–252.

- Davison, A. C. and D. V. Hinkley 1997. *Bootstrap Methods and their Application*. Cambridge: Cambridge University Press.
- Davison, M. L. 1992. *Multidimensional Scaling*. Melbourne, Fla.: Krieger. 81
- Day, Z. 2013, December 5. Measuring pitching with trackman: the secrets of fastball spin. *BaseballProspectus.com*. Retrieved from the World Wide Web, September 19, 2015, at
<http://www.baseballprospectus.com/article.php?articleid=22362/>.
- Dean, J. and S. Ghemawat 2004. MapReduce: Simplified Data Processing on Large Clusters. Retrieved from the World Wide Web at
http://static.usenix.org/event/osdi04/tech/full_papers/dean/dean.pdf.
- Delen, D., R. Sharda, and P. Kumar 2007. Movie forecast guru: A Web-based DSS for Hollywood managers. *Decision Support Systems* 43(4):1151–1170. 224
- Derderian, J.-C. 1978, May. Maximum hedges. *Mathematics Magazine* 51(3):188–192.
- DeSimone, B. 1999, March 9. Gambling and sports: laying it on the line. *The Chicago Tribune*.
- D'Esopo, D. A. and B. Lefkowitz 1977. The distribution of runs in a game of baseball. In S. P. Ladany and R. E. Machol (eds.), *Optimal Strategies in Sports*, pp. 55–62. New York: Elsevier-North Holland.
- Dewan, J. and B. Jedlovec 2015. *The Fielding Bible*. Chicago: ACTA Sports.

- Dickson, P. R. 1997. *Marketing Management* (second ed.). Orlando, Fla.: Harcourt Brace & Company. 111, 131
- Downing, D., M. Covington, M. Covington, C. A. Barrett, and S. Covington 2012. *Dictionary of Computer and Internet Terms* (eleventh ed.). Hauppauge, N.Y.: Barron's Educational Series.
- Draper, N. R. and H. Smith 1998. *Applied Regression Analysis* (third ed.). New York: Wiley.
- Dresow, K. 2015. *Offensive Football Systems*. San Bernardino, Calif. Self-published.
- Duda, R. O., P. E. Hart, and D. G. Stork 2001. *Pattern Classification* (second ed.). New York: Wiley.
- Dumais, S. T. 2004. Latent semantic analysis. In B. Cronin (ed.), *Annual Review of Information Science and Technology*, Volume 38, Chapter 4, pp. 189–230. Medford, N.J.: Information Today.
- Durbin, J. and S. J. Koopman 2012. *Time Series Analysis by State Space Methods* (second ed.). New York: Oxford University Press. 228
- Dwyre, B. 2015, September 23. NFL would like story to go away. *Los Angeles Times*:D1, D5. 185
- Early, G. L. 2011. *A Level Playing Field: African American Athletes and the Republic of Sports*. Cambridge, Mass.: Harvard University Press.
- Efron, B. 1986. Why isn't everyone a Bayesian (with commentary). *The American Statistician* 40(1):1–11.
- Efron, B. and R. Tibshirani 1993. *An Introduction to the Bootstrap*. London: Chapman and Hall.
- Ellis, B. 2015. *Real-Time Analytics: Techniques to Analyze*

- and Visualize Streaming Data*. New York: Wiley.
- Enders, W. 2010. *Applied Econometric Time Series* (third ed.). New York: Wiley.
- Engelbrecht, A. P. 2007. *Computational Intelligence: An Introduction* (second ed.). New York: Wiley. 216
- Epstein, D. 2013. *The Sports Gene: Inside the Science of Extraordinary Athletic Performance*. New York: Current/Penguin.
- Erdős, P. and A. Rényi 1959. On random graphs. *Publicationes Mathematicae* 6:290–297.
- Erdős, P. and A. Rényi 1960. On the evolution of random graphs. *Publicationes of the Mathematical Institute of the Hungarian Academy of Sciences* 5:17–61.
- Erdős, P. and A. Rényi 1961. On the strength of connectedness of a random graph. *Acta Mathematica Scientia Hungary* 12:261–267.
- ESPN.com 2015. MLB Park Factors 2015. Retrieved from the World Wide Web on September 12, 2015 at <http://espn.go.com/mlb/stats/parkfactor/>.
-
- 65
- Everitt, B. S., S. Landau, M. Leese, and D. Stahl 2011. *Cluster Analysis* (fifth ed.). New York: Wiley.
- Fair, R. C. 2008. Estimated age effects in baseball. *Journal of Quantitative Analysis in Sports* 4(1):1–39. 33
- FanGraphs.com 2015. RE24: FanGraphs Sabermetrics Library. Retrieved from the World Wide Web on September 8, 2015 at <http://www.fangraphs.com/library/misc/re24/>. 62

- Fawcett, T. 2003, January 7. ROC graphs: Notes and practical considerations for researchers.
<http://www.hpl.hp.com/techreports/2003/HPL-2003-4.pdf>.
- Feldman, D. M. 2002, Winter. The pricing puzzle. *Marketing Research*:14–19.
- Feldman, R. 2013. Techniques and applications for sentiment analysis. *Communications of the ACM* 56(4):82–89.
- Feldman, R. and J. Sanger 2007. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge, UK: Cambridge University Press.
- Feller, W. 1968. *An Introduction to Probability Theory and Its Applications* (third ed.), Volume I. New York: Wiley.
- Feller, W. 1971. *An Introduction to Probability Theory and Its Applications*, Volume 2. New York: Wiley.
- Few, S. 2009. *Now You See It: Simple Visualization Techniques and Quantitative Analysis*. Oakland, Calif.: Analytics Press.
- Fienberg, S. E. 2007. *Analysis of Cross-Classified Categorical Data* (second ed.). New York: Springer.
- Firth, D. 1991. Generalized linear models. In D. Hinkley and E. Snell (eds.), *Statistical Theory and Modeling: In Honour of Sir David Cox, FRS*, Chapter 3, pp. 55–82. London: Chapman and Hall.
- Fisher, R. A. 1970. *Statistical Methods for Research Workers* (fourteenth ed.). Edinburgh: Oliver and Boyd. First edition published in 1925. 205
- Fisher, R. A. 1971. *Design of Experiments* (ninth ed.). New York: Macmillan. First edition published in 1935. 205
- Fiske, D. W. 1971. *Measuring the Concepts of Personality*.

Chicago: Aldine. 35

- Flam, F. 2014, September 30. The odds, continually updated. *The New York Times*:D1. Retrieved from the World Wide Web at
http://www.nytimes.com/2014/09/30/science/the-odds-continually-updated.html?_r=0.
205
- Fogus, M. and C. Houser 2014. *The Joy of Clojure*. Shelter Island, N.Y.: Manning. 182
- Forbes 2015. Brooklyn Nets. Retrieved from the World Wide Web on August 24, 2015 at
<http://www.forbes.com/teams/brooklyn-nets/>. 140
- Fort, R. D. 2006. Value of Major League Baseball ownership. *International Journal of Sport Finance* 1(1):9–20. 142
- Fort, R. D. 2011. *Sports Economics* (third ed.). Upper Saddle River, N.J.: Prentice Hall.
- Fort, R. D. and J. Winfree 2013. *15 Sports Myths and Why They're Wrong* (third ed.). Stanford, Calif.: Stanford University Press.
- Foster, G. 2013, August 26. Behold the power of a free bobblehead doll. *The Wall Street Journal*. Retrieved from the World Wide Web at
<http://online.wsj.com/article/SB10001424127887323407104579037182599696524.html>.
111
- Fox, J. 2002, January. Robust regression: Appendix to an R and S-PLUS companion to applied regression. Retrieved from the World Wide Web at <http://cran.r-project.org/doc/contrib/Fox-Companion/appendix.R>

[project.org/doc/contrib/Fox-
Companion/appendix-robust-regression.pdf.](http://project.org/doc/contrib/Fox-Companion/appendix-robust-regression.pdf)
214

Fox, J. 2014. *car: Companion to Applied Regression*.

Comprehensive R Archive Network. 2014.

<http://cran.r->

project.org/web/packages/car/car.pdf.

Fox, J. and S. Weisberg 2011. *An R Companion to Applied Regression* (second ed.). Thousand Oaks, Calif.: Sage.

Frank, R. E., W. F. Massey, and Y. Wind 1972. *Market Segmentation*. Englewood Cliffs, N.J.: Prentice Hall.

Franks, B. 2012. *Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics*. Hoboken, N.J.: Wiley. 199

Franses, P. H. and R. Paap 2001. *Quantitative Models in Marketing Research*. Cambridge: Cambridge University Press.

Frees, E. W. and T. W. Miller 2004. Sales forecasting with longitudinal data models. *International Journal of Forecasting* 20:99–114.

Fréville, A. 2004. The multidimensional 0-1 knapsack problem: an overview. *European Journal of Operational Research* 155:1–21. 41

Gabriel, K. R. 1971. The biplot graphical display of matrices with application to principal component analysis. *Biometrika* 58:453–467. 81

Gandolfi, G. (ed.) 2009. *NBA Coaches Playbook: Techniques, Tactics, and Teaching Points*. Champaign, Ill.: Human Kinetics.

- Garcia-Molina, H., J. D. Ullman, and J. Widom 2009. *Database Systems: The Complete Book* (second ed.). Upper Saddle River, N.J.: Prentice-Hall.
- Gasston, P. 2011. *The Book of CSS3: A Developer's Guide to the Future of Web Design*. San Francisco: No Starch Press.
- Gasston, P. 2013. *The Modern Web: Multi-Device Web Development with HTML5, CSS3, and JavaScript*. San Francisco: No Starch Press.
- Geisser, S. 1993. *Predictive Inference: An Introduction*. New York: Chapman & Hall. 199, 205
- Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin 1995. *Bayesian Data Analysis*. London: Chapman & Hall. 205
- Gelman, A., J. Hill, Y.-S. Su, M. Yajima, and M. G. Pittau 2014. *mi: Missing Data Imputation and Model Checking*. Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/mi/mi.pdf>.
- Gershkov, A. and B. Moldavanu 2014. *Dynamic Allocation and Pricing: A Mechanism Design Approach*. Cambridge, Mass.: The MIT Press.
- Gheorghe, R., M. L. Hinman, and R. Russo 2015. *Elasticsearch in Action*. Shelter Island, N.Y.: Manning.
- Ghiselli, E. E. 1964. *Theory of Psychological Measurement*. New York: McGraw-Hill. 35
- Glickman, M. E. and H. S. Stern 1998. A state-space model for national football league scores. *Journal of the American Statistical Association* 93(441):25–35. 42
- Gnanadesikan, R. 1997. *Methods for Statistical Data Analysis of Multivariate Observations* (second ed.). New York:

Wiley.

Goldman, S. (ed.) 2005. *Mind Game: How the Boston Red Sox Got Smart, Won a World Series, and Created a New Blueprint for Winning*. New York: Workman Publishing.

Goodwin, D. K. 1997. *Wait Till Next Year*. New York: Simon & Schuster. 120

Gorman, J. and K. Calhoun 1994. *The Name of the Game: The Business of Sports*. New York: Wiley.

Gower, J. C. and D. J. Hand 1996. *Biplots*. London: Chapman & Hall. 81

Granger, C. W. 1969. Investigating causal relations by econometric models and cross-spectral methods.
Econometrica 37:424–438.

Graybill, F. A. 1961. *Introduction to Linear Statistical Models, Volume 1*. New York: McGraw-Hill.

Graybill, F. A. 2000. *Theory and Application of the Linear Model*. Stamford, Conn.: Cengage Learning.

Greene, W. H. 2002. *Econometric Analysis* (fifth ed.). Upper Saddle River, N.J.: Prentice Hall. 131

Greene, W. H. 2012. *Econometric Analysis* (seventh ed.). Upper Saddle River, N.J.: Pearson Prentice Hall.

Gries, M. 2015, September 9. Analyzing StatCast. Presentation at the Sports Innovation Conference, San Francisco, Calif. 170

Groeneveld, R. A. 1990, November. Ranking teams in a league with two divisions of t teams. *The American Statistician* 44(4):277–281. 42

Grolemund, G. 2014. *Hands-On Programming with R: Write*

Your Own Functions and Simulations. Sebastopol, Calif.: O'Reilly.

Grolemund, G. and H. Wickham 2011, April 7. Dates and times made easy with lubridate. *Journal of Statistical Software* 40(3):1–25.
<http://www.jstatsoft.org/v40/i03>.

Grolemund, G. and H. Wickham 2014. *lubridate: Make Dealing with Dates a Little Easier.* Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/lubridate/lubridate.pdf>.

Gromley, C. and Z. Tong 2015. *Elasticsearch: The Definitive Guide.* Sebastopol, Calif.: O'Reilly.

Grossman, A. 2015, August 25. Does Competitive Balance Drive Interest In Sports? Forbes/Business retrieved from the World Wide Web, October 5, 2015, at
<http://www.forbes.com/sites/kurtbadenhauen/2015/08/25/does-competitive-balance-drive-interest-in-sports/>.

Guilford, J. P. 1936. *Psychometric Methods.* New York: McGraw-Hill.

Guilford, J. P. 1954. *Psychometric Methods* (second ed.). New York: McGraw-Hill. First edition published in 1936.

Gulliksen, H. 1950. *Theory of Mental Tests.* New York: Wiley.
35

Guttag, J. V. 2013. *Introduction to Computation and Programming Using Python* (revised and expanded ed.). Cambridge, Mass.: The MIT Press.

Hadley, G. 1962. *Linear Programming.* Reading, Mass.:

Addison-Wesley.

Halberstam, D. 1994. *October 1964*. New York: Villard Books.

Halberstam, D. 2000. *Playing for Keeps: Michael Jordan & the World He Made*. New York: Broadway Books.

Halberstam, D. 2005. *The Education of a Coach*. New York: Hyperion. The coach described in this book is Bill Belichick.

Halberstam, D. 2009. *The Breaks of the Game*. New York: Hyperion.

Hale, C. and J. Scanlon 1999. *Wired Style: Principles of English Usage in the Digital Age*. New York: Broadway Books.

Hamilton, J. D. 1994. *Time Series Analysis*. Princeton, N.J.: Princeton University Press.

Hamble, Z. 2007. *Watching Baseball Smarter: A Professional Fan's Guide for Beginners, Semi-experts, and Deeply Serious Geeks* (Revised and Updated ed.). New York: Vintage/Random House.

Hand, D. J. 1997. *Construction and Assessment of Classification Rules*. New York: Wiley.

Hanssens, D. M., L. J. Parsons, and R. L. Schultz 2001. *Market Response Models: Econometric and Time Series Analysis* (second ed.). Boston: Kluwer.

Harrell, Jr., F. E. 2001. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. New York: Springer.

Hart, W. E., C. Laird, J.-P. Watson, and D. L. Woodruff 2012. *Pyomo—Optimization Modeling in Python*. New York:

Springer.

Hart, W. E. and D. L. Woodruff 2015. Pyomo Online Documentation 4.1. Retrieved from the World Wide Web on September 24, 2015 at
<https://software.sandia.gov/downloads/pu/b/pyomo/PyomoOnlineDocs.pdf/>.

Hastie, T., R. Tibshirani, and J. Friedman 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (second ed.). New York: Springer.

Hastie, T. J. 1992. Generalized linear models. In J. M. Chambers and T. J. Hastie (eds.), *Statistical Models in S*, Chapter 6, pp. 195–247. Pacific Grove, Calif.: Wadsworth & Brooks/Cole.

Hausser, R. 2001. *Foundations of Computational Linguistics: Human-Computer Communication in Natural Language* (second ed.). New York: Springer-Verlag.

Heer, J., M. Bostock, and V. Ogievetsky 2010, May 1. A tour through the visualization zoo: A survey of powerful visualization techniques, from the obvious to the obscure. *acmqueue: Association for Computing Machinery*:1–22. Retrieved from the World Wide Web at
<http://queue.acm.org/detail.cfm?id=1805128>.

Herbrich, R. and T. Graepel 2006, June. TrueSkill: A Bayesian skill rating system. Microsoft Research Technical Report MSR-TR-2006-80, retrieved from the World Wide Web, December 19, 2014, at
<http://research.microsoft.com/pubs/74419/tr-2006-80.pdf>.

- Higgins, R. 2015. *Analysis for Financial Management* (eleventh ed.). New York: McGraw-Hill.
- Hinkley, D. V., N. Reid, and E. J. Snell (eds.) 1991. *Statistical Theory and Modeling*. London: Chapman and Hall. 205
- Hirshleifer, J., A. Glazer, and D. Hirshleifer 2005. *Price Theory and Applications: Decisions, Markets, and Information* (seventh ed.). New York: Cambridge University Press. 131
- Hoberman, S. 2014. *Data Modeling for MongoDB: Building Well-Designed and Supportable MongoDB Databases*. Basking Ridge, N.J.: Technics Publications.
- Hoerl, A. E. and R. W. Kennard 2000. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics* 42(1):80–86. Reprinted from *Technometrics*, volume 12. 213
- Hoff, P. D. 2009. *A First Course in Bayesian Statistical Methods*. New York: Springer.
- Hoffman, P. and Scrapy Developers 2014, June 26. Scrapy documentation release 0.24.0. Supported at <http://scrapy.org/>. Documentation retrieved from the World Wide Web, October 26, 2014, at <https://media.readthedocs.org/pdf/scrapy/0.24/scrapy.pdf>.
- Holden, K., D. A. Peel, and J. L. Thompson 1990. *Economic Forecasting: An Introduction*. Cambridge, UK: Cambridge University Press.
- Honaker, J., G. King, and M. Blackwell 2014. *Amelia II: A Program for Missing Data*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/Amelia/index.html>

[project.org/web/packages/Amelia/Amelia.pdf](http://cran.r-project.org/web/packages/Amelia/Amelia.pdf).

Hosmer, D. W., S. Lemeshow, and S. May 2013. *Applied Survival Analysis: Regression Modeling of Time to Event Data* (second ed.). New York: Wiley.

Hosmer, D. W., S. Lemeshow, and R. X. Sturdivant 2013. *Applied Logistic Regression* (third ed.). New York: Wiley.

Hothorn, T., F. Leisch, A. Zeileis, and K. Hornik 2005, September. The design and analysis of benchmark experiments. *Journal of Computational and Graphical Statistics* 14(3):675– 699.

Hothorn, T., A. Zeileis, R. W. Farebrother, C. Cummins, G. Millo, and D. Mitchell 2014. *lmtest: Testing Linear Regression Models*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/lmtest/lmtest.pdf>.

Houng, B. 2014. *Package trueskill: Implementation of the TrueSkill Algorithm in R*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/trueskill/trueskill.pdf>.

Huet, S., A. Bouvier, M.-A. Poursat, and E. Jolivet 2004. *Statistical Tools for Nonlinear Regression: A Practical Guide with S-Plus and R Examples* (second ed.). New York: Springer.

Hughes, A. M. 2000. *Strategic Database Marketing: The Masterplan for Starting and Managing a Profitable Customer-Based Marketing Program* (second ed.). New

- York: McGraw-Hill.
- Humphreys, B. R. and D. R. Howard (eds.) 2008a. *The Business of Sports: Volume 1, Perspectives on the Sports Industry*, Volume 1. Westport, Conn.: Praeger.
- Humphreys, B. R. and D. R. Howard (eds.) 2008b. *The Business of Sports: Volume 2, Economic Perspectives on Sport*, Volume 2. Westport, Conn.: Praeger.
- Humphreys, B. R. and D. R. Howard (eds.) 2008c. *The Business of Sports: Volume 3, Bridging Research and Practice*, Volume 3. Westport, Conn.: Praeger.
- Hyndman, R. J., R. A. Ahmed, G. Athanasopoulos, and H. L. Shang 2011. Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis* 55:2579–2589.
- Hyndman, R. J. and G. Athanasopoulos 2014. *Forecasting: Principles and Practice*. Online: OTexts.
<https://www.otexts.org/fpp>.
- Hyndman, R. J., G. Athanasopoulos, S. Razbash, D. Schmidt, Z. Zhou, and Y. Khan 2014. *forecast: Forecasting Functions for Time Series and Linear Models*. Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/forecast/forecast.pdf>.
- Hyndman, R. J., A. B. Koehler, J. K. Ord, and R. D. Snyder 2008. *Forecasting with Exponential Smoothing: The State Space Approach*. New York: Springer. 228
- Hyslop, B. 2010. *The HTML Pocket Guide*. Upper Saddle River, N.J.: Pearson/Peachpit.

Ihaka, R., P. Murrell, K. Hornik, J. C. Fisher, and A. Zeileis
2014. *colorspace: Color Space Manipulation*.
Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/colorspace/colorspace.pdf>.

Indurkhy, N. and F. J. Damerau (eds.) 2010. *Handbook of Natural Language Processing* (second ed.). Boca Raton, Fla.: Chapman and Hall/CRC.

Ingersoll, G. S., T. S. Morton, and A. L. Farris 2013. *Taming Text: How to Find, Organize, and Manipulate It*. Shelter Island, N.Y.: Manning. 216

Issenberg, S. 2013. *The Victory Lab: The Secret Science of Winning Campaigns*. New York: Broadway Books.

Izenman, A. J. 2008. *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning*. New York: Springer.

James, B. 1997. *The Bill James Guide to Baseball Managers*. New York: Scribners.

James, B. 2010. *The New Bill James Historical Baseball Abstract*. New York: Free Press. 65

James, G., D. Witten, T. Hastie, and R. Tibshirani 2013. *An Introduction to Statistical Learning with Applications in R*. New York: Springer.

Jenkins, L. 2015a, June 22. Andre and the giant. *Sports Illustrated* 122(25):37–46.

Jenkins, L. 2015b, June 29. Class picture. *Sports Illustrated* 122(26):40–47.

Johnson, R. A. and D. W. Wichern 1998. *Applied Multivariate*

- Statistical Analysis* (fourth ed.). Upper Saddle River, N.J.: Prentice Hall.
- Jones, Maillardet, and Robinson 2014. *Introduction to Scientific Programming and Simulation Using R* (second ed.). Boca Raton, Fla.: Chapman & Hall/CRC.
- Jones, R. E. 1983. Scoring every inning. In L. R. Davis (ed.), *Insider's Baseball: The Finer Points of the Game, As Examined by The Society for American Baseball Research*, pp. 242– 247. New York: Charles Scribner's Sons.
- Joula, P. 2008. *Authorship Attribution*. Hanover, Mass.: Now Publishers. 225
- Jozsa, Jr., F. P. 2010. *The National Basketball Association: Business Organization and Strategy*. Danvers, Mass.: World Scientific Publishing.
- Judge, G. G., W. E. Griffiths, R. C. Hill, H. Lütkepohl, and T.-C. Lee 1985. *The Theory and Practice of Econometrics* (second ed.). New York: Wiley.
- Jurafsky, D. and J. H. Martin 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (second ed.). Upper Saddle River, N.J.: Prentice Hall.
- Kahn, L. M. 2000. The sports business as a labor market laboratory. *Journal of Economic Perspectives* 14(3):75–94.
- Kalyanaram, G. and R. S. Winer 1995. Empirical generalizations from reference price research. *Marketing Science* 14(3):G161–G169.
- Kaplan, E. H. and S. J. Garstka 2001. March madness and the

- office pool. *Management Science* 47(3):369–382. 42
- Karaesmen, I. Z., G. J. van Ryzin, K. T. Talluri, and G. J. Vulcano 2008. Revenue management: Models and methods. In S. Mason, R. R. Hill, L. Monch, O. Rose, T. Jefferson, and J. Fowler (eds.), *Proceedings of the 2008 Winter Simulation Conference*, pp. 145–155. New York: Springer. 131
- Karau, H., A. Konwinski, P. Wendell, and M. Zaharia 2015. *Learning Spark: Lightning-Fast Data Analysis*. Sebastopol, Calif.: O'Reilly.
- Kaufman, L. and P. J. Rousseeuw 1990. *Finding Groups in Data: An Introduction to Cluster Analysis*. New York: Wiley.
- Keating, P. 2015, October 12. The Owners Issue. *ESPN*:50–51. Introduction to special issue about owners of professional team sports. 130
- Keller, J. B. 1994. A characterization of the Poisson distribution and the probability of winning a game. *The American Statistician* 48(4):294–298.
- Kellerer, H., U. Pferschy, and D. Pisinger 2004. *Knapsack Problems*. New York: Springer. 41
- Kemeny, J. G. and J. L. Snell 1976. *Finite Markov Chains*. New York: Springer.
- Kennedy, P. 2008. *A Guide to Econometrics* (sixth ed.). New York: Wiley.
- Keri, J. (ed.) 2006. *Baseball Between the Numbers: Why Everything You Know About the Game is Wrong*. New York: Basic Books. 65
- Késenne, S. 2007. *The Economic Theory of Professional Team*

Sports: An Analytical Treatment. Cheltenham, U.K.:

Edward Elgar.

Kleiber, C. and A. Zeileis 2008. *Applied Econometrics with R*.
New York: Springer.

Knight, M. 2015. *The Best Team Money Can Buy: The Los Angeles Dodgers' Wild Struggle to Build a Baseball Powerhouse*. New York: Simon & Schuster. 111

Koller, M. 2014. *Simulations for Sharpening Wald-type Inference in Robust Regression for Small Samples*.

Comprehensive R Archive Network. 2014.

http://cran.r-project.org/web/packages/robustbase/vignettes/lmrob_simulation.pdf. 214

Koller, M. and W. A. Stahel 2011. Sharpening Wald-type inference in robust regression for small samples.
Computational Statistics and Data Analysis 55(8):2504–2515. 214

Konik, M. 2006. *The Smart Money: How the World's Best Sports Bettors Beat the Bookies Out of Millions*. New York: Simon & Schuster.

Kotler, P. and K. L. Keller 2012. *Marketing Management* (fourteenth ed.). Upper Saddle River, N.J.: Prentice Hall. 111, 121, 131

Krippendorff, K. H. 2012. *Content Analysis: An Introduction to Its Methodology* (third ed.). Thousand Oaks, Calif.: Sage. 223

Krishnamurthi, L. 2001. Pricing strategies and tactics. In D. Iacobucci (ed.), *Kellogg on Marketing*, Chapter 12, pp. 279–301. New York: Wiley.

- Kuhn, M. and K. Johnson 2013. *Applied Predictive Modeling*. New York: Springer.
- Kutner, M. H., C. J. Nachtsheim, J. Neter, and W. Li 2004. *Applied Linear Statistical Models* (fifth ed.). Boston: McGraw-Hill.
- Ladany, S. P. and R. E. Machol (eds.) 1977. *Optimal Strategies in Sports*. New York: North-Holland.
- Lamp, G. 2014. *ggplot for Python*. GitHub. 2014. <https://github.com/yhat/ggplot>. 238
- Lander, J. P. 2014. *R for Everyone: Advanced Analytics and Graphics*. Upper Saddle River, N.J.: Pearson Education.
- Langville, A. N. and C. D. Meyer 2012. *Who's 1?: The Science of Rating and Ranking*. Princeton, N.J.: Princeton University Press.
- Lantz, B. 2013. *Machine Learning with R*. Birmingham, U.K.: Packt Publishing.
- Laursen, G. H. N. and J. Thorlund 2010. *Business Analytics for Managers: Taking Business Intelligence Beyond Reporting*. Hoboken, N.J.: Wiley. 199
- Law, A. M. 2014. *Simulation Modeling and Analysis* (fifth ed.). New York: McGraw-Hill.
- Le, C. T. 1997. *Applied Survival Analysis*. New York: Wiley.
- Le, C. T. 1998. *Applied Categorical Data Analysis*. New York: Wiley.
- Ledoiter, J. 2013. *Data Mining and Business Analytics with R*. New York: Wiley.
- Lee, M. 1981. *Betting the Bases*. San Clemente, Calif.: Mike Lee.

- Leeds, M. A. and P. von Allmen 2014. *The Economics of Sports* (fifth ed.). Upper Saddle River, N.J.: Pearson.
- Leeflang, P. S. H., D. R. Wittink, M. Wedel, and P. A. Naert 2000. *Building Models for Marketing Decisions*. Boston: Kluwer.
- Leetaru, K. 2011. *Data Mining Methods for Content Analysis: An Introduction to the Computational Analysis of Content*. New York: Routledge. 223
- Lefkowitz, M. 2014. Twisted documentation. Contributors listed as Twisted Matrix Laboratories at <http://twistedmatrix.com/trac/wiki/TwistedMatrixLaboratories>. Documentation available at <http://twistedmatrix.com/trac/wiki/Documentation>.
- Leisch, F. and B. Gruen 2014. *CRAN Task View: Cluster Analysis & Finite Mixture Models*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/views/Cluster.html>.
- Lemke, R. J., M. Leonard, and K. Thokwane 2010. Estimating attendance at major league baseball games for the 2007 season. *Journal of Sports Economics* 11(3):316–348. 112
- Lewin-Koh, N. 2014. *Hexagon Binning: an Overview*. Comprehensive R Archive Network. 2014. http://cran.r-project.org/web/packages/hexbin/vignettes/hexagon_binning.pdf. 238
- Lewis, M. 2003. *Moneyball: The Art of Winning an Unfair Game*. New York: W. W. Norton & Company. 31
- Lewis, M. 2007. *The Blind Side*. New York: W. W. Norton.

- Lilien, G. L., P. Kotler, and K. S. Moorthy 1992. *Marketing Models*. Englewood Cliffs, N.J.: Prentice-Hall.
- Lilien, G. L. and A. Rangaswamy 2003. *Marketing Engineering: Computer-Assisted Marketing Analysis and Planning* (second ed.). Upper Saddle River, N.J.: Prentice Hall.
- Lindsey, G. 1963. An investigation of strategies in baseball. *Operations Research* 11(4):477–501.
- Lindsey, J. K. 1997. *Applying Generalized Linear Models*. New York: Springer.
- Little, R. J. A. and D. B. Rubin 1987. *Statistical Analysis with Missing Data*. New York: Wiley. 183
- Liu, B. 2010. Sentiment analysis and subjectivity. In N. Indurkhy and F. J. Damerau (eds.), *Handbook of Natural Language Processing* (second ed.), pp. 627–665. Boca Raton, Fla.: Chapman and Hall/CRC.
- Liu, B. 2011. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. New York: Springer. 216
- Liu, B. 2012. *Sentiment Analysis and Opinion Mining*. San Rafael, Calif.: Morgan & Claypool.
- Lloyd, C. J. 1999. *Statistical Analysis of Categorical Data*. New York: Wiley.
- Lodish, L. M., M. M. Abraham, J. Livesberger, B. Lubetkin, B. Richardson, and M. E. Stevens 1995. A summary of fifty-five in-market experimental estimates of the long-term effect of advertising. *Marketing Science* 14(3):G133–G140.
- Lord, F. M. and M. R. Novick 1968. *Statistical Theories of Mental Test Scores*. Reading, Mass.: Addison-Wesley. 35
- Lowenfish, L. 2010. *The Imperfect Diamond: A History of*

- Baseball's Labor Wars*. Lincoln, Neb.: University of Nebraska Press.
- Lubanovic, B. 2015. *Introducing Python: Modern Computing in Simple Packages*. Sebastopol, Calif.: O'Reilly.
- Luce, D. and J. Tukey 1964. Simultaneous conjoint measurement: A new type of fundamental measurement. *Journal of Mathematical Psychology* 1:1–27.
- Luger, G. F. 2008. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* (sixth ed.). Boston: Addison-Wesley. 216
- Lumley, T. 2010. *Complex Surveys: A Guide to Analysis Using R*. New York: Wiley. 183
- Lumley, T. 2014. *mitools: Tools for Multiple Imputation of Missing Data*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/mitools/mitools.pdf>.
- Lyon, D. W. 2000. Pricing research. In C. Chakrapani (ed.), *Marketing Research: State-of-the-Art Perspectives*, Chapter 19, pp. 551–582. Chicago: American Marketing Association.
- Lyon, D. W. 2002, Winter. The price is right (or is it)? *Marketing Research*:8–13.
- Lyons, R. S. 2002. *Palestra Pandemonium: A History of the Big Five*. Philadelphia: Temple University Press.
- Maechler, M. 2014a. *Package cluster*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/cluster/cluster.pdf>.

- Maechler, M. 2014b. *robustbase: Basic Robust Statistics*. Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/robustbase/robustbase.pdf>. 214
- Mahoney, B. 2015, July 8. NBA salary cap for 2015–16 rises to \$70 million. Retrieved from the World Wide Web on August 4, 2015 at
<http://www.nba.com/2015/news/07/08/ap-nba-salary-cap-rises-to-70-million.ap/>. 4
- Maisel, L. S. and G. Cokins 2014. *Predictive Business Analytics: Forward-Looking Capabilities to Improve Business Performance*. New York: Wiley. 199
- Major League Baseball 2015a. *Official Rules of Major League Baseball*. Chicago: Triumph Books.
- Major League Baseball 2015b. Transactions.
<http://mlb.mlb.com/mlb/transactions/>
- Makridakis, S., S. C. Wheelwright, and R. J. Hyndman 2005. *Forecasting Methods and Applications* (third ed.). New York: Wiley.
- Manly, B. F. J. 1994. *Multivariate Statistical Methods: A Primer* (second ed.). London: Chapman & Hall.
- Manning, C. D. and H. Schütze 1999. *Foundations of Statistical Natural Language Processing*. Cambridge: MIT Press.
- Manteris, A. 1991. *SuperBookie: Inside Las Vegas Sports Gambling*. Chicago: Contemporary Books.
- Marchi, M. and J. Albert 2014. *Analyzing Baseball Data with R*. Boca Raton, Fla.: Chapman & Hall/CRC. 34

- Marder, E. 1997. *The Laws of Choice: Predicting Consumer Behavior*. New York: Free Press.
- Maronna, R. A., D. R. Martin, and V. J. Yohai 2006. *Robust Statistics Theory and Methods*. New York: Wiley. 214
- Martin, L. 2016. *Sports Performance Measurement and Analytics: The Science of Assessing Performance, Predicting Future Outcomes, Interpreting Statistical Models, and Evaluating the Market Value of Athletes*. Old Tappan, N.J.: Pearson Education.
- <http://www.ftpress.com/martin>. vii Martin, L. and T. W. Miller in press. *A Model for Measurement in Sports*. Manhattan Beach, Calif.: Research Publishers.
- <http://www.research-publishers.com/>.
- Matloff, N. 2011. *The Art of R Programming*. San Francisco: no starch press.
- Maybury, M. T. (ed.) 1997. *Intelligent Multimedia Information Retrieval*. Menlo Park, Calif./Cambridge: AAAI Press / MIT Press.
- McCallum, Q. E. (ed.) 2013. *Bad Data Handbook*. Sebastopol, Calif.: O'Reilly.
- McCullagh, P. and J. A. Nelder 1989. *Generalized Linear Models* (second ed.). New York: Chapman and Hall.
- McDonagh, E. and L. Pappano 2008. *Playing with the Boys: Why Separate is Not Equal in Sports*. New York: Oxford University Press.
- McGrayne, S. B. 2011. *The Theory that Would Not Die: How Bayes' Rule Cracked the Enigma Code, Hunted Down Russian Submarines and Emerged Triumphant from Two Centuries of Controversy*. New Haven, Conn.: Yale

University Press. 205

McGuire, F. 1958. *Offensive Basketball*. Englewood Cliffs, N.J.: Prentice Hall.

McGuire, F. 1960. *Defensive Basketball*. Englewood Cliffs, N.J.: Prentice Hall.

Meadow, C. T., B. R. Boyce, and D. H. Kraft 2000. *Text Information Retrieval Systems* (second ed.). San Diego: Academic Press.

Mehrens, W. A. and R. L. Ebel (eds.) 1967. *Principles of Educational and Psychological Measurement: A Book of Selected Readings*. Chicago: Rand McNally.

Merkl, D. 2002. Text mining with self-organizing maps. In W. Klösgen and J. M. Zytkow (eds.), *Handbook of Data Mining and Knowledge Discovery*, Chapter 46.9, pp. 903–910. Oxford: Oxford University Press.

Michalawicz, Z. and D. B. Fogel 2004. *How to Solve It: Modern Heuristics*. New York: Springer. 216

Milgram, S. 1967. The small world problem. *Psychology Today* 1(1):60–67. 231

Miller, J. E. 1990. *The Baseball Business: Pursuing Pennants and Profits in Baltimore*. Chapel Hill, N.C.: University of North Carolina Press. 13

Miller, J. H. and S. E. Page 2007. *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*. Princeton, N. J.: Princeton University Press.

Miller, M. 1991. *A Whole Different Ball Game: The Sport and Business of Baseball*. New York: Bitch Lane Press/Carol Publishing Group.

Miller, S. and J. Wojciechowski (eds.) 2015. *Baseball*

Prospectus 2015: The Essential Guide to the 2015 Season.

New York: Wiley. 138

Miller, T. W. 2005. *Data and Text Mining: A Business Applications Approach*. Upper Saddle River, N.J.: Pearson Prentice Hall.

Miller, T. W. 2008. *Without a Tout: How to Pick a Winning Team*. Manhattan Beach, Calif.: Research Publishers.

Miller, T. W. 2015a. *Marketing Data Science: Modeling Techniques in Predictive Analytics with R and Python*. Old Tappan, N.J.: Pearson Education. Data sets and programs available at <http://www.ftpress.com/miller/> and <https://github.com/mtpa/>.

Miller, T. W. 2015b. *Modeling Techniques in Predictive Analytics with Python and R: A Guide to Data Science*. Upper Saddle River, N.J.: Pearson Education. Data sets and programs available at
<http://www.ftpress.com/miller/> and
<https://github.com/mtpa/>.

Miller, T. W. 2015c. *Modeling Techniques in Predictive Analytics: Business Problems and Solutions with R* (revised and expanded ed.). Upper Saddle River, N.J.: Pearson Education. Data sets and programs available at
<http://www.ftpress.com/miller/> and
<https://github.com/mtpa/>.

Miller, T. W. 2015d. *Web and Network Data Science: Modeling Techniques in Predictive Analytics with Python and R*. Upper Saddle River, N.J.: Pearson Education. Data sets and programs available at
<http://www.ftpress.com/miller/> and

<https://github.com/mtpa/>.

- Millman, C. 2001. *The Odds: One Season, Three Gamblers, and the Death of Their Las Vegas*. Cambridge, Mass.: Da Capo Press.
- Mitchell, M. 1996. *An Introduction to Genetic Algorithms*. Cambridge: MIT Press. 216
- Mitchell, M. 2009. *Complexity: A Guided Tour*. Oxford, U.K.: Oxford University Press. 166
- Mitchell, R. 2015. *Web Scraping with Python: Collecting Data from the Modern Web*. Sebastopol, Calif.: O'Reilly.
- Moran, M. and B. Hunt 2009. *Search Engine Marketing, Inc.: Driving Search Traffic to Your Company's Web Site* (second ed.). Boston: IBM Press/Pearson Education.
- Moreno, J. L. 1934. Who shall survive?: Foundations of sociometry, group psychotherapy, and sociodrama. Reprinted in 1953 (second edition) and in 1978 (third edition) by Beacon House, Inc., Beacon, N.Y. 230
- Morville, P. and J. Callender 2010. *Search Patterns*. Sebastopol, Calif.: O'Reilly.
- Moskowitz, T. J. and L. J. Wertheim 2011. *Scorecasting: The Hidden Influences Behind How Sports Are Played and Games Are Won*. New York: Crown Archetype. 67
- Mosteller, F. 1965. *Fifty Challenging Problems in Probability with Solutions*. Reading, Mass.: Addison-Wesley.
- Mosteller, F. 1997. Lessons from sports statistics. *The American Statistician* 51(4):305–310.
- Mosteller, F., R. E. K. Rourke, and G. B. Thomas, Jr. 1970. *Probability with Statistical Applications* (second ed.).

Reading, Mass.: Addison-Wesley.

Mosteller, F. and D. L. Wallace 1984. *Applied Bayesian and Classical Inference: The Case of “The Federalist” Papers* (second ed.). New York: Springer. Earlier edition published in 1964 by Addison-Wesley, Reading, Mass. The previous title was *Inference and Disputed Authorship: The Federalist*.

Mullin, B. J., S. Hardy, and W. A. Sutton 2014. *Sport Marketing* (fourth ed.). Champaign, Ill.: Human Kinetics.

Murphy, K. P. 2012. *Machine Learning: A Probabilistic Perspective*. Cambridge, Mass.: MIT Press. 205, 216

Murray, M. 2007. *Betting Baseball 2007*.

www.BettingBaseball.net: Michael Murray.

Murrell, P. 2011. *R Graphics* (second ed.). Boca Raton, Fla.: CRC Press.

Nagle, T. T. and J. Hogan 2005. *The Strategy and Tactics of Pricing: A Guide to Growing More Profitably* (fourth ed.). Upper Saddle River, N.J.: Prentice Hall.

Nash, E. L. 1995. *Direct Marketing: Strategy/Planning/Execution* (third ed.). New York: McGraw-Hill.

Nash, E. L. (ed.) 2000. *The Direct Marketing Handbook* (second ed.). New York: McGraw-Hill.

National Basketball Association 2015. 2015 NBA Finals.

<http://www.nba.com/playoffs/2015/finals/>.

National Sporting Goods Association 2015. NSGA sports participation in the united states (2015 edition).
<https://www.nsga.org/research/nsga-research-offerings/sports-participation->

us - 2015/.

- Neal, W. D. 2000. Market segmentation. In C. Chakrapani (ed.), *Marketing Research: State-of-the-Art Perspectives*, Chapter 1, pp. 375–399. American Marketing Association.
- Nelson, W. B. 2003. *Recurrent Events Data Analysis for Product Repairs, Disease Recurrences, and Other Applications*. Series on Statistics and Applied Probability. Philadelphia and Alexandria, Va.: ASA-SIAM.
- Nolan, D. and D. T. Lang 2014. *XML and Web Technologies for Data Sciences with R*. New York: Springer.
- North, M. J. and C. M. Macal 2007. *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford, U.K.: Oxford University Press.
- Null and Kaval 2001. *The Summer that Saved Baseball: The Ultimate Fan's Road Trip*. Nashville, Tenn.: Cumberland House. Documents visits to thirty MLB ballparks.
- Nunnally, J. C. 1967. *Psychometric Theory*. New York: McGraw-Hill. 35
- Nunnally, J. C. and I. H. Bernstein 1994. *Psychometric Theory* (third ed.). New York: McGraw-Hill. 35
- Obe, R. and L. Hsu 2012. *PostgreSQL up and Running: A Practical Guide to the Advanced Open Source Database*. Sebastopol, Calif.: O'Reilly.
- O'Hagan, A. 2010. *Kendall's Advanced Theory of Statistics: Bayesian Inference*, Volume 2B. New York: Wiley. 205
- Okrent, D. 1985. *Nine Innings*. New York: Ticknor & Fields.
- Okrent, D. and S. Wulf 1989. *Baseball Anecdotes*. New York:

- Oxford University Press.
- Oliver, D. 2004. *Basketball on Paper: Rules and Tools of Performance Analysis*. Dulles, Va.: Brassey's Press. 34
- Orme, B. K. 2006. *Getting Started with Conjoint Analysis: Strategies for Product Design and Pricing Research*. Madison, Wis.: Research Publishers LLC.
- Orme, B. K. 2013. *Getting Started with Conjoint Analysis: Strategies for Product Design and Pricing Research* (third ed.). Glendale, Calif.: Research Publishers LLC.
<http://research-publishers.com/rp/gsca.htm>. 79
- Osborne, J. W. 2013. *Best Practices in Data Cleaning: A Complete Guide to Everything You Need to Do Before and After Collecting Your Data*. Los Angeles: Sage.
- Osgood, C. 1962. Studies in the generality of affective meaning systems. *American Psychologist* 17:10–28. 225
- Osgood, C., G. Suci, and P. Tannenbaum (eds.) 1957. *The Measurement of Meaning*. Urbana, Ill.: University of Illinois Press. 225
- Ozanian, M. 2015, May 6. The world's most valuable soccer teams. *Forbes*. Retrieved from the World Wide Web, August 3, 2015, at
<http://www.forbes.com/soccer-valuations/>.
- Pankin, M. D. 1978, July–August. Evaluating offensive performance in baseball. *Operations Research* 26(4):610–619.
- Parker, R. J. 2010, May 5. Modeling basketball's points per possession with application to predicting the outcome of college basketball games. Retrieved from the World Wide

Web on August 4, 2015 at
http://www.basketballgeek.com/downloads/ryan_bach_essay.pdf/ with NBA season play-by-play data at
<http://www.basketballgeek.com/data/>.

Patra, K. 2015, March 2. NFL salary cap will be \$143.28 million in 2015. Retrieved from the World Wide Web on August 4, 2015 at
<http://www.nfl.com/news/story/0ap3000000475775/article/nfl-salary-cap-will-be-14328-million-in-2015/>. 4

Peppers, D. and M. Rogers 1993. *The One to One Future: Building Relationships One Customer at a Time*. New York: Doubleday.

Pessah, J. 2015. *The Game: Inside the Secret World of Major League Baseball's Power Brokers*. New York: Little Brown and Company. 13

Petris, G. 2010, October 13. An R package for dynamic linear models. *Journal of Statistical Software* 36(12):1–16.
<http://www.jstatsoft.org/v36/i12>.

Petris, G. and W. Gilks 2014. *dlm: Bayesian and Likelihood Analysis of Dynamic Linear Models*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/dlm/dlm.pdf>.

Petris, G., S. Petrone, and P. Campagnoli 2009. *Dynamic Linear Models with R*. New York: Springer.

Phillips, R. L. 2005. *Pricing and Revenue Management*. Stanford, Calif.: Stanford University Press. 131

Pindyck, R. S. and D. L. Rubinfeld 2000. *Econometric Models*

- and Economic Forecasts* (fourth revised ed.). New York:
McGraw-Hill. 131
- Pindyck, R. S. and D. L. Rubinfeld 2004. *Microeconomics*
(sixth ed.). Upper Saddle River, N.J.: Prentice-Hall. 131
- Pinker, S. 1994. *The Language Instinct*. New York: W.
Morrow and Co.
- Pinker, S. 1997. *How the Mind Works*. New York: W.W.
Norton & Company.
- Pinker, S. 1999. *Words and Rules: The Ingredients of
Language*. New York: HarperCollins.
- Pollard, R. 1973, June. Collegiate football scores and the
negative binomial distribution. *Journal of the American
Statistical Association* 68(342):351–352. 156
- Popping, R. 2000. *Computer-Assisted Text Analysis*. Thousand
Oaks, Calif.: Sage. 223
- Porter, M. E. 1980. *Competitive Strategy: Techniques for
Analyzing Industries and Competitors*. New York: The Free
Press.
- Powers, A. T. 2003. *The Business of Baseball*. Jefferson, N.C.:
McFarland & Company. 13
- Privault, N. 2013. *Understanding Markov Chains: Examples
and Applications*. New York: Springer.
- Provost, F. and T. Fawcett 2014. *Data Science for Business:
What You Need to Know About Data Mining and Data-
Analytic Thinking*. Sebastopol, Calif.: O'Reilly. 199
- Pustejovsky, J. and A. Stubbs 2013. *Natural Language
Annotation for Machine Learning*. Sebastopol, Calif.:
O'Reilly. 35

- Puterman, M. L. 2005. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: Wiley.
- Putler, D. S. and R. E. Krider 2012. *Customer and Business Analytics: Applied Data Mining for Business Decision Making Using R*. Boca Raton, Fla: Chapman & Hall/CRC.
- Quirk, J. and R. D. Fort 1992. *Pay Dirt: The Business of Professional Team Sports*. Princeton, N.J.: Princeton University Press.
- Radcliffe-Brown, A. R. 1940. On social structure. *Journal of the Royal Anthropological Society of Great Britain and Ireland* 70:1–12. 230
- Raiffa, H. 1968. *Decision Analysis: Introductory Lectures on Choices Under Uncertainty*. London: Longman Higher Education.
- Rajaraman, A. and J. D. Ullman 2012. *Mining of Massive Datasets*. Cambridge, UK: Cambridge University Press.
- Reep, C., R. Pollard, and B. Benjamin 1971. Skill and chance in ball games. *Journal of the Royal Statistical Society, Series A (General)* 134(4):623–629. 156
- Rein, I., P. Kotler, and B. Shields 2006. *The Elusive Fan: Reinventing Sports in a Crowded Marketplace*. New York: McGraw-Hill.
- Rein, I., B. Shields, and A. Grossman 2015. *The Sports Strategist: Developing Leaders for a High-Performance Industry*. New York: Oxford University Press. 12
- Rencher, A. C. and G. B. Schaalje 2008. *Linear Models in Statistics* (second ed.). New York: Wiley.
- Resnick, M. 1998. *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*.

- Cambridge, Mass.: MIT Press. 166
- Revelle, W. 2014. *psych: Procedures for Psychological, Psychometric, and Personality Research*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/psych/psych.pdf>.
- Richardson, M. 1938. Multidimensional psychophysics. *Psychological Bulletin* 35:659–660. 81
- Ripken, C. and B. Ripken 2004. *Play Baseball the Ripken Way: The Complete Illustrated Guide to the Fundamentals*. New York: Ballantine Books.
- Ripken, C. and B. Ripken 2007. *Coaching Youth Baseball the Ripken Way*. Champaign, Ill.: Human Kinetics.
- Ripley, B., W. Venables, D. M. Bates, K. Hornik, A. Gebhardt, and D. Firth 2015. *Package MASS: Functions and Datasets to Support Venables and Ripley “Modern Applied Statistics with S” (4th edition 2002)*. Cambridge, USA: Comprehensive R Archive Network. 2015. <http://cran.r-project.org/web/packages/MASS/MASS.pdf>.
- Robbins, J. N. 2003. *Learning Web Design: A Beginner’s Guide to HTML, CSS, JavaScript, and Web Graphics* (fourth ed.). Sebastopol, Calif.: O’Reilly.
- Robert, C. P. 2007. *The Bayesian Choice: From Decision Theoretic Foundations to Computational Implementation* (second ed.). New York: Springer. 205
- Robert, C. P. and G. Casella 2009. *Introducing Monte Carlo Methods with R*. New York: Springer. 205
- Roberts, C.W. (ed.) 1997. *Text Analysis for the Social Sciences: Methods for Drawing Statistical Inferences from*

- Texts and Transcripts*. Mahwah, N.J.: Lawrence Erlbaum Associates. 223
- Robinson, I., J. Webber, and E. Eifrem 2013. *Graph Databases*. Sebastopol, Calif.: O'Reilly.
- Rogers, H. . J., H. Swaminathan, and R. K. Hambleton 1991. *Fundamentals of Item Response Theory*. Newbury Park, Calif.: Sage.
- Roman, A. 2004. *The Art of Betting Baseball*. www.booksurge.com: BookSurge LLC.
- Romer, D. 2006. Do firms maximize? Evidence from professional football. *Journal of Political Economy* 114(2):340–365. 67
- Root, W. A. 1989. *Betting to Win on Sports*. New York: Bantam Books.
- Ross, K. 2004. *Mathematician at the Ballpark: Odds and Probabilities for Baseball Fans*. New York: PI Press.
- Ross, S. M. 2014. *Introduction to Probability Models* (eleventh ed.). New York: Academic Press.
- Rossi, P. 2014. *bayesm: Bayesian Inference for Marketing/Micro-econometrics*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/bayesm/bayesm.pdf>.
- Rossi, P. E., G. M. Allenby, and R. McCulloch 2005. *Bayesian Statistics and Marketing*. New York: Wiley.
- Rousseeuw, P. J. 1987. Silhouettes:A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 20:53–65.

- Rubin, D. B. 1987. *Multiple Imputation for Nonresponse in Surveys*. New York: Wiley. 183
- Ruggiero, J., L. Hadley, G. Ruggiero, and S. Knowles 1997. A note on the Pythagorean Theorem of baseball production. *Managerial and Decision Economics* 18:335–342.
- Runquist, W. 1995. *Baseball by the Numbers: How Statistics Are Collected, What They Mean, and How They Reveal the Game*. Jefferson, N.C.: McFarland & Company.
- Russell, M. A. 2014. *Mining the SocialWeb: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More*. Sebastopol, Calif.: O'Reilly. Python code is available at <https://github.com/ptwobrussell/Mining-the-SocialWeb-2nd-Edition>.
- Russell, S. and P. Norvig 2009. *Artificial Intelligence: A Modern Approach* (third ed.). Upper Saddle River, N.J.: Prentice Hall. 216
- Russell-Rose, T. and TylerTate 2013. *Designing the Search Experience: The Information Architecture of Discovery*. Waltham, Mass.: Elsevier/Morgan Kaufmann.
- Ryan, J. A. 2014. *Package quantmod: Quantitative Financial Modelling Framework*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/quantmod/quantmod.pdf>.
- Ryan, T. P. 2008. *Modern Regression Methods* (second ed.). New York: Wiley.
- Šalamon, T. 2011. *Design of Agent-Based Models: Developing Computer Simulations for a Better Understanding of Social Processes*. Czech Republic: Tomáš Bruckner, Řepín-

Živonín.

- Salton, G., A. Wong, and C. S. Yang 1975. A vector space model for automatic indexing. *Journal of Computational and Applied Mathematics* 18(11):613–620.
- Sarkar, D. 2008. *Lattice: Multivariate Data Visualization with R*. New York: Springer.
- Sarkar, D. 2014. *lattice: Lattice Graphics*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/lattice/lattice.pdf>.
- Sauer, R. D. 1998, December. The economics of wagering markets. *Journal of Economic Literature* 36:2021–2064.
- Sawchik, T. 2015. *Big Data Baseball: Math, Miracles, and the End of a 20-Year Losing Streak*. New York: Flatiron Books.
- Schafer, J. L. 1997. *Analysis of Incomplete Multivariate Data*. London: Chapman & Hall. 183
- Schauerhuber, M., A. Zeileis, D. Meyer, and K. Hornik 2008. Benchmarking open-source tree learners in R/RWeka. In C. Preisach, H. Burkhardt, L. Schmidt-Thieme, and R. Decker (eds.), *Data Analysis, Machine Learning, and Applications*, pp. 389–396. New York: Springer.
- Schlough, B. 2015. In-Stadium Sports Fan Experience Summit: Speaker Interview. Retrieved from the World Wide Web on October 4, 2015 at
<http://www.q1productions.com/speaker-interview-bill-schlough/>. 130
- Schnettler, S. 2009. A structured overview of 50 years of small-world research. *Social Networks* 31:165–178. 231
- Schwarz, A. 2004. *The Numbers Game: Baseball's Lifelong*

Fascination with Statistics. New York: St. Martin's Griffin.

65

Schwertman, N. C., T. A. McCready, and L. Howard 1991, February. Probability models for the NCAA regional basketball tournaments. *The American Statistician* 45(1):35–38. 42

Schwertman, N. C., K. L. Schenk, and B. C. Holbrook 1996, February. More probability models for the NCAA regional basketball tournaments. *The American Statistician* 50(1):34–38. 42

Seber, G. A. F. 2000. *Multivariate Observations*. New York: Wiley. Originally published in 1984.

Seidel, L. R. 2004. *Investing in College Basketball*. Bloomington, Ind.: AuthorHouse.

Sermas, R. 2014. *ChoiceModelR: Choice Modeling in R*. Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/ChoiceModelR/ChoiceModelR.pdf>.

Sharda, R. and D. Delen 2006. Predicting box office success of motion pictures with neural networks. *Expert Systems with Applications* 30:243–254. 224

Sharma, S. 1996. *Applied Multivariate Techniques*. New York: Wiley.

Shea, S. M. 2014. *Basketball Analytics: Spatial Tracking*. CreateSpace.

Shea, S. M. and C. E. Baker 2013. *Basketball Analytics: Objective and Efficient Strategies for Understanding How Teams Win*. Lake St. Louis, Mo.: Advanced Metrics, LLC.

- Shmueli, G. 2010. To explain or predict? *Statistical Science* 25(3):289–310.
- Shrout, P. E. and S. T. Fiske (eds.) 1995. *Personality Research, Methods, and Theory: A Festschrift Honoring Donald W. Fiske*. Hillsdale, N.J.: Lawrence Erlbaum Associates.
- Shumaker, R. P., O. K. Solieman, and H. Chen 2010. *Sports Data Mining*. New York: Springer.
- Siegel, E. 2013. *Predictive Analytics: The Power to Predict Who Will Click, Buy, Lie, or Die*. Hoboken, N.J.: Wiley. 199
- Silver, N. 2004, March 11. Baseball prospectus basics: the science of forecasting. *Baseball Prospectus*. Retrieved from the World Wide Web, September 27, 2015, at <http://www.baseballprospectus.com/article.php?articleid=2659/>.
- Silver, N. 2012. *The Signal and the Noise: Why So Many Predictions Fail—But Some Don’t*. New York: The Penguin Press.
- Sing, T., O. Sander, N. Beerenwinkel, and T. Lengauer 2005. ROCR: Visualizing classifier performance in R. *Bioinformatics* 21(20):3940–3941.
- Sirona, J. 2014, June 24. Structured Optimization Modeling with Pyomo and Coopr. Presentation retrieved from the World Wide Web on September 24, 2015 at <https://m.youtube.com/watch?v=cjMkVHjhSBI/>.
- Slatkin, B. 2015. *Effective Python: 59 Specific Ways to Write Better Python*. Upper Saddle River, N.J.: Pearson/Addison-Wesley.

- Snedecor, G. W. and W. G. Cochran 1989. *Statistical Methods* (eighth ed.). Ames, Iowa: Iowa State University Press. First edition published by Snedecor in 1937. 205
- Snijders, T. A. B. and R. J. Bosker 2012. *Multilevel Analysis: An Introduction to Basic and Advanced Multilevel Modeling* (second ed.). Thousand Oaks, Calif.: Sage. 183
- Socher, R., J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Sports Reference LLC 2015a. 2014 Major League Baseball team statistics and standings: team & league standard batting. Retrieved from the World Wide Web on September 7, 2015 at <http://www.baseball-reference.com/leagues/MLB/2014-standings.shtml/>. 32, 281, 289, 294
- Sports Reference LLC 2015b. MLB detailed statistics. Retrieved from the World Wide Web on August 3, 2015 at <http://www.baseball-reference.com/leagues/MLB/2014-standings.shtml/>.
- spotrac 2015a. MLB contracts. Retrieved from the World Wide Web on August 5, 2015 at <http://www.spotrac.com/mlb/contracts/>. 4
- spotrac 2015b. NBA contracts. Retrieved from the World Wide Web on August 5, 2015 at <http://www.spotrac.com/nba/contracts/>. 4
- spotrac 2015c. NFL contracts. Retrieved from the World Wide

Web on August 5, 2015 at
<http://www.spotrac.com/nfl/contracts/>. 4

Srivastava, A. N. and M. Sahami (eds.) 2009. *Text Mining: Classification, Clustering, and Applications*. Boca Raton, Fla.: CRC Press.

Stallings, J. and B. Bennett (eds.) 2003. *Baseball Strategies: Your Guide to the Game Within the Game*. Champaign, Ill.: Human Kinetics.

statista 2015. Tv viewership of the super bowl in the united states from 1990 to 2015 (in millions). Retrieved from the World Wide Web on August 5, 2015 at
<http://www.statista.com/statistics/21652/6/super-bowl-us-tv-viewership/>. 9

Stephen, E. 2014, January 15. Clayton Kershaw Contract Breakdown. Retrieved from the World Wide Web on September 20, 2015 at
<http://www.truebluela.com/2014/1/15/5313450/clayton-kershaw-contract-detials/>.

Stern, H. S. 1991, December. On the probability of winning a football game. *The American Statistician* 45(3):179–183. 42

Sternthal, B. and A. M. Tybout 2001. Segmentation and targeting. In D. Iacobucci (ed.), *Kellogg on Marketing*, Chapter 1, pp. 3–30. New York: Wiley.

Stevens, S. S. 1946, June 7. On the theory of scales of measurement. *Science* 103(2684): 677–680.
<http://www.jstor.org/stable/1671815>. 28

Stokey, E. and R. Zeckhauser 1978. *A Primer for Policy Analysis*. New York: W. W. Norton & Co. 142

Stone, P. J. 1997. Thematic text analysis: New agendas for

- analyzing text content. In C. W. Roberts (ed.), *Text Analysis for the Social Sciences: Methods for Drawing Statistical Inferences from Texts and Transcripts*, Chapter 2, pp. 35–54. Mahwah, N.J.: Lawrence Erlbaum Associates.
- Stone, P. J., D. C. Dunphy, M. S. Smith, and D. M. Ogilvie 1966. *The General Inquirer: A Computer Approach to Content Analysis*. Cambridge: MIT Press.
- Streeter, K. 2015, October 12. The Owners Issue: Chris Paul Will Not Be Played. *ESPN*: 67–72. 130
- Stuart, A., K. Ord, and S. Arnold 2010. *Kendall's Advanced Theory of Statistics: Classical Inference and the Linear Model*, Volume 2A. New York: Wiley. 205
- Suess, E. A. and B. E. Trumbo 2010. *Introduction to Probability Simulation and Gibbs Sampling with R*. New York: Springer. 205
- Sugar, B. R. 1992. *The Caesars Palace Sports Book of Betting*. New York: St. Martin's Press.
- Summerfield, M. 2012. *Programming in Go: Creating Applications for the 21st Century*. Upper Saddle River, N.J.: Pearson/Addison-Wesley.
- Sweigart, A. 2015. *Automate the Boring Stuff with Python: Practical Programming for Total Beginners*. San Francisco: No Starch Press.
- Szymanski, C. 2014. *dlmodeler: Generalized Dynamic Linear Modeler*. Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/dlmodeler/dlmodeler.pdf>
- Szymanski, S. 2009. *Playbooks and Checkbooks: An*

Introduction to the Economics of Modern Sports. Princeton, N.J.: Princeton University Press.

Szymanski, S. 2015. *Money and Soccer: A Soccernomics Guide*. New York: Nation Books.

Taddy, M. 2013a. Measuring political sentiment on Twitter: factor-optimal design for multinomial inverse regression. Retrieved from the World Wide Web at
<http://arxiv.org/pdf/1206.3776v5.pdf>.

Taddy, M. 2013b. Multinomial inverse regression for text analysis. Retrieved from the World Wide Web at
<http://arxiv.org/pdf/1012.2098v6.pdf>.

Taddy, M. 2014. *textir: Inverse Regression for Text Analysis*. 2014. <http://cran.r-project.org/web/packages/textir/textir.pdf>.

Talluri, K. T. and G. J. van Ryzin 2004. Revenue management under a general discrete choice model of consumer behavior. *Management Science* 50(1):15–33.

Tan, P.-N., M. Steinbach, and V. Kumar 2006. *Introduction to Data Mining*. Boston: Addison-Wesley.

Tang, W., H. He, and X. M. Tu 2012. *Applied Categorical and Count Data Analysis*. Boca Raton, Fla.: Chapman & Hall/CRC.

Tango, T. M., M. G. Lichtman, and A. E. Dolphin 2007. *The Book: Playing the Percentages in Baseball*. Dulles, Va.: Potomac Books. 65

Tannenbaum, A. S. and D. J. Wetherall 2010. *Computer Networks* (fifth ed.). Upper Saddle River, N.J.: Pearson/Prentice Hall.

- Tanner, M. A. 1996. *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions* (third ed.). New York: Springer. 205
- Thaler, R. H. 1994. *The Winner's Curse: Paradoxes and Anomalies of Economic Life*. Princeton, N.J.: Princeton University Press. 142
- Thaler, R. H. and C. R. Sunstein 2004. Market efficiency and rationality: the peculiar case of baseball. *Michigan Law Review* 102:1390–1403.
- Thaler, R. H. and C. R. Sunstein 2009. *Nudge: Improving Decisions About Health, Wealth, and Happiness*. New York: Penguin Books. 142
- Therneau, T. 2014. *survival: Survival Analysis*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/survival/survival.pdf>.
- Therneau, T. and C. Crowson 2014. *Using Time Dependent Covariates and Time Dependent Coefficients in the Cox Model*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/survival/vignettes/timedep.pdf>.
- Therneau, T. M. and P. M. Grambsch 2000. *Modeling Survival Data: Extending the Cox Model*. New York: Springer.
- Thompson, M. 1975. On any given Sunday: fair competitor orderings with maximum likelihood methods. *Journal of the American Statistical Association* 70(351):536–541. 42
- Thorn, J. 2011. *Baseball in the Garden of Eden: The Secret History of the Early Game*. New York: Nation Books.

- Thorn, J. and P. Palmer 1984. *The Hidden Game of Baseball: A Revolutionary Approach to Baseball and Its Statistics*. New York: Doubleday & Company.
- Thorn, J. and P. Palmer 1985. *The Hidden Game of Baseball: A Revolutionary Approach to Baseball and Its Statistics* (revised and updated ed.). New York: Doubleday. 65
- Thurman, W. N. and M. E. Fisher 1988. Chickens, eggs, and causality, or which came first? *American Journal of Agricultural Economics* 70(2):237–238.
- Thurstone, L. L. 1927. A law of comparative judgment. *Psychological Review* 34:273–286. 81
- Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B* 58:267–288. 213
- Torgerson, W. S. 1958. *Theory and Methods of Scaling*. New York: Wiley.
- Travers, J. and S. Milgram 1969, December. Experimental study of small world problem. *Sociometry* 32(4):425–443. 231
- Tsay, R. S. 2013. *An Introduction to Analysis of Financial Data with R*. New York: Wiley.
- Tufte, E. R. 1990. *Envisioning Information*. Cheshire, Conn.: Graphics Press.
- Tufte, E. R. 1997. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Cheshire, Conn.: Graphics Press.
- Tufte, E. R. 2004. *The Visual Display of Quantitative Information* (second ed.). Cheshire, Conn.: Graphics Press.
- Tufte, E. R. 2006. *Beautiful Evidence*. Cheshire, Conn.: Graphics Press.

- Tukey, J. W. 1977. *Exploratory Data Analysis*. Reading, Mass.: Addison-Wesley.
- Tukey, J. W. and F. Mosteller 1977. *Data Analysis and Regression: A Second Course in Statistics*. Reading, Mass.: Addison-Wesley.
- Turney, P. D. 2002, July 8–10. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02)*:417–424. Available from the National Research Council Canada publications archive. 224
- Tygiel, J. 2000. *Past Time: Baseball as History*. New York: Oxford University Press.
- Unwin, A., M. Theus, and H. Hofmann (eds.) 2006. *Graphics of Large Datasets: Visualizing a Million*. New York: Springer. 238
- USA Today 2015. Top team payrolls. *USA Today*. Retrieved from the World Wide Web on August 3, 2015 at <http://www.usatoday.com/sports/mlb/salaries/2014/team/all/>.
- Varian, H. R. 2005. *Intermediate Microeconomics: A Modern Approach* (seventh ed.). New York: Norton. 131
- Velleman, P. F. and L. Wilkinson 1993, February. Nominal, ordinal, interval, and ratio typologies are misleading. *The American Statistician* 47(1):65–72.
- Venables, W. N. and B. D. Ripley 2002. *Modern Applied Statistics with S* (fourth ed.). New York: Springer-Verlag. Champions of S, S-Plus, and R call this *the mustard book*.
- Walker, R. C. 2012. *Introduction to Mathematical*

- Programming* (fourth ed.). Upper Saddle River, N.J.: Pearson.
- Ward, B. 2015. *How Linux Works: What Every Superuser Should Know* (second ed.). San Francisco: No Starch Press.
- Ward, G. C. and K. Burns 1994. *Baseball: An Illustrated History*. New York: Alfred A. Knopf.
- Wasserman, L. 2010. *All of Statistics: A Concise Course in Statistical Inference*. New York: Springer. 205
- Wassertheil-Smoller, S. 1990. *Biostatistics and Epidemiology: A Primer for Health Professionals*. New York: Springer.
- Watson, J.-P. and D. L. Woodruff 2012. PySP: Modeling and solving stochastic programs in Python. *Mathematical Programming Computation* 4(2):109–149.
- Watts, D. J. 1999. *Small Worlds: The Dynamics of Networks between Order and Randomness*. Princeton, N.J.: Princeton University Press. 231
- Watts, D. J. and S. H. Strogatz 1998. Collective dynamics of ‘small-world’ networks. *Nature* 393(6684):440–442. 231
- Watts, R. G. and A. T. Bahill 2000. *Keep Your Eye on the Ball: Curve Balls, Knuckleballs, and Fallacies of Baseball* (revised and updated ed.). New York: W. H. Freeman and Company.
- Wedel, M. and W. Kamakura 2000. *Market Segmentation: Conceptual and Methodological Foundations* (second ed.). Boston: Kluwer.
- Weil, R. L., P. C. O’Brien, M. W. Maher, C. P. Stickney, and K. Fitzgerald 2005. *Accounting: The Language of Business* (eleventh ed.). Sun Lakes, Ariz.: Thomas Horton and Daughters.

Weinstock, J. 2012, January 11. Which umpire has the largest strike zone? Retrieved from the World Wide Web on September 12, 2015 at
<http://www.hardballtimes.com/which-umpire-has-the-largest-strikezone/>.

Weiss, S. M., N. Indurkhya, and T. Zhang 2010. *Fundamentals of Predictive Text Mining*. New York: Springer.

Wells, A. 2008. *Football & Chess: Tactics, Strategy, Beauty*. Devon, England: Hardinge Simpole.

West, B. T. 2006. A simple and flexible rating method for predicting success in the NCAA basketball tournament. *Journal of Quantitative Analysis in Sports* 2(3):1–14. 42

West, M. D. (ed.) 2001. *Theory, Method, and Practice in Computer Content Analysis*. Westport, Conn.: Ablex. 223

White, T. 2011. *Hadoop: The Definitive Guide* (second ed.). Sebastopol, Calif.: O'Reilly.

Wickham, H. 2014. *plyr: Tools for Splitting, Applying and Combining Data*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/plyr/plyr.pdf>.

Wickham, H. 2015. *Advanced R*. Boca Raton, Fla.: Chapman & Hall/CRC.

Wickham, H. and W. Chang 2014. *ggplot2: An Implementation of the Grammar of Graphics*. Comprehensive R Archive Network. 2014. <http://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf>. 238

Wikipedia 2015. Glossary of baseball. Retrieved from the World Wide Web on August 29, 2015 at

[https://en.m.wikipedia.org/wiki/Glossary_of_baseball.](https://en.m.wikipedia.org/wiki/Glossary_of_baseball)

- Wilkinson, L. 2005. *The Grammar of Graphics* (second ed.). New York: Springer.
- Will, G. 1990. *Men at Work: The Craft of Baseball*. New York: Macmillan.
- Will, G. F. 2014. *A Nice Little Place on the North Side: A History of Triumph, Mostly Defeat, and Incurable Hope at Wrigley Field*. New York: Three Rivers Press/Random House. 120
- Williams, H. P. 2013. *Model Building in Mathematical Programming* (fifth ed.). New York: Wiley.
- Wilson, J. 2008. *Inverting the Pyramid: The History of Soccer Tactics*. Devon, England: Hardinge Simpole.
- Winston, W. L. 2009. *Mathletics: How Gamblers, Managers, and Sports Enthusiasts Use Mathematics in Baseball, Basketball, and Football*. Princeton, N.J.: Princeton University Press.
- WinterGreen Research 2015. Sports Analytics: Market Shares, Strategies, and Forecasts, Worldwide, 2015 to 2021. Retrieved from the World Wide Web on October 3, 2015 at <http://www.reportsnreports.com/reports/401107-sports-analytics-market-shares-strategies-and-forecasts-worldwide-2015-to-2021.html>. 184
- Woody, C. 2014, December 19/21. Dodgers end season with record payroll, highest luxury tax in MLB. *Sports Illustrated*. Retrieved from the World Wide Web on August 4, 2015 at

<http://www.si.com/mlb/2014/12/19/los-angeles-dodgers-record-payroll-highest-luxury-tax/.9>

Worsley, J. C. and J. D. Drake 2002. *Practical PostgreSQL*. Sebastopol, Calif.: O'Reilly.

Wright, M. B. 2009. 50 years of OR in sport. *Journal of the Operational Research Society* 60: S161–S168.

Wunderman, L. 1996. *Being Direct: Making Advertising Pay*. New York: Random House.

Yau, N. 2011. *Visualize This: The FlowingData Guide to Design, Visualization, and Statistics*. New York: Wiley.

Yau, N. 2013. *Data Points: Visualization That Means Something*. New York: Wiley.

Young, G. and A. Householder 1938. Discussion of set of points in terms of their mutual distances. *Psychometrika* 3:19–22. 81

Zeileis, A., K. Hornik, and P. Murrell 2009, July. Escaping RGBland: Selecting colors for statistical graphics. *Computational Statistics and Data Analysis* 53(9):3259–3270.

Zeileis, A., K. Hornik, and P. Murrell 2014. *HCL-Based Color Palettes in R*. Comprehensive R Archive Network. 2014.
<http://cran.r-project.org/web/packages/colorspace/vignettes/hcl-colors.pdf>.

Zimbalist, A. 1992. *Baseball and Billions: A Probing Look Inside the Big Business of Our National Pastime*. New York: BasicBooks. 13

Zimbalist, A. 2003. *May the Best Team Win: Baseball*

Economics and Public Policy. Washington, D.C.: Brookings Institution Press.

Zinsser, W. K. 1989. *Spring Training*. New York: Harper & Row.

Zivot, E. and J. Wang 2003. *Modeling Financial Time Series with S-PLUS*. Seattle: Insightful Corporation.

Index

A

adjacency matrix, 261
advertising research, 229
agent, 261
agent-based modeling, 261
Akaike information criterion (AIC), 54
algebraic modeling system, 145, 261
Apache Hadoop, *see* Hadoop
Apache Lucene, *see* Lucene
Apache Software Foundation, 261
Apache Spark, *see* Spark
API, *see* application programming interface
application programming interface, 175, 178, 179, 261
ARPANET, 175, 262
ASP, 262
association rules, 262
asynchronous focus group, *see* blog

B

bandwidth, 262
baseball term
 “out”, 289
 “play ball”, 291
 “safe”, 293
 “time”, 296

ahead in the count, 279
All Star Game, 279
American League, 279
around the horn, 279
at bats (AB), 279
away team, 279
bailing out, 279
balk, 279
ball, 280
ban on women, 280
base, 280
base coach, 280
base hit, 280
base on balls (BB), 280
base runner, 280
baserunner, 280
baserunning error, 280
bases loaded, 280
batter, 280
batter in the hole, 280
batter on deck, 280
batter's box, 280
battery, 280
batting average (BA, AVG), 280
batting stance, 281
batting team, 281
behind in the count, 281
bench, 281

big leagues, 281
BIP (BPIP), 281
bloop single, 281
bunt, 281
call, 281
called game, 281
catcher, 281
caught looking, 281
caught off base, 281
caught stealing (CS), 281
center fielder, 282
Championship Series, 282
changeup, 282
check swing, 282
choking up, 282
chop single, 282
clean-up hitter, 282
closed batting stance, 282
closer, 282
clutch hitter, 282
coach, 282
command, 282
control, 282
cover the bases, 282
crowd the plate, 283
curveball (curve), 283
cut fastball (cutter), 283
cut-off position, 283

dead ball, 283
dead ball era, 283
dead red hitter, 283
defense, 283
defensive indifference, 283
designated hitter (DH), 283
diamond, 283
dig in, 284
Division Series, 284
double (2B), 284
double play, 284
double-header, 284
double-switch, 284
dugout, 284
earned run average (ERA), 284
expected runs, 62–65, 284
extra-base hit, 284
fair ball, 284
fair territory, 284
fan, 284
fantasy baseball, 284
fastball, 284
fielder, 285
fielder’s choice, 285
fielding error, 285
first base, 285
first baseman, 285
five-tool player, 285

fly ball, 285
fly out, 285
force out, 285
forfeited game, 285
foul ball, 285
foul territory, 285
foul tip, 285
frame (a pitch), 285
free agent, 285
full count, 285
game (G), 285
grand slam, 285
ground ball, 285
ground out, 286
ground-rule double, 286
hit (H), 286
hit batsman (hit by pitch, HBP), 286
hit for the cycle, 286
hit-and-run, 286
hitter, 286
hitter's park, 286
hitting for power, 286
hitting slump, 286
hitting streak, 286
holding runner on base, 286
home plate, 286
home run (HR), 286
home team, 287

illegal pitch, 287
in the hole, 287
infielder, 287
inning, 287
intentional base on balls, 287
interference, 287
keystone sack, 287
knuckleball, 287
lead-off hitter, 287
leave the yard (go yard), 287
left fielder (LF), 287
left on base (LOB), 287
lefty, 287
line drive, 288
lineup, 288
live ball, 288
live ball era, 288
making the turn, 288
manager, 288
manufactured run, 193, 288
men on base, 288
Mendoza Line, 288
middle infielder, 288
middle reliever, 288
National League, 288
neighborhood play, 288
no hitter, 289
no-no, 289

obstruction, 289
offense, 289
official scorer, 289
on the field (team), 289
on-base percentage (OBP), 289
open batting stance, 289
out, 289
outfielder, 289
overslide, 290
pace of play, 290
passed ball, 290
PECOTA, 33, 34, 290
perfect game, 290
pick off assignment, 290
pick off play, 290
pinch hitter, 290
pinch runner, 290
pitch, 290
pitch count, 290
pitcher (P), 291
pitcher's duel, 291
pitcher's park, 291
pitcher's plate, 291
pitching depth, 291
pitching from the stretch, 291
pitching mound, 291
pitching rotation, 291
pivot foot, 291

place hitter, 291
plate, 291
plate appearance, 291
platooning, 291
pop-up, 292
position number, 292
position player, 292
power hitter, 292
productive at bat, 292
pull hitter, 292
pull the string, 292
quick pitch, 292
reaching for the fences, 292
regulation game, 292
relief pitcher, 292
replay review, 292
retouch, 292
reverse curve, 292
right fielder (RF), 292
RISP, 293
rounding the bases, 293
run, 293
run batted in (RBI), 293
run down, 293
runner, 293
sacrifice bunt, 293
sacrifice fly, 293
scoring position, 293

screwball, 293
season, 293
second base, 293
second baseman, 293
secondary lead, 293
semi-intentional walk, 293
set position, 293
shadow ball, 294
shift, 294
shine ball, 294
shortstop, 294
shutout, 294
side-arm delivery, 294
single (1B), 294
slider, 294
slugging percentage (SLG), 294
small ball, 294
spin rate, 294
spitball, 295
squeeze play, 295
starting pitcher, 295
steal (stolen base, SB), 295
stepping in the bucket, 295
strike, 295
strike zone, 295
strikeout (K), 295
suspended game, 296
sweep, 296

switch hitter, 296
switch pitcher, 296
tag out, 296
tagging up, 296
take a lead (off base), 296
take a pitcher deep, 296
Texas Leaguer, 296
third base, 296
third baseman, 296
three-bagger, 296
throw, 296
throwing error, 296
tie game, 296
tip a pitch, 296
total bases (TB), 297
triple (3B), 297
triple crown, 297
triple-play, 297
two-bagger, 297
umpire, 297
umpire-in-chief, 297
up the middle, 297
up to bat (team), 297
visiting team, 297
VORP, 297
walk, 297
walk-off balk, 297
walk-off hit, 297

walk-off home run, 298
WAR (WARP), 298
WHIP, 298
Wild Card Game, 298
wild pitch, 298
windup position, 298
World Series, 298
Bayesian statistics, 54, 144, 198, 199, 204, 205, 262
 Bayes information criterion (BIC), 54
 Bayes' theorem, 204
 hierarchical modeling, 79, 267
best-case/worst-case approach, 144, 262
betweenness centrality, 232, 262
big data, 262
binary variable, 262
biologically-inspired methods, 216
black box model, 215
bootstrap method, 55, 57
bootstrap sampling, 262
bot, *see* crawler (web crawler)
boundary (of a network), 263
bps, 263
brand positioning, *see* marketing, brand positioning
breakeven analysis, *see* cost-volume-profit analysis
brute force approach, 263
bulletin board, *see* blog

C

C, C++, C#, 172, 173, 263
CART, 263
cascading style sheet (CSS), 263
case study
 Return of the Bobbleheads, 112
censoring, 112, 206, 207
chat room, 263
choice uncertainty, *see* decision analysis
classical statistics, 199, 203, 205, 263
 null hypothesis, 203
 power, 204
 statistical significance, 203, 204
classification, 198, 211–213, 215, 263
 predictive accuracy, 211–213
client, 263
client-server application, 263
closeness centrality, 232, 263
cloud computing, 183
cluster analysis, 71, 73, 122, 216, 263
coefficient of determination, 211
comma-delimited text (csv), 263
compile cycle, 264
complexity, of model, 213
conjoint analysis, 78, 79, 264
consumer heterogeneity, *see* market segmentation, consumer heterogeneity
consumer surplus, *see* pricing research, consumer surplus
content analysis, 264

continuous random variable, *see* random variable, continuous
cookie, 264
corpus, 264
cost-volume-profit analysis, 135–138
crawler, 175–178, 183, 264
cross-sectional data, 264
cross-validation, 55, 56, 214, 264
CSS, *see* cascading style sheet (CSS)
csv, *see* comma-delimited text (csv)
cumulative frequency distribution, *see* data visualization, cumulative frequency distribution
customer lifetime value, 125
CVP analysis, *see* cost-volume-profit analysis
Cython, 172

D

data mining, *see* machine learning
data organization, 227
data partitioning, 55
data preparation, 181
 missing data, 182
data science, 197, 198
data visualization, 264
 biplot, 81
 box plot, 102, 104
 cumulative frequency distribution, 239, 248
 diagnostics, 109, 213
 differential runs plot, 236

heat map, 161, 162
histogram, 10, 154, 157, 159, 160
lattice plot, 10, 105–108, 238
moving fraction plot, 237, 245, 246
Sankey diagram, 80
scatter plot, 11
spine chart, 72, 77, 78, 82, 88
strip plot, 105, 107
data-adaptive research, 52, 53
database system, 180, 181
MongoDB, 183, 270
MySQL, 270
non-relational, 183
NoSQL, 271
PostgreSQL, 183, 272
relational, 181, 183
decision analysis, 142, 143
decision tree, *see* decision analysis
decision uncertainty, *see* decision analysis
declarative language, 264
degree (of a network), 264
degree centrality, 265
degree distribution, 265
density (of a network), 265
descriptive statistics, 265
discounted cash flow analysis, 139–141
discount rate, 139
net present value (NPV), 139

payback period, 140
return on investment (ROI), 140
discrete random variable, *see* random variable, discrete
Document Object Model (DOM), 176, 265, 267
document store, 173, 177, 179, 180
DOM, *see* Document Object Model (DOM)

E

e-mail, 265
eigenvector centrality, 232, 265
Elasticsearch, 173, 182, 183, 225, 265
Elo rating/ranking system, 40
emoji, *see* emoticon
emoticon, 265
ethnography, 265
expected value, 265
experimental research, 265
explanatory model, 198, 266
explanatory variable, 51, 52, 124, 207, 266
exploratory data analysis, 102

F

factor analysis, 81, 266
fantasy sports, 35, 129, 130
forecasting, 229
Fortran, 172
frame, 266
ftp, 266
functional language, 266

G

game theory, 266
game-day strategy, *see* strategy, playing
General Inquirer, 223
generalized linear model, 214, 266
generative grammar, 266
genetic algorithm, 266
genetic algorithms, 216
Go (Golang), 172, 182, 266
graph theory, 267
grounded theory, 267

H

Hadoop, 267
heuristic, 216, 267
hierarchical modeling, *see* Bayesian statistics, hierarchical modeling
HTML, 175, 267
HTTP, 267

I

imputation, *see* multiple imputation
in-game strategy, *see* strategy, playing
indexing, 267
inferential statistics, 267
information retrieval, 182, 183, 220, 225, 267
injuries, 185, 189
integer knapsack problem, *see* mathematical programming, knapsack problem

integer programming, 267, *see* mathematical programming,
integer programming
integrated development environment (IDE), 267
interaction effect, 213
Internet, 175, 267
Internet of Things, 170, 268
Internet Services Provider, 268
intranet, 268
investment analysis, *see* discounted cash flow analysis
IoT, *see* Internet of Things
IRC, 268
IT, 268

J

Java, 172, 173, 268
Java Virtual Machine, *see* Java
JavaScript, 268
JavaScript Object Notation (JSON), 179, 268
JPEG, 268
JSON, *see* JavaScript Object Notation (JSON)
JVM, *see* Java

K

kbps, 268
keyword, 268

L

LAMP, 268
leading indicator, 229

leave-one-out cross-validation, *see* [cross-validation](#)
levels of measurement, 269
linear model, 207, 214
linear predictor, 207
linear programming, 269, *see* [mathematical programming](#),
[linear programming](#)
linear regression, 269
Linux, 173, 183
listserv, 269
logging, *see* [system logging](#)
logistic regression, 42, 52, 211, 269
longitudinal data, 269
Lucene, 173, 225, 269

M

machine learning, 215, 216, 269
Major League Baseball, *see* [MLB](#)
Major League Soccer, *see* [MLS](#)
market basket analysis, 73
market response model, 111, 226–229
market segmentation, 79, 120–122
 consumer heterogeneity, 127, 131
marketing
 competitive analysis, 128, 129
 mass marketing, 123
 one-to-one marketing, 123
product positioning, 71, 72, 74
strategy, 128, 129

substitute products, 74
target marketing, 120
marketing mix model, 111, 228
Markov chain, 62, 269
 transition probability, 62, 269
mass marketing, *see* marketing, mass marketing
mathematical programming, 127, 200, 269
 integer programming, 200, 201, 267
 knapsack problem, 41, 200, 201, 268
 linear programming, 200, 269
 mixed integer programming, 200, 269
 stochastic programming, 134, 145, 275
measurement, 23–35, 269
 accessible measure, 31
 comprehensible measure, 31, 32
 construct validity, 24, 27
 content validity, 24, 224
 convergent validity, 27
 discriminant validity, 27
 explicit measure, 31
 face validity, 24, 224
 levels of measurement, 28–29
 multitrait-multimethod matrix, 24–26
 predictive validity, 24
 reliability, 24–26, 30, 274
 tractable measure, 31
 transparent measure, 31, 33
 validity, 24, 26, 27, 31, 32, 277

MEG, *see* [listserv](#)

mixed integer programming, [269](#), *see* [mathematical programming](#), [mixed integer programming](#)

MKP (multidimensional knapsack problem), *see* [mathematical programming](#), [knapsack problem](#)

MLB, [4](#), [5](#), [9–11](#), [64](#), [65](#), [170](#), [201](#), [232](#), [233](#), [256](#), [288](#)

MLS, [257](#)

model, [270](#)

model-dependent research, [52](#), [53](#)

modem, [270](#)

Moneyball, [9](#)

MongoDB, *see* [database system](#), [MongoDB](#)

morphology, [270](#)

multi-fold cross-validation, *see* [cross-validation](#)

multi-level categorical variable, [270](#)

multi-level modeling, *see* [Bayesian statistics](#), [hierarchical modeling](#)

multidimensional knapsack problem, *see* [mathematical programming](#), [knapsack problem](#)

multidimensional scaling, [71](#), [72](#), [81](#)

 Jaccard index, [73](#)

 perceptual map, [12](#), [13](#), [70–72](#), [241](#)

 similarity judgment, [71–73](#)

multinomial variable, *see* [multi-level categorical variable](#)

multiple imputation, [270](#)

multivariate methods, [81](#)

MySQL, *see* [database system](#), [MySQL](#)

N

näive Bayes model, 270

National Basketball Association, *see* NBA

National Football League, *see* NFL

National Hockey League, *see* NHL

natural language processing, *see* text analytics, natural language processing

NBA, 4, 6, 9, 10, 12, 38–40, 65–67, 170, 238, 239, 258

nearest-neighbor analysis, 73

net present value, *see* discounted cash flow analysis, net present value (NPV)

netiquette, 270

network, 270

network science, 270

neural network, 270

NFL, 4, 7, 9, 10, 170, 259

NHL, 260

NLP, *see* text analytics, natural language processing

NoSQL, *see* database system, NoSQL

NPV, *see* discounted cash flow analysis, net present value (NPV)

O

object-oriented language, 271

observational research, 271

on-base percentage plus slugging (OPS), 289

one-to-one marketing, *see* marketing, one-to-one marketing

online community, 271, 277

operations research, 271

optimization, 216

organic search, 271

Orme, B. K., 131

outlier, 271

over-fitting, 213

P

PageRank, 271

paid search, 271

paired comparisons, 42

panel, 271

panel data, 271, *see also* longitudinal data

parameter, 272

parametric models, 213

parser, 176, 177, 272

payback period, *see also* discounted cash flow analysis

PECOTA, *see also* baseball term, PECOTA

perceptual map, *see also* multidimensional scaling, perceptual map

Perl, 272

PHP, 272

Poisson distribution, 272

Poisson regression, 206, 207, 210, 214, 272

population, 272

population distribution, 272

Porter five-forces model, 128, 129

post, 272

posterior distribution, 272

PostgreSQL, *see also* database system, PostgreSQL

predictive analytics

definition, [51](#)
predictive model, [198](#), [272](#)
preference scaling, [81](#)
price elasticity, *see* [pricing research](#), [price elasticity](#)
price sensitivity, *see* [pricing research](#), [price sensitivity](#)
pricing research, [126](#)
 consumer surplus, [128](#)
 price elasticity, [126](#)
 price sensitivity, [127](#)
 reference price, [127](#)
 three Cs of pricing, [126](#)
 willingness to pay, [75](#), [78](#), [79](#), [126](#), [127](#)
principal component analysis, [81](#), [216](#), [272](#)
prior distribution, [273](#)
probability, [64](#)
 binomial distribution, [158](#)
 negative binomial distribution, [156](#), [158](#), [161](#)
 Poisson distribution, [156](#), [158](#), [161](#)
procedural language, [273](#)
product positioning, [81](#), *see* [marketing](#), [product positioning](#)
psychographics, [273](#)
Python, [171–173](#), [182](#), [273](#)
Python package
 json, [189](#)
 matplotlib, [86](#), [116](#), [242](#)
 numpy, [86](#), [99](#), [116](#), [168](#), [242](#)
 os, [186](#)
 pandas, [99](#), [116](#), [242](#)

patsy, [99](#)
scipy, [116](#), [168](#)
scrapy, [186](#)
sklearn, [86](#)
statsmodels, [99](#), [116](#), [242](#)
twitter, [189](#)

Q

queuing theory, [273](#)

R

R, [171–173](#), [182](#), [273](#)

R package

car, [113](#)
ggplot2, [18](#), [245](#), [246](#), [248](#)
grid, [18](#), [245](#), [246](#), [248](#)
lattice, [16](#), [113](#), [167](#)
lubridate, [248](#)
MASS, [83](#)
plyr, [248](#)
support.CEs, [88](#)

R-squared, [211](#)

random forests, [273](#)

random network (random graph), [231](#), [273](#)

random variable, [273](#)

continuous, [273](#)

discrete, [273](#)

real-time analytics, [171](#), [172](#), [182](#)

real-time focus group, *see* [chat room](#)

recency-frequency-monetary value (RFM) model, 126
reference price, *see pricing research*, reference price regression, 52, 108–111, 197, 206, 213, 273
 nonlinear regression, 214
 robust methods, 214
 time series regression, 228
regular expressions, 178, 273
regularized regression, 213
relational database, 274
reliability, *see measurement*, reliability
resampling, 274
response, 51
response variable, 124, 206
return on investment, *see discounted cash flow analysis*
robot, *see crawler (web crawler)*
ROI, *see discounted cash flow analysis*
root-mean-square error (RMSE), 211

S

sales forecasting, 226, 228
sampling, 274
 sampling variability, 204
sampling distribution, 274
sampling frame, 274
Scala, 172, 274
scatter plot, 274
scheduling, 216
scraper, 176, 177, 274

search, *see* information retrieval
semantic web, 274
semantics, 274
semi-supervised learning, 216, 274
sensitivity analysis, 145, 274
sentiment analysis, 222–225
shrinkage estimators, 213
similarity judgment, *see* multidimensional scaling, similarity judgment
simulated annealing, 274
simulation, 148, 153, 214, 275
 benchmark study, 214, 215
 game-day, 59, 151, 152
 what-if analysis, 198
small-world network, 231
smoothing methods, 213
 splines, 214
social network analysis, 230, 275
Solr, 225, 275
Spark, 172, 182, 275
spatial data, 275
spider, *see* crawler (web crawler)
SQL, 181, 275
statistic
 interval estimate, 203
 p-value, 203
 point estimate, 203
 test statistic, 203

statistical learning, *see* [machine learning](#)
stemming (word stemming), [275](#)
stochastic process, [275](#)
stochastic programming, *see* [mathematical programming](#),
[stochastic programming](#)
strategy
 playing, [61–68](#)
strength of schedule, [38](#)
structured query language, *see* [SQL](#)
substitute product, *see* [marketing](#), [substitute product](#)
supervised learning, [206](#), [216](#), [222](#), [275](#)
support vector machine, [275](#)
survival analysis, [206](#), [207](#)
synchronous focus group, *see* [chat room](#)
syntax, [275](#)
system logging, [275](#)

T

target marketing, *see* [marketing](#), [target marketing](#)
TCP/IP, [275](#)
telnet, [276](#)
term frequency-inverse document frequency, *see* [TF-IDF](#)
text analytics, [217–225](#)
 content analysis, [223](#)
 document annotation, [35](#)
 generative grammar, [217](#), [218](#)
 latent Dirichlet allocation, [216](#)
 latent semantic analysis, [216](#)

morphology, 218

natural language processing, 217, 224, 270

semantics, 218

stemming, 219

syntax, 218

terms-by-documents matrix, 219, 221

text analysis, 276

text feature, 35

text measure, 35, 223–225

text summarization, 222

thematic analysis, 216, 223

text measure, 276

text mining, 276

text parser, *see parser*

TF-IDF, 220, 276

thread, of discussion, 276

time series, 276

time series analysis, 226–229

- ARIMA model, 227
- state space model, 228

time-value of money, 142

traditional research, 52

traditional statistics, 276

training-and-test regimen, 54, 55, 105, 108, 198, 199

transition probability, *see Markov chain, transition probability*

transitivity, 276

tree-structured model, 276

U

unidimensional scaling, 38, 40, 42, 43
unsupervised learning, 216, 220, 276
URL, 175, 277
Usenet, 277

V

validity, *see* measurement, validity
variable transformation, 213
virtual facility, 277
virtual private network (VPN), 277
VPN, *see* virtual private network

W

web, *see* World Wide Web
web board, *see* blog
web presence testing, 277
web scraper, *see* scraper
weblog, *see* blog
Wiki, 277
willingness to pay, *see* pricing research, willingness to pay
WNBA, 255
Women's National Basketball Association, *see* WNBA
World Wide Web, 174, 175, 183, 277
WWW, *see* World Wide Web

X

XML, 277
XPath, 176, 277

Code Snippets

```
# MLB, NBA, and NFL Player Salaries (R)

library(lattice) # statistical graphics

# variables in contract data from spotrac.com (August 2015)
#   player: player name (contract years)
#   position: position on team
#   team: team abbreviation
#   teamsignedwith: team that signed the original contract
#   age: age in years as of August 2015
#   years: years as player in league
#   contract: dollars in contract
#   guaranteed: guaranteed dollars in contract
#   guaranteedpct: percentage of contract dollars guaranteed
#   salary: annual salary in dollars
#   yearfreeagent: year player becomes free agent
#
#   additional created variables
#   salarymm: salary in millions
#   leaguename: full league name
#   league: league abbreviation

# read data for Major League Baseball
mlb_contract_data <- read.csv("mlb_player_salaries_2015.csv")
mlb_contract_data$leaguename <- rep("Major League Baseball",
  length = nrow(mlb_contract_data))
for (i in seq(along = mlb_contract_data$yearfreeagent))
  if (mlb_contract_data$yearfreeagent[i] == 0)
    mlb_contract_data$yearfrccagcnt[i] <- NA
for (i in seq(along = mlb_contract_data$age))
  if (mlb_contract_data$age[i] == 0)
    mlb_contract_data$age[i] <- NA
```

```
mlb_contract_data$salarymm <- mlb_contract_data$salary/1000000
mlb_contract_data$league <- rep("MLB", length = nrow(mlb_contract_data))
print(summary(mlb_contract_data))
# variables for plotting
mlb_data_plot <- mlb_contract_data[, c("salarymm","leaguename")]

nba_contract_data <- read.csv("nba_player_salaries_2015.csv")
nba_contract_data$leaguename <- rep("National Basketball Association",
length = nrow(nba_contract_data))
for (i in seq(along = nba_contract_data$yearfreeagent))
  if (nba_contract_data$yearfreeagent[i] == 0)
    nba_contract_data$yearfreeagent[i] <- NA
for (i in seq(along = nba_contract_data$age))
  if (nba_contract_data$age[i] == 0)
    nba_contract_data$age[i] <- NA
nba_contract_data$salarymm <- nba_contract_data$salary/1000000
nba_contract_data$league <- rep("NBA", length = nrow(nba_contract_data))
print(summary(nba_contract_data))
# variables for plotting
nba_data_plot <- nba_contract_data[, c("salarymm","leaguename")]
```

```
nfl_contract_data <- read.csv("nfl_player_salaries_2015.csv")
nfl_contract_data$leaguename <- rep("National Football League",
length = nrow(nfl_contract_data))
for (i in seq(along = nfl_contract_data$yearfreeagent))
  if (nfl_contract_data$yearfreeagent[i] == 0)
    nfl_contract_data$yearfreeagent[i] <- NA
for (i in seq(along = nfl_contract_data$age))
  if (nfl_contract_data$age[i] == 0)
    nfl_contract_data$age[i] <- NA
nfl_contract_data$salarymm <- nfl_contract_data$Salary/1000000
nfl_contract_data$league <- rep("NFL", length = nrow(nfl_contract_data))
print(summary(nfl_contract_data))

# variables for plotting
nfl_data_plot <- nfl_contract_data[, c("salarymm", "leaguename")]

# merge contract data with variables for plotting
plotting_data_frame <- rbind(mlb_data_plot, nba_data_plot, nfl_data_plot)

# generate the histogram lattice for comparing player salaries
# across the three leagues in this study
lattice_object <- histogram(~ salarymm | leaguename, plotting_data_frame,
type = "density", xlab = 'Annual Salary ($ millions)', layout = c(1,3))

# print to file
pdf(file = "fig_understanding_markets_player_salaries.pdf",
width = 8.5, height = 11)
print(lattice_object)
dev.off()
```

```
# Payroll and Performance in Major League Baseball (R)

library(ggplot2) # statistical graphics

# functions used with grid graphics to split the plotting region
# to set margins and to plot more than one ggplot object on one page/screen
vplayout <- function(x, y)
  viewport(layout.pos.row=x, layout.pos.col=y)

# user-defined function to plot a ggplot object with margins
ggplot.print.with.margins <- function(ggplot.object.name,
  left.margin.pct=10,
  right.margin.pct=10,top.margin.pct=10,bottom.margin.pct=10)
  { # begin function for printing ggplot objects with margins
    # margins expressed as percentages of total... use integers
    grid.newpage()
    pushViewport(viewport(layout=grid.layout(100,100)))
    print(ggplot.object.name,
      vp=vplayout((0 + top.margin.pct):(100 - bottom.margin.pct),
      (0 + left.margin.pct):(100 - right.margin.pct)))
  } # end function for printing ggplot objects with margins

# read in payroll and performance data
# including annotation text for team abbreviations
mlb_data <- read.csv("mlb_payroll_performance_2014.csv")
```

```
mlb_data$millions <- mlb_data$payroll/1000000
mlb_data$winpercent <- mlb_data$wl pct * 100

cat("\nCorrelation between Payroll and Performance:\n")
with(mlb_data, print(cor(millions, winpercent)))

cat("\nProportion of win/loss percentage explained by payrolls:\n")
with(mlb_data, print(cor(millions, winpercent)^2))

pdf(file = "fig_understanding_markets_payroll_performance.pdf",
     width = 5.5, height = 5.5)
ggplot_object <- ggplot(data = mlb_data,
    aes(x = millions, y = winpercent)) +
    geom_point(colour = "darkblue", size = 3) +
    xlab("Team Payroll (Millions of Dollars)") +
    ylab("Percentage of Games Won") +
    geom_text(aes(label = textleft), size = 3, hjust = 1.3) +
    geom_text(aes(label = textright), size = 3, hjust = -0.25)

ggplot.print.with.margins(ggplot_object, left.margin.pct = 5,
    right.margin.pct = 5, top.margin.pct = 5, bottom.margin.pct = 5)

dev.off()
```

```

# Making a Perceptual Map of Sports (R)

library(MASS) # includes functions for multidimensional scaling
library(wordcloud) # textplot utility to avoid overlapping text

USE_METRIC_MDS <- FALSE # metric versus non-metric toggle

# utility function for converting a distance structure
# to a distance matrix as required for some routines and
# for printing of the complete matrix for visual inspection.
make.distance.matrix <- function(distance_structure)
{
  n <- attr(distance_structure, "Size")
  full <- matrix(0,n,n)
  full[lower.tri(full)] <- distance_structure
  full+t(full)
}

# enter data into a distance structure as required for various
# distance based routines. That is, we enter the upper triangle
# of the distance matrix as a single vector of distances
distance_structure <-
  as.single(c(9,11,10,5,14,4,15,6,12,13,15,1,18,2,20,7,3,19,17,8,21))

# provide a character vector of sports names
sport_names <- c("Baseball", "Basketball", "Football",
  "Soccer", "Tennis", "Hockey", "Golf")

attr(distance_structure, "Size") <- length(sport_names) # set size attribute

```

```
# check to see that the distance structure has been entered correctly
# by converting the distance structure to a distance matrix
# using the utility function make.distance.matrix, which we had defined
distance_matrix <- unlist(make.distance.matrix(distance_structure))
cat("\n","Distance Matrix of Seven Sports","\n")
print(distance_matrix)

if (USE_METRIC_MDS)
{
  # apply the metric multidimensional scaling algorithm and plot the map
  mds_solution <- cmdscale(distance_structure, k=2, eig=T)
}

# apply the non-metric multidimensional scaling algorithm
# this is more appropriate for rank-order data
# and provides a more satisfactory solution here

if (!USE_METRIC_MDS)
{
  mds_solution <- isoMDS(distance_matrix, k = 2, trace = FALSE)
}

pdf(file = "plot_nonmetric_mds_seven_sports.pdf",
     width=8.5, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))

# original solution
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2]
```

```

# set up the plot but do not plot points... use names for points
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # first page of pdf plots
# We plot the sport names in the locations where points normally go.
text(First_Dimension, Second_Dimension, labels = sport_names,
     offset = 0.0, cex = 1.5)
title("Seven Sports (initial solution)")

# reflect the horizontal dimension
# multiply the first dimension by -1 to get reflected image
First_Dimension <- mds_solution$points[,1] * -1
Second_Dimension <- mds_solution$points[,2]
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # second page of pdf plots
text(First_Dimension, Second_Dimension, labels = sport_names,
     offset = 0.0, cex = 1.5)
title("Seven Sports (horizontal reflection)")

# reflect the vertical dimension
# multiply the second dimension by -1 to get reflected image
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2] * -1
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # third page of pdf plots
text(First_Dimension, Second_Dimension, labels = sport_names,
     offset = 0.0, cex = 1.5)
title("Seven Sports (vertical reflection)")

# multiply the first and second dimensions by -1
# for reflection in both horizontal and vertical directions
First_Dimension <- mds_solution$points[,1] * -1
Second_Dimension <- mds_solution$points[,2] * -1
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,

```

```

xlim = c(-15, 15), ylim = c(-15, 15)) # fourth page of pdf plots
text(First_Dimension, Second_Dimension, labels = sport_names,
     offset = 0.0, cex = 1.5)
title("Seven Sports (horizontal and vertical reflection)")
dev.off() # closes the pdf plotting device

pdf(file = "plot_pretty_original_mds_seven_sports.pdf",
     width=8.5, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] # no reflection
Second_Dimension <- mds_solution$points[,2] # no reflection
# wordcloud utility for plotting with no overlapping text
textplot(x = First_Dimension,
          y = Second_Dimension,
          words = sport_names,
          show.lines = FALSE,
          xlim = c(-15, 15), # extent of horizontal axis range
          ylim = c(-15, 15), # extent of vertical axis range
          xaxt = "n", # suppress tick marks
          yaxt = "n", # suppress tick marks
          cex = 1.15, # size of text points
          mgp = c(0.85, 1, 0.85), # position of axis labels
          cex.lab = 1.5, # magnification of axis label text
          xlab = "",
          ylab = "")
dev.off() # closes the pdf plotting device

```

```
pdf(file = "fig_sports_perceptual_map.pdf",
     width=8.5, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] + -1 # reflect horizontal
Second_Dimension <- mds_solution$points[,2]
# wordcloud utility for plotting with no overlapping text
textplot(x = First_Dimension,
          y = Second_Dimension,
          words = sport_names,
          show.lines = FALSE,
          xlim = c(-15, 15), # extent of horizontal axis range
          ylim = c(-15, 15), # extent of vertical axis range
          xaxt = "n", # suppress tick marks
          yaxt = "n", # suppress tick marks
          cex = 1.15, # size of text points
          wgp = c(0.85, 1, 0.85), # position of axis labels
          cex.lab = 1.5, # magnification of axis label text
          xlab = "First Dimension (Individual/Team, Degree of Contact)",
          ylab = "Second Dimension (Anaerobic/Aerobic, Other")
dev.off() # closes the pdf plotting device
```

```

# Assessing Team Strength by Unidimensional Scaling (3)

# NBA Regular Season 2014-2015 data from Basketball-Reference.com

# read comma-delimited text file to create data frame
nba_input <- read.csv("basketball_2014_2015_season.csv",
  stringsAsFactors = FALSE)

# add record number to records
record_number <- seq(nrow(nba_input))
nba_scores <- cbind(data.frame(record_number), nba_input)

# explore the data
with(nba_scores, plot(home_points, visitor_points))
with(nba_scores, table(overtime))

# define minutes played using overtime information
nba_scores$minutes <- rep(48, length = nrow(nba_scores)) # standard game
for (i in seq(along = nba_scores$minutes)) {
  if(nba_scores$overtime[i] == "OT")
    nba_scores$minutes[i] <- nba_scores$minutes[i] + 5 # 1 OT period
  if(nba_scores$overtime[i] == "2OT")
    nba_scores$minutes[i] <- nba_scores$minutes[i] + 10 # 2 OT periods
  if(nba_scores$overtime[i] == "3OT")
    nba_scores$minutes[i] <- nba_scores$minutes[i] + 15 # 3 OT periods
}
with(nba_scores, table(overtime, minutes)) # check minute calculations

# compute points per minute for each team
nba_scores$visitor_ppm <- nba_scores$visitor_points / nba_scores$minutes
nba_scores$home_ppm <- nba_scores$home_points / nba_scores$minutes

```

```

# check calculations
print(head(nba_scores))
print(tail(nba_scores))

# explore points-per-minute data
with(nba_scores, plot(home_ppm, visitor_ppm))

# read in team names and abbreviations
team_info <- read.csv("nba_team_names_abbreviations.csv",
  stringsAsFactors = FALSE)

# append team information to nba_scores data frame
list_of_team_names <- team_info$team_name
# first do the visitor team
nba_scores$visitor_conference <- rep("", length = nrow(nba_scores))
nba_scores$visitor_division <- rep("", length = nrow(nba_scores))
nba_scores$visitor_code <- rep("", length = nrow(nba_scores))
for (i in seq(along = list_of_team_names)) {
  this_team_info <-
    team_info[(team_info$team_name == list_of_team_names[i]),]
  indices_for_visitor_team <-
    which(nba_scores$visitor_team == list_of_team_names[i])
  for (j in seq(along = indices_for_visitor_team)) {
    nba_scores$visitor_conference[indices_for_visitor_team[j]] <-
      this_team_info$conference[1]
    nba_scores$visitor_division[indices_for_visitor_team[j]] <-
      this_team_info$division[1]
    nba_scores$visitor_code[indices_for_visitor_team[j]] <-
      this_team_info$abbreviation[1]
  }
}

```

```

# next do the home team
nba_scores$home_conference <- rep("", length = nrow(nba_scores))
nba_scores$home_division <- rep("", length = nrow(nba_scores))
nba_scores$home_code <- rep("", length = nrow(nba_scores))
for (i in seq(along = list_of_team_names)) {
  this_team_info <-
    team_info[(team_info$team_name == list_of_team_names[i]),]
  indices_for_home_team <-
    which (nba_scores$home_team == list_of_team_names[i])
  for (j in seq(along = indices_for_home_team)) {
    nba_scores$home_conference[indices_for_home_team[j]] <-
      this_team_info$conference[1]
    nba_scores$home_division[indices_for_home_team[j]] <-
      this_team_info$division[1]
    nba_scores$home_code[indices_for_home_team[j]] <-
      this_team_info$abbreviation[1]
  }
}

# define winning team (Visitor or Home)
nba_scores$win_visitor_home <- rep("Home", length = nrow(nba_scores))
for (i in seq(along = nba_scores$record_number))
  if (nba_scores$visitor_ppm[i] > nba_scores$home_ppm[i])
    nba_scores$win_visitor_home[i] <- "Visitor"

# who wins more games.. visitor or home team
with(nba_scores, table(win_visitor_home))
with(nba_scores, plot(home_ppm, visitor_ppm))

# define winning and losing teams by three-character abbreviation/code
nba_scores$win_team_code <- rep("", length = nrow(nba_scores))
nba_scores$loss_team_code <- rep("", length = nrow(nba_scores))
for (i in seq(along = nba_scores$record_number)) {
  if (nba_scores$visitor_ppm[i] > nba_scores$home_ppm[i]) {

```

```

nba_scores$win_team_code[i] <- nba_scores$visitor_code[i]
nba_scores$lose_team_code[i] <- nba_scores$home_code[i]
}
if (nba_scores$visitor_ppm[i] < nba_scores$home_ppm[i]) {
  nba_scores$win_team_code[i] <- nba_scores$home_code[i]
  nba_scores$lose_team_code[i] <- nba_scores$visitor_code[i]
}
}

# check the nba_scores data frame
print(head(nba_scores))
print(tail(nba_scores))

# create OKC data frame for testing and visualization example
OKC_visitor <- nba_scores[(nba_scores$visitor_code == "OKC"),]
OKC_home <- nba_scores[(nba_scores$home_code == "OKC"),]
OKC_data <- rbind(OKC_visitor, OKC_home)
OKC_data <- OKC_data[sort.list(OKC_data$record_number),]
# save file for Oklahoma City Thunder for data visualization work
write.csv(OKC_data, file = "okc_data_2014_2015.csv", row.names = FALSE)

# write win team score and lose team score in points per minute
nba_scores$win_team_ppm <- rep(0, length = nrow(nba_scores))
nba_scores$lose_team_ppm <- rep(0, length = nrow(nba_scores))
for (i in seq(along = nba_scores$record_number)) {
  if (nba_scores$visitor_ppm[i] > nba_scores$home_ppm[i]) {
    nba_scores$win_team_ppm[i] <- nba_scores$visitor_ppm[i]
    nba_scores$lose_team_ppm[i] <- nba_scores$home_ppm[i]
  }
  if (nba_scores$visitor_ppm[i] < nba_scores$home_ppm[i]) {
    nba_scores$win_team_ppm[i] <- nba_scores$home_ppm[i]
    nba_scores$lose_team_ppm[i] <- nba_scores$visitor_ppm[i]
  }
}

```

```

}

# compute margin of victory in points per minute
nba_scores$margin_ppm <- nba_scores$win_team_ppm - nba_scores$lose_team_ppm

# initialize matrices for entire set of 30 teams
ordered_team_codes <- sort(team_info$abbreviation)
# total number of wins
wins_mat <- matrix(0, nrow = 30, ncol = 30,
                     dimnames = list(ordered_team_codes, ordered_team_codes))
prop_mat <- wins_mat # proportion wins
ppm_mat <- wins_mat # sum of ppm scores

# build matrices for entire set of 30 teams
for (i in seq(along = nba_scores$record_number)) {
  # tally the number of times team i beats team j
  wins_mat[nba_scores$lose_team_code[i], nba_scores$win_team_code[i]] <-
    wins_mat[nba_scores$lose_team_code[i], nba_scores$win_team_code[i]] + 1
  # sum the ppm scores for teams and enter in ppm mat
  # two entries needed for each record in nba_scores
  ppm_mat[nba_scores$lose_team_code[i], nba_scores$win_team_code[i]] <-
    ppm_mat[nba_scores$lose_team_code[i], nba_scores$win_team_code[i]] +
    nba_scores$win_team_ppm[i]
  ppm_mat[nba_scores$win_team_code[i], nba_scores$lose_team_code[i]] <-
    ppm_mat[nba_scores$win_team_code[i], nba_scores$lose_team_code[i]] +
    nba_scores$lose_team_ppm[i]
}

# check wins_mat entries... should be 82 games for every team
for (j in 1:length(ordered_team_codes))
  print(sum(wins_mat[,j]) + sum(wins_mat[j,]))

```

```

# compute matrix of proportions prop_mat
# proportion of times team j beats team k
for(j in 1:length(ordered_team_codes)) { # begin outer for-loop
    for(k in 1:length(ordered_team_codes)) { # begin inner for-loop
        if (j == k) prop_mat[j,k] <- NA # set diagonal entries missing
        if (j > k) { # begin outer if-block
            between_team_games <-
                wins_mat[ordered_team_codes[j],ordered_team_codes[k]] +
                wins_mat[ordered_team_codes[k],ordered_team_codes[j]]

            # if teams never play others within the season
            # we set the entry in the cell
            # to be the mean of the team
            if (between_team_games == 0) { # begin first inner if-block
                prop_mat[j,k] <- sum(wins_mat[j,]) /
                    (sum(wins_mat[,j]) + sum(wins_mat[j,]))
                prop_mat[k,j] <- sum(wins_mat[k,]) /
                    (sum(wins_mat[,k]) + sum(wins_mat[k,]))
            } # end first inner if-block

            # when teams play other teams at least once
            # we compute the proportion of times they beat one another
            if (between_team_games > 0) { # begin second inner if-block
                prop_mat[j,k] <-
                    wins_mat[ordered_team_codes[j],ordered_team_codes[k]]/
                        between_team_games
                prop_mat[k,j] <-
                    wins_mat[ordered_team_codes[k],ordered_team_codes[j]]/
                        between_team_games
            } # end second inner if-block
        } # end outer if-block
    } # end inner for-loop
} # end outer for-loop

```

```

# check prop_mat against known results for the season
# results should be close but not exact
# because prop_mat adjusts for strength of schedule
for(i in seq(along = ordered_team_codes))
  cat("\n", ordered_team_codes[i], ":", mean(prop_mat[,i], na.rm = TRUE))
# unidimensional scaling work begins here
# begin by defining the paired_comparisons scaling matrix
pc_mat <- prop_mat
for(j in 1:length(ordered_team_codes))
  for(k in 1:length(ordered_team_codes))
    if (j == k) pc_mat[j,k] <- 0.50 # set diagonal entries missing

nobjects <- length(ordered_team_codes)

mean_pc_mat <- numeric(nobjects)
object_quantile <- numeric(nobjects)
for(j in 1:nobjects)
{
  mean_pc_mat[j] <- mean(pc_mat[,j])
  object_quantile[j] <- qnorm(mean_pc_mat[j])
}

# user-defined function provides mean 500 standard deviation 100
z.score.converter <- function(z) {round(((100*z) + 500),digits=0)}

scale_score <- numeric(nobjects)
for(j in 1:nobjects)
  scale_score[j] <- z.score.converter(object_quantile[j])

sorted_team_info <- team_info[sort.list(team_info$abbreviation),]
sorted_team_info$scale_ecore <- ecalscore

```

```

sorted_team_info$name_with_score <- rep("", length = nrow(sorted_team_info))
for(i in seq(along = ordered_team_codes))
  sorted_team_info$name_with_score[i] <-
    paste(sorted_team_info$team_name[i],
          " (", sorted_team_info$scale_score[i], ")",
          sep = "")

scaling_frame <-
  sorted_team_info[,c("abbreviation", "team_name",
                     "name_with_score", "scale_score")]
names(scaling_frame) <-
  c("object.name", "long.object.name",
    "long.object.name.with.score", "scale_score")

ordered_scaling_frame <-
  scaling_frame[scrt.list(scale_score, decreasing=FALSE),]

# plotting to external pdf file using standard R graphics
pdf(file = "fig_unidimensional_scaling_analysis.pdf",
     width = 11, height = 8.5)
text.angle = 45
par(mfrow=c(1,1), xpd=NA, cex=1, yaxt= "n" , lwd=3, cty="n",
     srt=text.angle, mar=c(5, 0, 4, 0) + 0.1)
plot(ordered_scaling_frame$scale_score,
      rep(0.1,length(ordered_scaling_frame$scale_score)), type="h",
      xlab=paste("Team Scale Scores Adjusted for Strength of Schedule"),
      ylab="",
      ylim=c(min(ordered_scaling_frame$scale_score) - 50,
            max(ordered_scaling_frame$scale_score) + 50))

text(ordered_scaling_frame$scale_score + 40,
      rep(0.66,length(ordered_scaling_frame$scale_score)),
      ordered_scaling_frame$long.object.name.with.score, pos=2, cex=1)

dev.off()

```

```

# Product Positioning of Entertainment Events and Activities (R)

library(MASS) # includes functions for multidimensional scaling
library(wordcloud) # textplot utility to avoid overlapping text

USE_METRIC_MDS <- FALSE # metric versus non-metric toggle

# utility function for converting a distance structure
# to a distance matrix as required for some routines and
# for printing of the complete matrix for visual inspection.
make.distance.matrix <- function(distance_structure)
{
  n <- attr(distance_structure, "Size")
  full <- matrix(0,n,n)
  full[lower.tri(full)] <- distance_structure
  full+t(full)
}

# enter data into a distance structure as required for various
# distance-based routines. That is, we enter the upper triangle
# of the distance matrix as a single vector of distances
distance_structure <-
  as.single(c(6,11,5,8,15,14,10,3,2,19,18,9,4,17,16,7,13,12,21,20,1))

# provide a character vector of entertainment event or activity names
activity_names <-
  c("Comedy", "Symphony", "Zoo", "Pop Music", "Museum", "Hockey", "Football")

attr(distance_structure, "Size") <- length(activity_names) # set size attribute

# check to see that the distance structure has been entered correctly
# by converting the distance structure to a distance matrix
# using the utility function make.distance.matrix, which we had defined

```

```

distance_matrix <- unlist(make.distance.matrix(distance_structure))
cat("\n","Distance Matrix of Seven Activities","\n")
print(distance_matrix)

if (USE_METRIC_MDS)
{
  # apply the metric multidimensional scaling algorithm and plot the map
  mds_solution <- cmdscale(distance_structure, k=2, eig=T)
}

# apply the non-metric multidimensional scaling algorithm
# this is more appropriate for rank-order data
# and provides a more satisfactory solution here

if (!USE_METRIC_MDS)
{
  mds_solution <- isoMDS(distance_matrix, k = 2, trace = FALSE)
}

pdf(file = "plot_nonmetric_mds_seven_activities.pdf",
     width=8.5, height=8.5) # opens pdf plotting device

# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))

# original solution
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2]

# set up the plot but do not plot points... use names for points
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # first page of pdf plots
# We plot the sport names in the locations where points normally go.
text(First_Dimension, Second_Dimension, labels = activity_names,

```

```

    offset = 0.0, cex = 1.5)
title("Seven Activities (initial solution)")

# reflect the horizontal dimension
# multiply the first dimension by -1 to get reflected image
First_Dimension <- mds_solution$points[,1] * -1
Second_Dimension <- mds_solution$points[,2]
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # second page of pdf plots
text(First_Dimension, Second_Dimension, labels = activity_names,
     offset = 0.0, cex = 1.5)
title("Seven Activities (horizontal reflection)")

# reflect the vertical dimension
# multiply the second dimension by -1 to get reflected image
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2] * -1
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # third page of pdf plots
text(First_Dimension, Second_Dimension, labels = activity_names,
     offset = 0.0, cex = 1.5)
title("Seven Activities (vertical reflection)")

# multiply the first and second dimensions by -1
# for reflection in both horizontal and vertical directions
First_Dimension <- mds_solution$points[,1] * -1
Second_Dimension <- mds_solution$points[,2] * -1
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-15, 15), ylim = c(-15, 15)) # fourth page of pdf plots
text(First_Dimension, Second_Dimension, labels = activity_names,

```

```
    offset = 0.0, cex = 1.5)
title("Seven Activities (horizontal and vertical reflection)")
dev.off() # closes the pdf plotting device

pdf(file = "plot_pretty_original_mds_seven_activities.pdf",
     width=8.5, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] # no reflection
Second_Dimension <- mds_solution$points[,2] # no reflection
# wordcloud utility for plotting with no overlapping text
textplot(x = First_Dimension,
          y = Second_Dimension,
          words = activity_names,
          show.lines = FALSE,
          xlim = c(-15, 15), # extent of horizontal axis range
          ylim = c(-15, 15), # extent of vertical axis range
          xaxt = "n", # suppress tick marks
          yaxt = "n", # suppress tick marks
          cex = 1.15, # size of text points
          mgp = c(0.85, 1, 0.85), # position of axis labels
          cex.lab = 1.6, # magnification of axis label text
          xlab = "First Dimension",
          ylab = "Second Dimension")
dev.off() # closes the pdf plotting device
```

```

# Product Positioning of Entertainment Events and Activities (Python)

# prepare for Python version 3x features and functions
from __future__ import division, print_function

# import packages for multivariate analysis
import numpy as np # arrays and numerical processing
import scipy
import matplotlib.pyplot as plt # 2D plotting

# alternative distance metrics for multidimensional scaling
from sklearn.metrics import euclidean_distances
from sklearn.metrics.pairwise import linear_kernel as cosine_distances
from sklearn.metrics.pairwise import manhattan_distances as manhattan_distances

from sklearn import manifold # multidimensional scaling

# These are the original data from one respondent
# Pairs of activities are judged on their similarity
# Smaller numbers are more similar to one another
# Zero on the diagonal means no difference
#   0   6   11   5   8   15   14 Comedy
#   6   0   10   3   2   19   18 Symphony
#   11   10   0   9   4   17   16 Zoo
#   5   3   9   0   7   13   12 Pop Music
#   8   2   4   7   0   21   20 Muscum
#   15   19   17   13   21   0   1 Hockey
#   14   18   16   12   20   1   0 Football

# define a numpy array for these data
distances_matrix = np.array([
    [0, 6, 11, 5, 8, 15, 14], \
    [6, 0, 10, 3, 2, 19, 18], \

```

```

[11, 10, 9, 4, 17, 16], \
[5, 3, 9, 0, 7, 13, 12], \
[8, 2, 4, 7, 0, 21, 20], \
[15, 19, 17, 13, 21, 0, 1], \
[14, 18, 16, 12, 20, 1, 0]])

# check to see that the distance structure has been entered correctly
print(distance_matrix)
print(type(distance_matrix))

# apply the multidimensional scaling algorithm and plot the map
mds_method = manifold.MDS(n_components = 2, random_state = 9999, \
    dissimilarity = 'precomputed')
mds_fit = mds_method.fit(distance_matrix)
mds_coordinates = mds_method.fit_transform(distance_matrix)

activity_label = ["Comedy", "Symphony", "Zoo", \
    "Pop Music", "Museum", "Hockey", "Football"]

# plot mds solution in two dimensions using activity labels
# defined by multidimensional scaling
plt.figure()
plt.scatter(mds_coordinates[:,0], mds_coordinates[:,1], \
    facecolors = 'none', edgecolors = 'none') # points in white (invisible)
labels = activity_label
for label, x, y in zip(labels, mds_coordinates[:,0], mds_coordinates[:,1]):
    plt.annotate(label, (x,y), xycoords = 'data')
plt.xlabel('First Dimension')
plt.ylabel('Second Dimension')
plt.show()
plt.savefig('fig_positioning_products_mds_activities_python.pdf',
    bbox_inches = 'tight', dpi=None, facecolor='w', edgecolor='b',
    orientation='landscape', papertype=None, format=None,
    transparent=True, pad_inches=0.25, frameon=None)

```

```

# Consumer Preferences for Sporting Events---Conjoint Analysis (R)

library(support.CEs) # package for survey construction

# generate a balanced set of product profiles for survey
# varies with each call to the function
# so each individual gets a different set of sixteen items
provider.survey <- Lma.design(attribute.names =
  list(price = c("PRICE $20", "PRICE $45", "PRICE $70", "PRICE $95"),
  seating = c("Field", "Loge", "Reserved", "Top Deck"),
  boxvip = c("Box/VIP NO", "Box/VIP YES"),
  frontrow = c("Front Row NO", "Front Row YES"),
  promotion = c("Ball", "Bobblehead", "Cap", "None")),
  nalternatives = 1, nblocks=1, seed = 7777)
print(questionnaire(provider.survey)) # print survey design for review

sink("questions_for_survey.txt") # send survey to external text file
questionnaire(provider.survey)
sink() # send output back to the screen
print.digits <- 2 # set number of digits on print and spine chart

# user-defined function for printing conjoint measures
if (print.digits == 2)
  pretty.print <- function(x) {sprintf("%1.2f", round(x,digits = 2))}
if (print.digits == 3)
  pretty.print <- function(x) {sprintf("%1.3f", round(x,digits = 3))}

# -----
# user-defined function for printing conjoint measures
# spine chart accommodates up to 45 part-worths on one page
# |part-worth| <= 40 can be plotted directly on the spine chart
# |part-worths| > 40 can be accommodated through
#   standardization# user-defined function for spine chart

```

```

# -----
spine.chart <- function(conjoint.results,
  color.for.part.worth.point = "blue",
  color.for.part.worth.line = "blue",
  left.side.symbol.to.print.around.part.worths = "(",
  right.side.symbol.to.print.around.part.worths = ")",
  left.side.symbol.to.print.around.importance = "",
  right.side.symbol.to.print.around.importance = "",
  color.for.printing.importance.text = "dark red",
  color.for.printing.part.worth.text = "black",
  draw.gray.background = TRUE,
  draw.optional.grid.lines = TRUE,
  print.internal.consistency = TRUE,
  fix.max.to.4 = FALSE,
  put.title.on.spine.chart = FALSE,
  title.on.spine.chart = paste("TITLE GOES HERE IF WE ASK FOR ONE",sep ""),
  plot.framing.box = TRUE,
  do.ordered.attributes = TRUE) {

  # fix.max.to.4 option to override the range for part-worth plotting

  if(!do.ordered.attributes) effect.names <- conjoint.results$attributes
  if(do.ordered.attributes) effect.names <-
    conjoint.results$ordered.attributes

  number.of.levels.of.attribute <- NULL
  for(index.for.factor in seq(along=effect.names))
    number.of.levels.of.attribute <- c(number.of.levels.of.attribute,
      length(conjoint.results$xlevels[[effect.names[index.for.factor]]]))

  # total number of levels needed for vertical length of spine the spine plot
  total.number.of.levels <- sum(number.of.levels.of.attribute)
}

```

```
# define size of spaces based upon the number of part-worth levels to plot
if(total.number.of.levels <= 20) {
  smaller.space <- 0.01
  small.space <- 0.02
  medium.space <- 0.03
  large.space <- 0.04
}

if(total.number.of.levels > 20) {
  smaller.space <- 0.01 + 0.9
  small.space <- 0.02 + 0.9
  medium.space <- 0.03 + 0.9
  large.space <- 0.04 + 0.9
}

if(total.number.of.levels > 22) {
  smaller.space <- 0.01 + 0.85
  small.space <- 0.02 + 0.85
  medium.space <- 0.03 + 0.825
  large.space <- 0.04 + 0.8
}

if(total.number.of.levels > 25) {
  smaller.space <- 0.01 + 0.8
  small.space <- 0.02 + 0.8
  medium.space <- 0.03 + 0.75
  large.space <- 0.04 + 0.75
}

if(total.number.of.levels > 35) {
  smaller.space <- 0.01 + 0.65
  small.space <- 0.02 + 0.65
  medium.space <- 0.03 + 0.6
  large.space <- 0.04 + 0.6
```

```
}

# of course there is a limit to how much we can plot on one page
if (total.number.of.levels > 45)
  stop("\n\nTERMINATED: More than 45 part-worths on spine chart\n")
part.worth.plotting.list <- conjoint.results$part.worths

# check the range of part-worths to see which path to go down for plotting
# initialize these toggles to start
max.is.less.than.40 <- FALSE
max.is.less.than.20 <- FALSE
max.is.less.than.10 <- FALSE
max.is.less.than.4 <- FALSE
max.is.less.than.2 <- FALSE
max.is.less.than.1 <- FALSE

if (max(abs(min(unlist(part.worth.plotting.list),na.rm=TRUE)),
       max(unlist(part.worth.plotting.list),na.rm=TRUE)) <= 40) {
  max.is.less.than.40 <- TRUE
  max.is.less.than.20 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

if (max(abs(min(unlist(part.worth.plotting.list),na.rm=TRUE)),
       max(unlist(part.worth.plotting.list),na.rm=TRUE)) <= 20) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- TRUE
  max.is.less.than.10 <- FALSE
  max.is.less.than.4 <- FALSE
```

```
max.is.less.than.2 <- FALSE
max.is.less.than.1 <- FALSE
}

if(max(abs(min(unlist(part.worth.plotting.list),na.rm=TRUE)),
      max(unlist(part.worth.plotting.list),na.rm=TRUE)) <= 10) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.10 <- TRUE
  max.is.less.than.4 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

if (max(abs(min(unlist(part.worth.plotting.list),na.rm=TRUE)),
       max(unlist(part.worth.plotting.list),na.rm=TRUE)) <= 4) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.4 <- TRUE
  max.is.less.than.10 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}

if(max(abs(min(unlist(part.worth.plotting.list),na.rm=TRUE)),
      max(unlist(part.worth.plotting.list),na.rm=TRUE)) <= 2) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.2 <- TRUE
  max.is.less.than.1 <- FALSE
```

```
}

if(max(abs(min(unlist(part.worth.plotting.list),na.rm=TRUE)),
      max(unlist(part.worth.plotting.list),na.rm=TRUE)) <= 1) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.4 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- TRUE
}

# sometimes we override the range for part-worth plotting
# this is not usually done... but it is an option
if (fix.max.to.4) {
  max.is.less.than.40 <- FALSE
  max.is.less.than.20 <- FALSE
  max.is.less.than.10 <- FALSE
  max.is.less.than.4 <- TRUE
  max.is.less.than.2 <- FALSE
  max.is.less.than.1 <- FALSE
}
if (!max.is.less.than.1 & !max.is.less.than.2 & !max.is.less.than.4 &
    !max.is.less.than.10 & !max.is.less.than.20 & !max.is.less.than.40)
  stop("\n\nTERMINATED: Spine chart cannot plot |part-worth| > 40")

# determine point positions for plotting part-worths on spine chart
if (max.is.less.than.1 | max.is.less.than.2 | max.is.less.than.4 |
    max.is.less.than.10 | max.is.less.than.20 | max.is.less.than.40) {
  # begin if-block plotting when all part-worths in absolute value
  # are less than one of the tested range values
  # part-worth positions for plotting
```

```
# end if-block plotting when all part-worths in absolute value
# are less than one of the tested range values
# offsets for plotting vary with the max.is.less.than setting
if(max.is.less.than.1) {
  list.scaling <- function(x) {0.75 + x/5}
  part.worth.point.position <-
    lapply(part.worth.plotting.list,list.scaling)
}

if(max.is.less.than.2) {
  list.scaling <- function(x) {0.75 + x/10}
  part.worth.point.position <-
    lapply(part.worth.plotting.list,list.scaling)
}

if(max.is.less.than.4) {
  list.scaling <- function(x) {0.75 + x/20}
  part.worth.point.position <-
    lapply(part.worth.plotting.list,list.scaling)
}

if(max.is.less.than.10) {
  list.scaling <- function(x) {0.75 + x/60}
  part.worth.point.position <-
    lapply(part.worth.plotting.list,list.scaling)
}

if(max.is.less.than.20) {
  list.scaling <- function(x) {0.75 + x/100}
  part.worth.point.position <-
    lapply(part.worth.plotting.list,list.scaling)
}

if(max.is.less.than.40) {
  list.scaling <- function(x) {0.75 + x/200}
```

```
part.worth.point.position <-
  lapply(part.worth.plotting.list,list.scaling)
}

part.worth.point.position <- lapply(part.worth.plotting.list,list.scaling)
}

if (plot.framing.box) plot(c(0,0,1,1),c(0,1,0,1),xlab="",ylab="",
  type="n",xaxt="n",yaxt="n")

if (!plot.framing.box) plot(c(0,0,1,1),c(0,1,0,1),xlab="",ylab="",
  type="n",xaxt="n",yaxt="n", bty="n")

if (put.title.on.spine.chart) {
  text(c(0.50),c(0.975),pos=3,labels=title.on.spine.chart,cex=01.5)
  y.location <- 0.925 # starting position with title
}

if (!put.title.on.spine.chart) y.location <- 0.975 # no-title start

# store top of vertical line for later plotting needs
y.top.of.vertical.line <- y.location

x.center.position <- 0.75 # horizontal position of spine

# begin primary plotting loop
# think of a plot as a collection of text and symbols on screen or paper
# we are going to construct a plot one text string and symbol at a time
# (note that we may have to repeat this process at the end of the program)
for(k in seq(along=effect.names)) {
```

```
y.location <- y.location - large.space
text(c(0.4),c(y.location),pos=2,
  labels=paste(effect.name.map(effect.names[k])," ",sep=""),cex=01.0)
text(c(0.525),c(y.location),pos=2,col=color.for.printing.importance.text,
  labels=paste(" ",left.side.symbol.to.print.around.importance,
  pretty.print(
    unlist(conjoint.results$attribute.importance[effect.names[k]])), "%",
    right.side.symbol.to.print.around.importance,sep=""),cex=01.0)

# begin loop for printing part-worths
for(m in seq(1:number.of.levels.of.attribute[k])) {
  y.location <- y.location - medium.space
  text(c(0.4),c(y.location),pos=2,
    conjoint.results$x.level[[effect.names[k]]][m],cex=01.0)
#  part.worth.label.data.frame[k,m],cex=01.0)

  text(c(0.525),c(y.location),pos=2,
    col=color.for.printing.part.worth.text,
    labels=paste(" ",left.side.symbol.to.print.around.part.worths,
    pretty.print(part.worth.plotting.list[[effect.names[k]]][m]),
    right.side.symbol.to.print.around.part.worths,sep=""),cex=01.0)
```

```
points(part.worth.point.position[[effect.names[k]]][m],y.location,
      type = "p", pch = 20, col = color.for.part.worth.point, cex = 2)
segments(x.center.position, y.location,
         part.worth.point.position[[effect.names[k]]][m], y.location,
         col = color.for.part.worth.line, lty = 1, lwd = 2)
    }
}

y.location <- y.location - medium.space

# begin center axis and bottom plotting
y.bottom.of.vertical.line <- y.location # store top of vertical line

below.y.bottom.of.vertical.line <- y.bottom.of.vertical.line - small.space/2

if (!draw.gray.background) {
  # four optional grid lines may be drawn on the plot parallel to the spine
  if (draw.optional.grid.lines) {
    segments(0.55, y.top.of.vertical.line, 0.55,
            y.bottom.of.vertical.line, col = "black", lty = "solid", lwd = 1)

    segments(0.65, y.top.of.vertical.line, 0.65,
            y.bottom.of.vertical.line, col = "gray", lty = "solid", lwd = 1)

    segments(0.85, y.top.of.vertical.line, 0.85,
            y.bottom.of.vertical.line, col = "gray", lty = "solid", lwd = 1)

    segments(0.95, y.top.of.vertical.line, 0.95,
            y.bottom.of.vertical.line, col = "black", lty = "solid", lwd = 1)
  }
}
```

```

# gray background for plotting area of the points
if (draw.gray.background) {
  rect(xleft = 0.55, ybottom = y.bottom.of.vertical.line,
    xright = 0.85, ytop = y.top.of.vertical.line, density = -1, angle = 45,
    col = 'light gray', border = NULL, lty = "solid", lwd = 1)

# four optional grid lines may be drawn on the plot parallel to the spine
if (draw.optional.grid.lines) {
  segments(0.55, y.top.of.vertical.line, 0.55,
    y.bottom.of.vertical.line, col = 'black', lty = "solid", lwd = 1)

  segments(0.55, y.top.of.vertical.line, 0.85,
    y.bottom.of.vertical.line, col = 'white', lty = "solid", lwd = 1)

  segments(0.85, y.top.of.vertical.line, 0.85,
    y.bottom.of.vertical.line, col = 'white', lty = "solid", lwd = 1)

  segments(0.85, y.top.of.vertical.line, 0.95,
    y.bottom.of.vertical.line, col = 'black', lty = "solid", lwd = 1)
}

# draw the all-important spine on the plot
segments(x.center.position, y.top.of.vertical.line, x.center.position,
  y.bottom.of.vertical.line, col = "black", lty = "dashed", lwd = 1)

# horizontal line at top
segments(0.55, y.top.of.vertical.line, 0.95, y.top.of.vertical.line,
  col = "black", lty = 1, lwd = 1)

# horizontal line at bottom
segments(0.55, y.bottom.of.vertical.line, 0.95, y.bottom.of.vertical.line,
  col = "black", lty = 1, lwd = 1)

```

```

# plot for ticks and labels
segments(0.55, y.bottom.of.vertical.line,
  0.55, below.y.bottom.of.vertical.line,
  col = "black", lty = 1, lwd = 1) # tick line at bottom

segments(0.65, y.bottom.of.vertical.line,
  0.65, below.y.bottom.of.vertical.line,
  col = "black", lty = 1, lwd = 1) # tick line at bottom

segments(0.75, y.bottom.of.vertical.line,
  0.75, below.y.bottom.of.vertical.line,
  col = "black", lty = 1, lwd = 1) # tick line at bottom

segments(0.85, y.bottom.of.vertical.line,
  0.85, below.y.bottom.of.vertical.line,
  col = "black", lty = 1, lwd = 1) # tick line at bottom

segments(0.95, y.bottom.of.vertical.line,
  0.95, below.y.bottom.of.vertical.line,
  col = "black", lty = 1, lwd = 1) # tick line at bottom

# axis labels vary with the max.is.less.than range being used
if (max.is.less.than.1) text(c(0.55,0.65,0.75,0.85,0.95),
  rep(below.y.bottom.of.vertical.line,times=5),
  pos=1,labels=c("-1","-0.5","0","+0.5","+1"),cex=0.75)

if (max.is.less.than.2) text(c(0.55,0.65,0.75,0.85,0.95),
  rep(below.y.bottom.of.vertical.line,times=5),
  pos=1,labels=c("-2","-1","0","+1","+2"),cex=0.75)

if (max.is.less.than.4) text(c(0.55,0.65,0.75,0.85,0.95),
  rep(below.y.bottom.of.vertical.line,times=5),
  pos=1,labels=c("-4","-2","0","+2","+4"),cex=0.75)

```

```
if (max.is.less.than.10) text(c(0.55,0.65,0.75,0.85,0.95),
  rep(below.y.bottom.of.vertical.line,times=5),
  pos=1,labels=c("-10","-5","0","+5","+10"),cex=0.75)

if (max.is.less.than.20) text(c(0.55,0.65,0.75,0.85,0.95),
  rep(below.y.bottom.of.vertical.line,times=5),
  pos=1,labels=c("-20","-10","0","+10","+20"),cex=0.75)

if (max.is.less.than.40) text(c(0.55,0.65,0.75,0.85,0.95),
  rep(below.y.bottom.of.vertical.line,times=5),
  pos=1,labels=c("-40","-20","0","+20","+40"),cex=0.75)

y.location <- below.y.bottom.of.vertical.line - small.space

text(.75,y.location,pos=1,labels=c('Part-Worth'),
  cex=0.95)

y.location <- below.y.bottom.of.vertical.line - small.space

text(0.75,y.location,pos=1,labels=c("Part-Worth"), cex=0.95)

if(print.internal.consistency) {
  y.location <- y.location - medium.space
  text(c(0.525),c(y.location),pos=2,labels=paste("Internal consistency: ",
  pretty.print(conjoint.results$internal.consistency),
  sep=""))
}
```

```
# if we have grid lines we may have plotted over part-worth points
# if we have a gray background then we have plotted over part-worth points
# so let us plot those all-important part-worth points and lines once again
if(draw.gray.background || draw.optional.grid.lines) {
  y.location <- y.top.of.vertical.line # retrieve the starting value

  # repeat the primary plotting loop
  for(k in seq(along=effect.names)) {
    y.location <- y.location - large.space
    text(c(0.4),c(y.location),pos=2,
         labels=paste(effect.name.map(effect.names[k])," ",sep=""),cex=01.0)
```

```

text(c(0.525),c(y.location),pos=2,col=color.for.printing.importance.text,
  labels=paste(" ",left.side.symbol.to.print.around.importance,
  pretty.print(
  unlist(conjoint.results$attribute.importance[effect.names[k]])), "%",
  right.side.symbol.to.print.around.importance,sep=""),cex=01.0)

# begin loop for printing part-worths
for(m in seq(1:number.of.levels.of.attribute[k])) {
  y.location <- y.location - medium.space
  text(c(0.4),c(y.location),pos=2,
  conjoint.results$xlevel[[effect.names[k]]][m],cex=01.0)

  text(c(0.525),c(y.location),
    pos=2,col=color.for.printing.part.worth.text,
    labels=paste(" ",left.side.symbol.to.print.around.part.worths,
    pretty.print(part.worth.plotting.list[[effect.names[k]]][m]),
    right.side.symbol.to.print.around.part.worths,sep=""),cex=01.0)

  points(part.worth.point.position[[effect.names[k]]][m],y.location,
    type = "p", pch = 20, col = color.for.part.worth.point, cex = 2)
  segments(x.center.position, y.location,
  part.worth.point.position[[effect.names[k]]][m], y.location,
  col = color.for.part.worth.line, lty = 1, lwd = 2)
}
}
}
}

```

```

# user-defined function for plotting descriptive attribute names
effect.name.map <- function(effect.name) {
  if(effect.name=="price") return("Ticket Price")
  if(effect.name=="seating") return("Seating Area")
  if(effect.name=="boxvip") return("Box/VIP")
  if(effect.name=="frontrow") return("Front Row")
  if(effect.name=="promotion") return("Promotion")
}
# read in conjoint survey profiles with respondent ranks
conjoint.data.frame <- read.csv("sporting_event_ranking.csv")

# set up sum contrasts for effects coding as needed for conjoint analysis
options(contrasts=c("contr.sum","contr.poly"))
# main effects model specification
main.effects.model <-
  (ranking ~ price + seating + boxvip + frontrow + promotion)

# fit linear regression model using main effects only (no interaction terms)
main.effects.model.fit <- lm(main.effects.model, data=conjoint.data.frame)
print(summary(main.effects.model.fit))

# save key list elements of the fitted model as needed for conjoint measures
conjoint.results <-
  main.effects.model.fit[c("contrasts","xlevels","coefficients")]
conjoint.results$attributes <- names(conjoint.results$contrasts)

# compute and store part-worths in the conjoint.results list structure
part.worths <- conjoint.results$xlevels # list of same structure as xlevels
end.index.for.coefficient <- 1 # initialize skipping the intercept
part.worth.vector <- NULL # used for accumulation of part worths
for(index.for.attribute in seq(along=conjoint.results$contrasts)) {
  nlevels <- length(unlist(conjoint.results$xlevels[index.for.attribute]))

```

```

begin.index.for.coefficient <- end.index.for.coefficient + 1
end.index.for.coefficient <- begin.index.for.coefficient + nlevels -2
last.part.worth <- -sum(conjoint.results$coefficients[
  begin.index.for.coefficient:end.index.for.coefficient])
part.worths[index.for.attribute] <-
  list(as.numeric(c(conjoint.results$coefficients[
    begin.index.for.coefficient:end.index.for.coefficient],
    last.part.worth)))
part.worth.vector <-
  c(part.worth.vector,unlist(part.worths[index.for.attribute]))
}
conjoint.results$part.worths <- part.worths

# compute and store part-worth ranges for each attribute
part.worth.ranges <- conjoint.results$contrasts
for(index.for.attribute in seq(along=conjoint.results$contrasts))
  part.worth.ranges[index.for.attribute] <-
    dist(range(conjoint.results$part.worths[index.for.attribute]))
conjoint.results$part.worth.ranges <- part.worth.ranges

sum.part.worth.ranges <- sum(as.numeric(conjoint.results$part.worth.ranges))

# compute and store importance values for each attribute
attribute.importance <- conjoint.results$contrasts
for(index.for.attribute in seq(along=conjoint.results$contrasts))
  attribute.importance[index.for.attribute] <-
    (dist(range(conjoint.results$part.worths[index.for.attribute]))/
     sum.part.worth.ranges) * 100
conjoint.results$attribute.importance <- attribute.importance

```

```

# data frame for ordering attribute names
attribute.name <- names(conjoint.results$contrasts)
attribute.importance <- as.numeric(attribute.importance)
temp.frame <- data.frame(attribute.name,attribute.importance)
conjoint.results$ordered.attributes <-
  as.character(temp.frame[sort.list(
    temp.frame$attribute.importance,decreasing = TRUE),"attribute.name"])

# respondent internal consistency added to list structure
conjoint.results$internal.consistency <- summary(nain.effects.model.fit)$r.squared

# user-defined function for printing conjoint measures
if (print.digits == 2)
  pretty.print <- function(x) {sprintf("%1.2f",round(x,digits = 2))}
if (print.digits == 3)
  pretty.print <- function(x) {sprintf("%1.3f",round(x,digits = 3))}

# report conjoint measures to console
# use pretty.print to provide nicely formated output
for(k in seq(along=conjoint.results$ordered.attributes)) {
  cat("\n","\n")
  cat(conjoint.results$ordered.attributes[k],"Levels: ",
  unlist(conjoint.results$xlevels[conjoint.results$ordered.attributes[k]]))

  cat("\n"," Part-Worths: ")
  cat(pretty.print(unlist(conjoint.results$part.worths
  [conjoint.results$ordered.attributes[k]])))

```

```
cat("\n", " Attribute Importance: ")
cat(pretty.print(unlist(conjoint.results$attribute.importance
  [conjoint.results$ordered.attributes[k]])))
}

# plotting of spine chart begins here
# all graphical output is routed to external pdf file
pdf(file = "fig_zsports_willingness_to_pay.pdf", width=8.5, height=11)
spine.chart(conjoint.results)
dev.off() # close the graphics output device
```

```
# Consumer Preferences for Sporting Events---Conjoint Analysis (Python)

# prepare for Python version 3x features and functions
from __future__ import division, print_function

# import packages for analysis and modeling
import pandas as pd # data frame operations
import numpy as np # arrays and math functions
import statsmodels.api as sm # statistical models (including regression)
import statsmodels.formula.api as smf # R-like model specification
from patsy.contrasts import Sum

# read in conjoint survey profiles with respondent ranks
conjoint_data_frame = pd.read_csv('sporting_event_ranking.csv')

# set up sum contrasts for effects coding as needed for conjoint analysis
# using C(effect, Sum) notation within main effects model specification
main_effects_model = 'ranking ~ C(price, Sum) + C(seating, Sum) + \
    C(boxvip, Sum) + C(frontrow, Sum) + C(promotion, Sum)'

# fit linear regression model using main effects only (no interaction terms)
main_effects_model_fit = \
    smf.ols(main_effects_model, data = conjoint_data_frame).fit()
print(main_effects_model_fit.summary())
conjoint_attributes = ['price', 'seating', 'boxvip', 'frontrow', 'promotion']

# build part-worth information one attribute at a time
level_name = []
part_worth = []
part_worth_range = []
```

```

end = 1 # initialize index for coefficient in params
for item in conjoint_attributes:
    level_set = set(conjoint_data_frame[item])
    nlevels = len(level_set)
    level_name.append(list(sorted(list(level_set))))
    begin = end
    end = begin + nlevels - 1
    new_part_worth = list(main_effects_model_fit.params[begin:end])
    new_part_worth.append((-1) * sum(new_part_worth))
    part_worth_range.append(max(new_part_worth) - min(new_part_worth))
    part_worth.append(new_part_worth)
    # end set to begin next iteration

# compute attribute relative importance values from ranges
attribute_importance = []
for item in part_worth_range:
    attribute_importance.append(round(100 * (item / sum(part_worth_range)),2))
# user-defined dictionary for printing descriptive attribute names
effect_name_dict = {'price' : 'Ticket Price', \
    'seating' : 'Seating Area', 'boxvip' : 'Box/VIP', \
    'frontrow' : 'Front Row', 'promotion' : 'Promotion'}

# report conjoint measures to console
index = 0 # initialize for use in for-loop
for item in conjoint_attributes:
    print('\nAttribute:', effect_name_dict[item])
    print('    Importance:', attribute_importance[index])
    print('    Level Part-Worths')
    for level in range(len(level_name[index])):
        print('        ',level_name[index][level], part_worth[index][level])
    index = index + 1

```

```
# Predictive Model for Los Angeles Dodgers Promotion and Attendance (R)

library(car) # special functions for linear regression
library(lattice) # graphics package

# read in data and create a data frame called dodgers
dodgers <- read.csv("dodgers.csv")
print(str(dodgers)) # check the structure of the data frame

# define an ordered day-of-week variable
# for plots and data summaries
dodgers$ordered_day_of_week <- with(data=dodgers,
  ifelse ((day_of_week == "Monday"),1,
  ifelse ((day_of_week == "Tuesday"),2,
  ifelse ((day_of_week == "Wednesday"),3,
  ifelse ((day_of_week == "Thursday"),4,
  ifelse ((day_of_week == "Friday"),5,
  ifelse ((day_of_week == "Saturday"),6,7))))))

dodgers$ordered_day_of_week <- factor(dodgers$ordered_day_of_week, levels=1:7,
  labels=c("Mon", "Tue", "Wed", "Thur", "Fri", "Sat", "Sun"))

# exploratory data analysis with standard graphics: attendance by day of week
with(data=dodgers,plot(ordered.day.of.week, attend/1000,
xlab = "Day of Week", ylab = 'Attendance (thousands)',
col = "violet", las = 1))

# when do the Dodgers use bobblehead promotions
with(dodgers, table(bobblehead,ordered.day.of.week)) # bobbleheads on Tuesday

# define an ordered month variable
# for plots and data summaries
dodgers$ordered_month <- with(data=dodgers,
  ifelse ((month == "APR"),4,
```

```

ifelse ((month == "MAY"),5,
ifelse ((month == "JUN"),6,
ifelse ((month == "JUL"),7,
ifelse ((month == "AUG"),8,
ifelse ((month == "SEP"),9,10))))))
dodgers$ordered_month <- factor(dodgers$ordered_month, levels=4:10,
labels = c("April", "May", "June", "July", "Aug", "Sept", "Oct"))

# exploratory data analysis with standard R graphics: attendance by month
with(data=dodgers,plot(ordered_month,attend/1000, xlab = "Month",
ylab = "Attendance (thousands)", col = "light blue", las = 1))

# exploratory data analysis displaying many variables
# looking at attendance and conditioning on day/night
# the skies and whether or not fireworks are displayed
library(lattice) # used for plotting
# let us prepare a graphical summary of the dodgers data
group.labels <- c("No Fireworks","Fireworks")
group.symbols <- c(21,24)
group.colors <- c("black","black")
group.fill <- c("black","red")
xyplot(attend/1000 ~ temp | skies + day_night,
       data = dodgers, groups = fireworks, pch = group.symbols,
       aspect = 1, cex = 1.5, col = group.colors, fill = group.fill,
       layout = c(2, 2), type = c("p", "g"),
       strip=strip.custom(strip.levels=TRUE,strip.names=FALSE, style=1),
       xlab = "Temperature (Degrees Fahrenheit)",
       ylab = "Attendance (thousands)",
       key = list(space = "top",
                  text = list(rev(group.labels),col = rev(group.colors)),
                  points = list(pch = rev(group.symbols), col = rev(group.colors),
                               fill = rev(group.fill)))))

# attendance by opponent and day/night game

```

```

group.labels <- c("Day","Night")
group.symbols <- c(1,20)
group.symbols.size <- c(2,2.75)
bwplot(opponent ~ attend/1000, data = dodgers, groups = day_night,
       xlab = "Attendance (thousands)",
       panel = function(x, y, groups, subscripts, ...)
         {panel.grid(h = (length(levels(dodgers$opponent)) - 1), v = -1)
          panel.stripplot(x, y, groups = groups, subscripts = subscripts,
                          cex = group.symbols.size, pch = group.symbols, col = "darkblue")
         },
       key = list(space = "top",
                  text = list(group.labels,col = "black"),
                  points = list(pch = group.symbols, cex = group.symbols.size,
                                col = "darkblue")))

# employ training-and-test regimen for model validation
set.seed(1234) # set seed for repeatability of training-and-test split
training_test <- c(rep(1,length=trunc((2/3)*nrow(dodgers))),
rep(2,length=(nrow(dodgers) - trunc((2/3)*nrow(dodgers)))))

dodgers$training_test <- sample(training_test) # random permutation
dodgers$training_test <- factor(dodgers$training_test,
                                levels=c(1,2), labels=c("TRAIN","TEST"))

dodgers.train <- subset(dodgers, training_test == "TRAIN")
print(str(dodgers.train)) # check training data frame
dodgers.test <- subset(dodgers, training_test == "TEST")
print(str(dodgers.test)) # check test data frame

# specify a simple model with bobblehead entered last
my.model <- {attend ~ ordered_month + ordered_day_of_week + bobblehead}
# fit the model to the training set
train.model.fit <- lm(my.model, data = dodgers.train)
# summary of model fit to the training set
print(summary(train.model.fit))

```

```

# training set predictions from the model fit to the training set
dodgers.train$predict_attend <- predict(train.model.fit)
# test set predictions from the model fit to the training set
dodgers.test$predict_attend <- predict(train.model.fit,
                                         newdata = dodgers.test)
# compute the proportion of response variance
# accounted for when predicting out-of-sample
cat("\n","Proportion of Test Set Variance Accounted for: ",
    round((with(dodgers.test,cor(attend,predict_attend)^2)),
          digits=3),"\n",sep=" ")
# merge the training and test sets for plotting
dodgers.plotting.frame <- rbind(dodgers.train,dodgers.test)

# generate predictive modeling visual for management
group.labels <- c("No Bobbleheads","Bobbleheads")
group.symbols <- c(21,24)
group.colors <- c("black","black")
group.fill <- c("black","red")
xyplot(predict.attend/1000 ~ attend/1000 | training_test,
       data = dodgers.plotting.frame, groups = bobblehead, cex = 2,
       pch = group.symbols, col = group.colors, fill = group.fill,
       layout = c(2, 1), xlim = c(20,65), ylim = c(20,65),
       aspect=1, type = c("p","g"),
       panel=function(x,y, ...)
         {panel.xyplot(x,y,...)
          panel.segments(25,25,60,60,col="black",cex=2)
        },

```

```
strip=function(...) strip.default(..., style=1),
xlab = "Actual Attendance (thousands)",
ylab = "Predicted Attendance (thousands)",
key = list(space = "top",
           text = list(rev(group.labels), col = rev(group.colors)),
           points = list(pch = rev(group.symbols),
                         col = rev(group.colors),
                         fill = rev(group.fill)))))

# use the full data set to obtain an estimate of the increase in
# attendance due to bobbleheads, controlling for other factors
my.model.fit <- lm(my.model, data = dodgers) # use all available data
print(summary(my.model.fit))

# tests statistical significance of the bobblehead promotion
# type I anova computes sums of squares for sequential tests
print(anova(my.model.fit))
cat("\n","Estimated Effect of Bobblehead Promotion on Attendance: ",
round(my.model.fit$coefficients[length(my.model.fit$coefficients)],
digits = 0),"\n",sep="")

# standard graphics provide diagnostic plots
plot(my.model.fit)
# additional model diagnostics drawn from the car package
library(car)
residualPlots(my.model.fit)
marginalModelPlots(my.model.fit)
print(outlierTest(my.model.fit))
```

```
# Predictive Model for Los Angeles Dodgers Promotion and Attendance (Python)

# prepare for Python version 3x features and functions
from __future__ import division, print_function
from future_builtins import ascii, filter, hex, map, oct, zip

# import packages for analysis and modeling
import pandas as pd # data frame operations
from pandas.tools.rplot import RPlot, TrellisGrid, GeomPoint,\n    ScaleRandomColour # trellis/lattice plotting
import numpy as np # arrays and math functions
from scipy.stats import uniform # for training-and-test split
import statsmodels.api as sm # statistical models (including regression)
import statsmodels.formula.api as smf # R-like model specification
import matplotlib.pyplot as plt # 2D plotting

# read in Dodgers bobbleheads data and create data frame
dodgers = pd.read_csv("dodgers.csv")
# examine the structure of the data frame
print("\nContents of dodgers data frame -----")
# attendance in thousands for plotting
dodgers['attend_000'] = dodgers['attend']/1000
# print the first five rows of the data frame
print(pd.DataFrame.head(dodgers))

mondays = dodgers[dodgers['day_of_week'] == 'Monday']
tuesdays = dodgers[dodgers['day_of_week'] == 'Tuesday']
wednesdays = dodgers[dodgers['day_of_week'] == 'Wednesday']
thursdays = dodgers[dodgers['day_of_week'] == 'Thursday']
fridays = dodgers[dodgers['day_of_week'] == 'Friday']
saturdays = dodgers[dodgers['day_of_week'] == 'Saturday']
sundays = dodgers[dodgers['day_of_week'] == 'Sunday']
```

```

# convert days' attendance into list of vectors for box plot
data = [mondays['attend_000'], tuesdays['attend_000'],
    wednesdays['attend_000'], thursdays['attend_000'],
    fridays['attend_000'], saturdays['attend_000'],
    sundays['attend_000']]
ordered_day_names = ['Mon', 'Tus', 'Wed', 'Thur', 'Fri', 'Sat', 'Sun']

# exploratory data analysis: box plot for day of the week
fig, axis = plt.subplots()
axis.set_xlabel('Day of Week')
axis.set_ylabel('Attendance (thousands)')
day_plot = plt.boxplot(data, sym='o', vert=1, whis=1.5)
plt.setp(day_plot['boxes'], color = 'black')
plt.setp(day_plot['whiskers'], color = 'black')
plt.setp(day_plot['fliers'], color = 'black', marker = 'o')
axis.set_xticklabels(ordered_day_names)
plt.show()
plt.savefig('fig_advert_promo_dodgers_eda_day_of_week_Python.pdf',
    bbox_inches = 'tight', dpi=None, facecolor='w', edgecolor='b',
    orientation='portrait', papertype=None, format=None,
    transparent=True, pad_inches=0.25, frameon=None)

april = dodgers[dodgers['month'] == 'APR']
may = dodgers[dodgers['month'] == 'MAY']
june = dodgers[dodgers['month'] == 'JUN']
july = dodgers[dodgers['month'] == 'JUL']
august = dodgers[dodgers['month'] == 'AUG']
september = dodgers[dodgers['month'] == 'SEP']
october = dodgers[dodgers['month'] == 'OCT']
data = [april['attend_000'], may['attend_000'],
    june['attend_000'], july['attend_000'],
    august['attend_000'], september['attend_000'],
    october['attend_000']]
ordered_month_names = ['April', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct']

```

```

fig, axis = plt.subplots()
axis.set_xlabel('Month')
axis.set_ylabel('Attendance (thousands)')
day_plot = plt.boxplot(data, sym='o', vert=1, whis=1.5)
plt.setp(day_plot['boxes'], color = 'black')
plt.setp(day_plot['whiskers'], color = 'black')
plt.setp(day_plot['fliers'], color = 'black', marker = 'o')
axis.set_xticklabels(ordered_month_names)
plt.show()
plt.savefig('fig_advert_promo_dodgers_eda_month_Python.pdf',
            bbox_inches = 'tight', dpi=None, facecolor='w', edgecolor='b',
            orientation='portrait', papertype=None, format=None,
            transparent=True, pad_inches=0.25, frameon=None)

# trellis/lattice plot attendance by temp, conditioning on skies
# and day_night with bubblehead NO/YES shown in distinct colors
plt.figure()
plot = RPlot(dodgers, x = 'temp', y = 'attend_000')
plot.add(TrellisGrid(['day_night', 'skies']))
plot.add(GeomPoint(colour = ScaleRandomColour('bubblehead')))
plot.render(plt.gcf())
plt.show()
plt.savefig('fig_advert_promo_dodgers_eda_many.pdf',
            bbox_inches = 'tight', dpi=None, facecolor='w', edgecolor='b',
            orientation='portrait', papertype=None, format=None,
            transparent=True, pad_inches=0.25, frameon=None)

# map day_of_week to ordered_day_of_week
day_to_ordered_day = {'Monday' : '1Monday',
                      'Tuesday' : '2Tuesday',
                      'Wednesday' : '3Wednesday',
                      'Thursday' : '4Thursday',
                      'Friday' : '5Friday',

```

```

'Saturday' : '6Saturday',
'Sunday' : '7Sunday'}
dodgers['ordered_day_of_week'] = dodgers['day_of_week'].map(day_to_ordered_day)

# map month to ordered_month
month_to_ordered_month = {'APR' : '1April',
                           'MAY' : '2May',
                           'JUN' : '3June',
                           'JUL' : '4July',
                           'AUG' : '5Aug',
                           'SEP' : '6Sept',
                           'OCT' : '7Oct'}
dodgers['ordered_month'] = dodgers['month'].map(month_to_ordered_month)

# employ training-and-test regimen for model validation
np.random.seed(1234)
dodgers['runiform'] = uniform.rvs(loc = 0, scale = 1, size = len(dodgers))
dodgers_train = dodgers[dodgers['runiform'] >= 0.33]
dodgers_test = dodgers[dodgers['runiform'] < 0.33]
# check training data frame
print('\ndodgers_train data frame (rows, columns): ',dodgers_train.shape)
print(dodgers_train.head())
# check test data frame
print('\ndodgers_test data frame (rows, columns): ',dodgers_test.shape)
print(dodgers_test.head())

# specify a simple model with bobblehead entered last
my_model = str('attend ~ ordered_month + ordered_day_of_week + bobblehead')

# fit the model to the training set
train_model_fit = smf.ols(my_model, data = dodgers_train).fit()
# summary of model fit to the training set
print(train_model_fit.summary())

```

```
# training set predictions from the model fit to the training set
dodgers_train['predict_attend'] = train_model_fit.fittedvalues

# test set predictions from the model fit to the training set
dodgers_test['predict_attend'] = train_model_fit.predict(dodgers_test)

# compute the proportion of response variance
# accounted for when predicting out-of-sample
print('\nProportion of Test Set Variance Accounted for: ',\
      round(np.power(dodgers_test['attend'].values -\
                      corr(dodgers_test['predict_attend']),2),3))

# use the full data set to obtain an estimate of the increase in
# attendance due to bobbleheads, controlling for other factors
my_model_fit = cmf.cls(my_model, data = dodgers).fit()
print(my_model_fit.summary())
print('\nEstimated Effect of Bobblehead Promotion on Attendance: ',\
      round(my_model_fit.params[13],0))
```

```

# Game-day Simulator for Baseball (R)
library(lattice) # graphics package for probability matrix visual
simulator <- function(home_mean,away_mean,niterations) {
  # input runs scored means, output probability of winning for home team
  set.seed(1234) # set to obtain reproducible results
  away_game_score <- numeric(niterations)
  home.game.score <- numeric(niterations)
  home.win <- numeric(niterations)
  i <- 1
  while (i < niterations + 1) {
    away_game_score[i] <- rnbinom(1,mu=away_mean, size = 4)
    home.game.score[i] <- rnbinom(1,mu=home_mean, size = 4)
    if(away_game_score[i] > home.game.score[i]) home.win[i] <- 1
    if(away_game_score[i] > home.game.score[i] || 
       away_game_score[i] < home.game.score[i]) i <- i + 1
  }
  n_home_win <- sum(home.win)
  n_home_win/niterations # return probability of away team winning
}
niterations <- 100000 # use smaller number for testing
# probability matrix for results... home team is rows, away team is columns
probmat <- matrix(data = NA, nrow = 9, ncol = 9,
  dimnames = list(c(as.character(1:9)), c(as.character(1:9))))
for (index_home in 1:9)
  for (index_away in 1:9)
    if (index_home != index_away) {
      probmat[index_home,index_away] <-
        simulator(index_home, index_away, niterations)

```

```
}

pdf(file = "fig_sports_analytics_prob_matrix.pdf", width = 8.5, height = 8.5)
x <- rep(1:nrow(probmat), times=ncol(probmat))
y <- NULL
for (i in 1:ncol(probmat)) y <- c(y,rep(i,times=nrow(probmat)))
probtext <- sprintf("%0.3f", as.numeric(probmat)) # fixed format 0.XXX
text_data_frame <- data.frame(x, y, probtext)
text_data_frame$probtext <- as.character(text_data_frame$probtext)
text_data_frame$probtext <- ifelse((text_data_frame$probtext == "NA"),
                                    NA, text_data_frame$probtext) # define diagonal cells as missing
text_data_frame <- na.omit(text_data_frame) # diagonal cells
print(levelplot(probmat, cuts = 25, tick.number = 9,
                 col.rampPalette=c("violet", "white", "light blue")),
      xlab = "Visiting Team Runs Expected",
      ylab = "Home Team Runs Expected",
      panel = function(...) {
        panel.levelplot(...)
        panel.text(text_data_frame$x, text_data_frame$y,
                   labels = text_data_frame$probtext)
      })
dev.off()
```

```
# Game-day Simulator for Baseball (Python)

from __future__ import division, print_function

import numpy as np
from scipy.stats import nbinom

def simulator(home_mean, away_mean, niterations):
    # estimates probability of home team win
    seed(1234) # set to obtain reproducible results
    home_game_score = [0] * niterations
    away_game_score = [0] * niterations
    home_win = [0] * niterations
    i = 0
    while (i < niterations):
        home_game_score[i] = \
            nbinom.rvs(n = 4.0, p = 4.0/(4.0 + home_mean), size = 1)[0]
        away_game_score[i] = \
            nbinom.rvs(n = 4.0, p = 4.0/(4.0 + away_mean), size = 1)[0]
        if (home_game_score[i] > away_game_score[i]):
            home_win[i] = 1
        if ((away_game_score[i] > home_game_score[i]) or \
            (away_game_score[i] < home_game_score[i])):
```

```
i = i + 1
n_home_win = sum(home_win)
return n_home_win / niterations

niterations = 100000 # use smaller number for testing
# probability matrix for results... home team rows, away team columns
probmat = array([[0.0] * 9] * 9)

# matrix representation of home and away team runs for table
homemat = array([[9] * 9, [8] * 9, [7] * 9, [6] * 9, [5] * 9,\ 
    [4] * 9, [3] * 9, [2] * 9, [1] * 9])
awayrow = array([1, 2, 3, 4, 5, 6, 7, 8, 9])
awaymat = array([awayrow] * 9)

# generate table of probabilities
for index_home in range(9):
    for index_away in range(9):
        if (homemat[index_home, index_away] != awaymat[index_home, index_away]):
            print(index_home, index_away)
            probmat[index_home, index_away] = \
                simulator(float(homemat[index_home, index_away]), \
                float(awaymat[index_home, index_away]), niterations)

print(probmat)
```

```
# Simple One-Site Web Crawler and Scraper (Python)
#
# prepare for Python version 3x features and functions
from __future__ import division, print_function

# scrapy documentation at http://doc.scrapy.org/

# workspace directory set to outer folder/directory wnde_chapter_3b
# the operating system commands in this example are Mac OS X

import scrapy # object-oriented framework for crawling and scraping
import os # operating system commands

# function for walking and printing directory structure
def list_all(current_directory):
    for root, dirs, files in os.walk(current_directory):
        level = root.replace(current_directory, '').count(os.sep)
        indent = ' ' * 4 * (level)
        print('{}{}'.format(indent, os.path.basename(root)))
        subindent = ' ' * 4 * (level + 1)
        for f in files:
            print('{}{}'.format(subindent, f))

# initial directory should have this form (except for items beginning with .):
#   sads_exhibit_11_1
#       run_one_site_crawler.py
#       scrapy.cfg
#       scrapy_application/
#           __init__.py
#           items.py
#           pipelines.py
#           settings.py
```

```
#           spiders
#               __init__.py
#               one_site_crawler.py

# examine the directory structure
current_directory = os.getcwd()
list_all(current_directory)

# list the available spiders, showing names to be used for crawling
os.system('scrapy list')

# decide upon the desired format for exporting output: csv, JSON, or XML.

# run the scraper exporting results as a JSON text file items.jsonlines
# this file provides text information with linefeeds to provide
# text output that is easily readable in a plain text editor
os.system('scrapy crawl TOUTBAY -o items.jsonlines')

# output format commented out (choose the one needed for further parsing work)
# run the scraper exporting results as a comma-delimited text file items.csv
# os.system('scrapy crawl TOUTBAY -o items.csv')

# run the scraper exporting results as a JSON text file items.json
# os.system('scrapy crawl TOUTBAY -o items.json')

# run the scraper exporting results as a dictionary XML text file items.xml
# os.system('scrapy crawl TOUTBAY -o items.xml')
```

```
# -----
# MyItem class defined by
# items.py
# -----
# location in directory structure:
# sads_exhibit_11_1/scrapy_application/items.py

# establishes data fields for scraped items

import scrapy # object-oriented framework for crawling and scraping

class MyItem(scrapy.item.Item):
    # define the data fields for the item (just one field used here)
    paragraph = scrapy.item.Field() # paragraph content

# -----
# MyPipeline class defined by
# pipelines.py
# -----
# location in directory structure:
# sads_exhibit_11_1/scrapy_application/pipelines.py

class MyPipeline(object):
    def process_item(self, item, spider):
        return item
```

```
# -----
# settings for scrapy.cfg
# settings.py
# -----
# location in directory structure:
# sads_exhibit_11_1/scrapy_application/settings.py

BOT_NAME = 'MyBot'
BOT_VERSION = '1.0'

SPIDER_MODULES = ['scrapy_application.spiders']
NEWSPIDER_MODULE = 'scrapy_application.spiders'
USER_AGENT = '%s/%s' % (BOT_NAME, BOT_VERSION)
COOKIES_ENABLED = False
DOWNLOAD_DELAY = 2
RETRY_ENABLED = False
DOWNLOAD_TIMEOUT = 15
REDIRECT_ENABLED = False
DEPTH_LIMIT = 50

# -----
# spider class defined by
# script one_site_crawler.py
# -----
# location in directory structure:
# sads_exhibit_11_1/scrapy_application/spiders/one_site_crawler.py

# prepare for Python version 3x features and functions
from __future__ import division, print_function
```

```
# each spider class gives code for crawling and scraping

import scrapy # object-oriented framework for crawling and scraping
from scrapy_application.items import MyItem # item class
from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor

# spider subclass inherits from BaseSpider
# this spider is designed to crawl just one website
class MySpider(CrawlSpider):
    name = "TOUTBAY" # unique identifier for the spider
    allowed_domains = ['toutbay.com'] # limits the crawl to this domain list
    start_urls = ['http://www.toutbay.com'] # first url to crawl in domain

    # define the parsing method for the spider
    def parse(self, response):
        html_scraper = scrapy.selector.HtmlXPathSelector(response)
        divs = html_scraper.select('//div') # identify all <div> nodes
        # XPath syntax to grab all the text in paragraphs in the <div> nodes
        results = [] # initialize list
        this_item = MyItem() # use this item class
        this_item['paragraph'] = divs.select('.//p').extract()
        results.append(this_item) # add to the results list
        return results
```

```
# Gathering Opinion Data from Twitter: Football Injuries (Python)

# prepare for Python version 3x features and functions
from __future__ import division, print_function

import twitter # work with Twitter APIs
import json # methods for working with JSON data

windows_system = False # set to True if this is a Windows computer
if windows_system:
    line_termination = '\r\n' # Windows line termination
else (windows_system == False):
    line_termination = '\n' # Unix/Linus/Mac line termination

# name used for JSON file storage
json_filename = 'my_tweet_file.json'

# name for text file for review of results
full_text_filename = 'my_tweet_review_file.txt'

# name for text from tweets
partial_text_filename = 'my_tweet_text_file.txt'

# See Russell (2014) and Twitter site for documentation
# https://dev.twitter.com/rest/public
# Go to http://twitter.com/apps/new to provide an application name
# to Twitter and to obtain OAuth credentials to obtain API data

# -----
# Twitter authorization a la Russell (2014) section 9.1
# Insert credentials in place of the "blah blah blah" strings
# Sample usage of oauth() function
# twitter_api = oauth_login()
def oauth_login():
```

```
CONSUMER_KEY = 'blah'
CONSUMER_SECRET = 'blah blah'
OAUTH_TOKEN = 'blah blah blah'
OAUTH_TOKEN_SECRET = 'blah blah blah blah'

auth = twitter.oauth.OAuth(OAUTH_TOKEN, OAUTH_TOKEN_SECRET,
                           CONSUMER_KEY, CONSUMER_SECRET)

twitter_api = twitter.Twitter(auth=auth)
return twitter_api

# -----
# searching the RFST API à la Russell (2014) section 9.4
def twitter_search(twitter_api, q, max_results=200, **kw):
    # See https://dev.twitter.com/docs/api/1.1/get/search/tweets and
    # https://dev.twitter.com/docs/using-search for details on advanced
    # search criteria that may be useful for keyword arguments

    # See https://dev.twitter.com/docs/api/1.1/get/search/tweets
    search_results = twitter_api.search.tweets(q=q, count=100, **kw)

    statuses = search_results['statuses']

    # iterate through batches of results by following the cursor until we
    # reach the desired number of results, keeping in mind that OAuth users
    # can "only" make 180 search queries per 15-minute interval. See
    # https://dev.twitter.com/docs/rate-limiting/1.1/limits
    # for details. A reasonable number of results is 1000, although
    # that number of results may not exist for all queries.

    # Enforce a reasonable limit
    max_results = min(1000, max_results)
```

```

for _ in range(10): # 10*100 = 1000
    try:
        next_results = search_results['search_metadata']['next_results']
    except KeyError, e: # No more results when next_results doesn't exist
        break

    # Create a dictionary from next_results, which has the following form:
    # ?max_id=313519052523986943&q=NCAA&include_entities=1
    kwargs = dict([ kv.split('=')
                    for kv in next_results[1:].split("&") ])

    search_results = twitter_api.search.tweets(**kwargs)
    statuses += search_results['statuses']

    if len(statuses) > max_results:
        break

return statuses

```

use the predefined functions from Russell to conduct the search
here we see what people are saying about football injuries

```

twitter_api = oauth_login()
print(twitter_api) # verify the connection

# 'football injuries' # one of many possible search strings
results = twitter_search(twitter_api, q, max_results = 200) # limit to 200 tweets

examining the results object... should be list of dictionary objects
print('\n\n\ttype of results:', type(results))
print('\n\tnumber of results:', len(results))
print('\n\ttype of results elements:', type(results[0]))

```

```
# -----
# working with JSON files composed of multiple JSON objects
# results is a list of dictionary items obtained from twitter
# these functions assume that each dictionary item
# is written as a JSON object on a separate line
item_count = 0 # initialize count of objects dumped to file
with open(json_filename, 'w') as outfile:
    for dict_item in results:
        json.dump(dict_item, outfile, encoding = 'utf-8')
        item_count = item_count + 1
        if item_count < len(results):
            outfile.write(line_termination) # new line between items

# -----
# working with text file for reviewing multiple JSON objects
# this text file will show the full contents of each tweet
# results is a list of dictionary items obtained from twitter
# these functions assume that each dictionary item
# is written as group of lines printed with indentation
item_count = 0 # initialize count of objects dumped to file
with open(full_text_filename, 'w') as outfile:
    for dict_item in results:
        outfile.write('Item index: ' + str(item_count) + \
                     ' -----' + line_termination)
        # indent for pretty printing
        outfile.write(json.dumps(dict_item, indent = 4))
        item_count = item_count + 1
        if item_count < len(results):
            outfile.write(line_termination) # new line between items
```

```
# -----
# working with text file for reviewing text from multiple JSON objects
# this text file will show only the text from each tweet
# results is a list of dictionary items obtained from twitter
# those functions assume that the text of each tweet
# is written to a separate line in the output text file
item_count = 0 # initialize count of objects dumped to file
with open(partial_text_filename, 'w') as outfile:
    for dict_item in results:
        outfile.write(json.dumps(dict_item['text']))
        item_count = item_count + 1
        if item_count < len(results):
            outfile.write(line_termination) # new line between text items
```

```

# The Anscombe Quartet (Python)
# demonstration data from
# Anscombe, F. J. 1973, February. Graphs in statistical analysis.
# The American Statistician 27: 1721.

# prepare for Python version 3x features and functions
from __future__ import division, print_function

# import packages for Anscombe Quartet demonstration
import pandas as pd # data frame operations
import numpy as np # arrays and math functions
import statsmodels.api as sm # statistical models (including regression)
import matplotlib.pyplot as plt # 2D plotting

# define the anscombe data frame using dictionary of equal-length lists
anscombe = pd.DataFrame({'x1' : [10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5],
    'x2' : [10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5],
    'x3' : [10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5],
    'x4' : [8, 8, 8, 8, 8, 8, 19, 8, 8, 8],
    'y1' : [9.04, 6.95, 7.58, 8.01, 8.33, 9.96, 7.24, 4.25, 10.84, 4.82, 5.68],
    'y2' : [9.14, 8.14, 8.74, 8.77, 9.26, 8.1, 8.13, 3.1, 9.13, 7.26, 4.74],
    'y3' : [7.46, 6.77, 12.74, 7.11, 7.81, 8.84, 6.08, 5.39, 8.15, 6.42, 5.73],
    'y4' : [5.58, 5.76, 7.71, 8.84, 8.47, 7.04, 5.25, 12.5, 5.56, 7.91, 5.89]})

# fit linear regression models by ordinary least squares
set_I_design_matrix = sm.add_constant(anscombe['x1'])
set_I_model = sm.OLS(anscombe['y1'], set_I_design_matrix)
print(set_I_model.fit().summary())

set_II_design_matrix = sm.add_constant(anscombe['x2'])
set_II_model = sm.OLS(anscombe['y2'], set_II_design_matrix)
print(set_II_model.fit().summary())

```

```
set_III_design_matrix = sm.add_constant(anscombe['x3'])
set_III_model = sm.OLS(anscombe['y3'], set_III_design_matrix)
print(set_III_model.fit().summary())

set_IV_design_matrix = sm.add_constant(anscombe['x4'])
set_IV_model = sm.OLS(anscombe['y4'], set_IV_design_matrix)
print(set_IV_model.fit().summary())

# create scatter plots
fig = plt.figure()
set_I = fig.add_subplot(2, 2, 1)
set_I.scatter(anscombe['x1'],anscombe['y1'])
set_I.set_title('Set I')
set_I.set_xlabel('x1')
set_I.set_ylabel('y1')
set_I.set_xlim(2, 20)
set_I.set_ylim(2, 14)

set_II = fig.add_subplot(2, 2, 2)
set_II.scatter(anscombe['x2'],anscombe['y2'])
set_II.set_title('Set II')
set_II.set_xlabel('x2')
set_II.set_ylabel('y2')
set_II.set_xlim(2, 20)
set_II.set_ylim(2, 14)

set_III = fig.add_subplot(2, 2, 3)
set_III.scatter(anscombe['x3'],anscombe['y3'])
set_III.set_title('Set III')
set_III.set_xlabel('x3')
set_III.set_ylabel('y3')
set_III.set_xlim(2, 20)
set_III.set_ylim(2, 14)
```

```
set_IV = fig.add_subplot(2, 2, 4)
set_IV.scatter(anscombe['x4'],anscombe['y4'])
set_IV.set_title('Set IV')
set_IV.set_xlabel('x4')
set_IV.set_ylabel('y4')
set_IV.set_xlim(2, 20)
set_IV.set_ylim(2, 14)

plt.subplots_adjust(left=0.1, right=0.925, top=0.925, bottom=0.1,
                   wspace = 0.3, hspace = 0.4)
plt.savefig('fig_anscombe_Python.pdf', bbox_inches = 'tight', dpi=None,
            facecolor='w', edgecolor='b', orientation='portrait', papertype=None,
            format=None, transparent=True, pad_inches=0.25, frameon=None)

# Suggestions for the student:
# See if you can develop a quartet of your own,
# or perhaps just a duet, two very different data sets
# with the same fitted model.
```

```

# The Anscombe Quartet (R)

# demonstration data from
# Anscombe, F. J. 1973, February. Graphs in statistical analysis.
# The American Statistician 27: 1721.

# define the anscombe data frame
anscombe <- data.frame(
  x1 = c(10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5),
  x2 = c(10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5),
  x3 = c(10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5),
  x4 = c(8, 0, 8, 8, 0, 19, 0, 0, 8, 8),
  y1 = c(8.04, 9.95, 7.58, 8.81, 8.33, 9.96, 7.24, 4.26, 10.84, 4.82, 5.68),
  y2 = c(9.14, 8.14, 8.74, 8.77, 9.26, 8.1, 6.13, 3.1, 9.13, 7.26, 4.74),
  y3 = c(7.46, 6.77, 12.74, 7.11, 7.81, 8.84, 6.08, 5.39, 8.15, 6.42, 5.73),
  y4 = c(6.58, 5.76, 7.71, 8.84, 8.47, 7.04, 5.25, 12.5, 5.56, 7.91, 5.89))

# show results from four regression analyses
with(anscombe, print(summary(lm(y1 ~ x1, data = anscombe))))
with(anscombe, print(summary(lm(y2 ~ x2, data = anscombe))))
with(anscombe, print(summary(lm(y3 ~ x3, data = anscombe))))
with(anscombe, print(summary(lm(y4 ~ x4, data = anscombe)))))

# place four plots on one page using standard R graphics
# ensuring that all have the same scales
# for horizontal and vertical axes
pdf(file = "fig_anscombe_R.pdf", width = 8.5, height = 8.5)
par(mfrow=c(2,2), mar=c(5.1, 4.1, 4.1, 2.1))
with(anscombe, plot(x1, y1, xlim=c(2,20), ylim=c(2,14), pch = 19,
  col = "darkblue", cex = 1.5, las = 1, xlab = "x1", ylab = "y1"))
title("Set I")
with(anscombe, plot(x2, y2, xlim=c(2,20), ylim=c(2,14), pch = 19,
  col = "darkblue", cex = 1.5, las = 1, xlab = "x2", ylab = "y2"))

```

```
title("Set II")
with(amscombe,plot(x3, y3, xlim=c(2,20), ylim=c(2,14), pch = 19,
    col = "darkblue", cex = 1.5, las = 1, xlab = "x3", ylab = "y3"))
title("Set III")
with(amscombe,plot(x4, y4, xlim=c(2,20), ylim=c(2,14), pch = 19,
    col = "darkblue", cex = 1.5, las = 1, xlab = "x4", ylab = "y4"))
title("Set IV")
dev.off()

# par(mfrow=c(1,1),mar=c(5.1, 4.1, 4.1, 2.1)) # return to plotting defaults

# Suggestions for the student:
# See if you can develop a quartet of your own,
# or perhaps just a duet, two very different data sets
# with the same fitted model.
```

```
# Making Differential Runs Plots for Baseball (R)

# data visualization with R standard graphics

excess.runs.data.frame.ARI <- read.csv("MLB_2007_ARI_data_frame.csv")

pdf(file=paste("fig_differential_runs_ARI.pdf",sep=""),
     width = 11, height = 8.5)
par(mfrow=c(1,1), xpd=NA, tex=1)

plot(excess.runs.data.frame.ARI$excess.runs, type="h",
      las=1, xlab="Game Number and Opponent", ylab="Differential Runs",
      ylim=c(-(max(abs(excess.runs.data.frame.ARI$excess.runs),14) + 1),
             (max(abs(excess.runs.data.frame.ARI$excess.runs),14) + 1)))
abline(h=0,lty="solid",xpd=FALSE,lwd=.5)

legend("topright", title=NULL,
       legend=c("Home Game ","Away Game "), pch=c(19,21))

# plot all as open circles first
points(excess.runs.data.frame.ARI$excess.runs, pch=21)

# then fill in the circles for the home games
for(i in seq(along=excess.runs.data.frame.ARI$game.number))
```

```
if(excess.runs.data.frame.AR1$home.away[1]=="home")
points(excess.runs.data.frame.AR1$game.number[i],
       excess.runs.data.frame.AR1$excess.runs[1], pch=19)

for(i in seq(along=excess.runs.data.frame.AR1$game.number))
text(excess.runs.data.frame.AR1$game.number[i],-
     (max(abs(excess.runs.data.frame.AR1$excess.runs),15) + 1),
     excess.runs.data.frame.AR1$other.team.label[i], cex=.85, pos=4, srt=90)

text(excess.runs.data.frame.AR1$game.number[1],
     (max(abs(excess.runs.data.frame.AR1$excess.runs),14) + 1),
     map.team.to.name(excess.runs.data.frame.AR1$this.team[1]),cex=2,pos=4)

dev.off()
```

```
# Moving Fraction Plot: A Basketball Example (R)

library(grid) # graphics utilities needed for split-plotting
library(ggplot2) # graphics package with ribbon plot

# read game summary data for the Oklahoma City Thunder
OKC_data <- read.csv("okc_data_2014_2015.csv", stringsAsFactors = FALSE)

# check the structure of the data
print(str(OKC_data))

# create winning fraction plot to explore streakiness during the season
# a window of twelve games is used following Albert and Bennett (2001)

# add game number to data frame
OKC_data$game_number <- seq(1:nrow(OKC_data))

# set binary indicator if this team has won
OKC_data$win_bin <- rep(0, length = nrow(OKC_data))
for (i in seq(along = OKC_data$win_bin)) {
    if (OKC_data$win_team_code[i] == "OKC")
        OKC_data$win_bin[i] <- 1

# window across the season twelve games at a time
OKC_data$win_window <- rep(NA, length = nrow(OKC_data))
for (i in seq(along = OKC_data$win_window)) {
    if (i > 6)
        OKC_data$win_window[i] <-
            mean(OKC_data$win_bin[(i - 6):(i + 5)]) * 100
```

```
}

moving_fraction_plotting_frame <-
  OKC_data[, c("game_number", "win_window")]
moving_fraction_plotting_frame$ymax_topwhite <-
  rep(NA, length = nrow(moving_fraction_plotting_frame))
moving_fraction_plotting_frame$ymax_bottomwhite <-
  rep(NA, length = nrow(moving_fraction_plotting_frame))
moving_fraction_plotting_frame$ymin_topwhite <-
  rep(NA, length = nrow(moving_fraction_plotting_frame))
moving_fraction_plotting_frame$ymin_bottomwhite <-
  rep(NA, length = nrow(moving_fraction_plotting_frame))

for (i in seq(along = moving_fraction_plotting_frame$win_window)) {
  if (is.na(moving_fraction_plotting_frame$win_window[i])) {
    moving_fraction_plotting_frame$ymax_topwhite[i] <- 100
    moving_fraction_plotting_frame$ymin_topwhite[i] <- 50.1
    moving_fraction_plotting_frame$ymax_bottomwhite[i] <- 49.99
    moving_fraction_plotting_frame$ymin_bottomwhite[i] <- 0
  }

  if (!is.na(moving_fraction_plotting_frame$win_window[i])) {
    if (moving_fraction_plotting_frame$win_window[i] > 50) {
      moving_fraction_plotting_frame$ymax_topwhite[i] <- 100
      moving_fraction_plotting_frame$ymin_topwhite[i] <-
        moving_fraction_plotting_frame$win_window[i]
      moving_fraction_plotting_frame$ymax_bottomwhite[i] <- 49.99
      moving_fraction_plotting_frame$ymin_bottomwhite[i] <- 0
    }
  }
}
```

```

if (moving_fraction_plotting_frame$win_window[i] <= 50) {
  moving_fraction_plotting_frame$ymax_topwhite[i] <- 100
  moving_fraction_plotting_frame$ymin_topwhite[i] <- 50.1
  moving_fraction_plotting_frame$ymax_bottomwhite[i] <-
    moving_fraction_plotting_frame$win_window[i]
  moving_fraction_plotting_frame$ymin_bottomwhite[i] <- 0
}
}

greenmin <- 50.01
greenmax <- 100
redmin <- 0
redmax <- 49.99

pdf(file = "fig_moving_fraction_plot.pdf", width = 8.8, height = 8.5)
ggobject <- ggplot() +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=greenmin, ymax=greenmax),
  stat="identity", colour="white", fill="darkgreen") +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=redmin, ymax=redmax),
  stat="identity", colour="white", fill="darkred") +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=ymin_topwhite, ymax=ymax_topwhite),
  stat="identity", colour="white", fill="white") +
  geom_ribbon(data=moving_fraction_plotting_frame,
  mapping=aes(x=game_number, ymin=ymin_bottomwhite, ymax=ymax_bottomwhite),
  stat="identity", colour="white", fill="white") +
  annotate("segment", x = 0, xend = 82, y = 49.99, yend = 50.01) +
  ylab("Moving Fraction (Winning Percentage)") +
  xlab("Sequence of Games")

print(ggobject)

dev.off()

```

```
# Visualizing Basketball Games (R)

library(lubridate) # data/time functions and lakers data frame
# lubridate imports plyr package
library(ggplot2) # statistical graphics, grid graphics assumed

# functions used with grid graphics to split the plotting region
# to set margins and to plot more than one ggplot object on one page/screen
vlayout <- function(x, y)
  viewport(layout.pos.row=x, layout.pos.col=y)

# user-defined function to plot a ggplot object with margins
ggplot.print.with.margins <- function(ggplot.object.name,
  left.margin.pct=10,
  right.margin.pct=10,top.margin.pct=10,bottom.margin.pct=10)
{
  # begin function for printing ggplot objects with margins
  # margins expressed as percentages of total... use integers
  grid.newpage()
  pushViewport(viewport(layout=grid.layout(100,100)))
  print(ggplot.object.name,
    vp=vlayout((0 + top.margin.pct):(100 - bottom.margin.pct),
    (0 + left.margin.pct):(100 - right.margin.pct)))
}

# end function for printing ggplot objects with margins

# lakers data frame from lubridate includes
# play-by-play data for the Los Angeles Lakers 2008-2009 season
# original data from Parker (2010) http://www.basketballgeek.com/data/
# date: year/month/day text to be converted to date object
# opponent: three-character abbreviation for other team
# game_type: home or away for Los Angeles Lakers
# time: clock time remaining minutes:seconds converted to duration
# period: period of play 1, 2, 3, or 4
# ctypc: one of ten event types: foul, free throw, jump ball,
#        rebound, shot, sub, timeout, turnover, violation
```

```
# team: three-character abbreviation for team
#      or OFF for neither team as with a jump ball
# player: one of 371 players involved in Los Angeles Lakers games
# result: missed or made
# points: points scored
# type: one of 74 event descriptions: hook, off, layup,
#       shooting, personal, jump, pullup jump,
#       def, driving layup, driving finger roll layup, regular,
#       offensive, 3pt, turnaround jump, putback layup,
#       slam dunk, tip, dunk, defensive goaltending,
#       hook bank, running layup, official, driving slam dunk,
#       short, driving reverse layup, kicked ball, putback dunk,
#       technical, alley oop dunk, turnaround fade away, running jump,
#       delay of game, defense 3 second, fade away bank, floating jump,
#       driving dunk, loose ball, running bank, running dunk,
#       fade away jumper, finger roll layup, turnaround hook, reverse layup,
#       reverse dunk, jump hook, jump bank, double technical,
#       running hook, driving jump, turnaround bank, stop back jump,
#       turnaround bank hook, pullup bank, alley oop layup, putback slam dunk,
#       flagrant type 1, running reverse layup, running finger roll layup,
#       reverse slam dunk, hanging technical, running slam dunk,
#       driving bank hook, jump ball, away from play, double personal,
#       driving bank, running bank hook, driving hook, lane, clear path,
#       jump bank hook, flagrant type 2, inbound
# x, y: shot location, standing behind the offensive teams hoop then
#       the X axis runs from left to right and the Y axis runs from
#       bottom to top. The center of the hoop is located at (25, 5.25).
```

```
# 2008-2009 team abbreviations/code names
team_code_2008 = c("ATL", "BOS", "CHA", "CHI", "CLE",
                  "DAL", "DEN", "DET", "GSW", "HOU",
                  "IND", "LAC", "LAL", "MEM", "MIA",
                  "MIL", "MIN", "NJN", "NOH", "NYK",
                  "OKC", "ORL", "PHI", "PHX", "POR",
                  "SAC", "SAS", "TOR", "UTA", "WAS")

# Note. Charlotte Bobcats later became the Charlotte Hornets
# New Orleans Hornets later became the New Orleans Pelicans
team_name_2008 = c("Atlanta Hawks", "Boston Celtics",
                  "Charlotte Hornets", "Chicago Bulls",
                  "Cleveland Cavaliers", "Dallas Mavericks",
                  "Denver Nuggets", "Detroit Pistons",
                  "Golden State Warriors", "Houston Rockets",
                  "Indiana Pacers", "Los Angeles Clippers",
                  "Los Angeles Lakers", "Memphis Grizzlies",
                  "Miami Heat", "Milwaukee Bucks",
                  "Minnesota Timberwolves", "New Jersey Nets",
                  "New Orleans Hornets", "New York Knicks",
                  "Oklahoma City Thunder", "Orlando Magic",
                  "Philadelphia 76ers", "Phoenix Suns",
                  "Portland Trail Blazers", "Sacramento Kings",
                  "San Antonio Spurs", "Toronto Raptors",
                  "Utah Jazz", "Washington Wizards")

# Note. Charlotte Bobcats became the Charlotte Hornets
# New Orleans Hornets became the New Orleans Pelicans
# New Jersey Nets became the Brooklyn Nets
```

```
# lakers$date <- ymd(lakers$date) # code as date variable
lakers$time <- ms(lakers$time) # code as period object
lakers$timc <- as.duration(lakers$time) # code as durations

# convert time to gametime with periods of 12 minutes, overtime 5 minutes
# here we convert to minutes and fractions of minutes
lakers$gametime <- dminutes(c(12, 24, 36, 48, 53)[lakers$period]) -
  as.duration(lakers$timc)
lakers$minutes <- as.numeric(seconds(lakers$gametime))/60
print(str(lakers)) # examine stucture of the data frame
```

```

# get rid of observations with team OFF (jump ball)
lakers_games <- lakers[lakers$Team != "OFF",]

# route plots to external plotting device... pdf file
pdf(file = "plot_lakers_basketball_2008_2009.pdf",
     width = 8.5, height = 8.5)
# -----
# cycle through all Lakers games for the season
gamedate <- unique(lakers_games$date)
for (igame in seq(along = gamedate)) { # begin for-loop for all games
  this_game <- lakers_games[lakers_games$date == gamedate[igame],]

# work with the current game, compute score as cumulative sum for each team
# using ddply function from the plyr package, required for lubridate package
  this_game_scores <- ddply(this_game, "team", transform,
    score = cumsum(points))

# identify team names and scores for this game
  first_team_name <- team_name_2008[which(team_code_2008 == "LAL")]
  second_team_name <- team_name_2008[which(team_code_2008 ==
    this_game_scores$opponent[1])]
  first_team_score <-
    max(this_game_scores[(this_game_scores$team == "LAL"), "score"])
  second_team_score <-
    max(this_game_scores[(this_game_scores$team ==
      this_game_scores$opponent[1]), "score"])

# summary for this game to be used in plot title
  this_game_summary_text <-
    paste0(gsub(" UTC","", ymd(this_game_scores$date)[1]),
      " ", first_team_name, " ", this_game_scores$game_type[1],
      " (", first_team_score, ") versus ",
      second_team_name, " ", "(", second_team_score, ") ", sep = "")

```

```
# create visualization for this game with annotation shading
# for the second and fourth 12-minute periods in the game
# and with customized legend for the step function score lines
ggplot_object <-
  ggplot(data = subset(this_game_scores, team == "LAL"),
  aes(x = minutes, y = score)) +
  layer(geom = "step", size = 0.75, colour = "purple") +
  layer(data = subset(this_game_scores,
    team == this_game_scores$opponent[1]),
    geom = "step", size = 1.5, colour = "blue") +
  annotate("rect", xmin = 12.01, xmax = 24.00,
    ymin = min(this_game_scores$score),
    ymax = max(this_game_scores$score),
    alpha = 0.1, fill = "black") +
  annotate("rect", xmin = 36.01, xmax = 48.00,
    ymin = min(this_game_scores$score),
    ymax = max(this_game_scores$score),
    alpha = 0.1, fill = "black") +
  annotate("rect", xmin = 0, xmax = 24.00,
    ymin = max(this_game_scores$score) - 15,
    ymax = max(this_game_scores$score),
    alpha = 1, colour = "darkgrey", fill = "white") +
  geom_segment(x = 1.0, xend = 3.0,
    y = max(this_game_scores$score) - 5,
    yend = max(this_game_scores$score) - 5,
    colour = "purple", size = 0.75) +
  geom_segment(x = 1.0, xend = 3.0,
    y = max(this_game_scores$score) - 10,
    yend = max(this_game_scores$score) - 10,
```

```
    colour = "blue", size = 1.5) +
  annotate("text", x = 4, y = max(this_game_scores$score) - 5,
    colour = "black", alpha = 0.9, size = 4.5,
    label = first_team_name, hjust = 0) +
  annotate("text", x = 4, y = max(this_game_scores$score) - 10,
    colour = "black", alpha = 0.9, size = 4.5,
    label = second_team_name, hjust = 0) +
  xlab("Game Minutes") + ylab("Score") +
  ggtitle(this_game_summary_text) +
  theme(axis.title.x = element_text(size = rel(1.25))) +
  theme(axis.title.y = element_text(size = rel(1.25)))

ggplot.print.with.margins(ggplot.object,
  left.margin.pct = 5, right.margin.pct = 5,
  top.margin.pct = 5, bottom.margin.pct = 5)

} # end for-loop for all games
dev.off() # closer the pdf plotting device
```

```
# Seeing Data Science as an Eclectic Discipline (R)

# Program for multidimensional scaling of ten academic disciplines

library(MASS) # includes functions for multidimensional scaling
library(wordcloud) # textplot utility to avoid overlapping text

USE_METRIC_MDS <- FALSE # metric versus non-metric toggle

# define a utility function for converting a distance structure
# to a distance matrix as required for some routines and
# for printing of the complete matrix for visual inspection.
make.distance.matrix <- function(distance.structure)
{
  n <- attr(distance.structure, "Size")
  full <- matrix(0,n,n)
  full[lower.tri(full)] <- distance.structure
  full+t(full)
}

# enter data into a distance structure as required for various
# distance-based routines. That is, we enter the upper triangle
# of the distance matrix as a single vector of distances
distance.structure <-
  as.single(c(35, 45, 40, 43, 18, 32, 3, 20, 19,
           8, 12, 9, 33, 30, 25, 7, 21,
           1, 15, 16, 13, 41, 17, 10,
           28, 36, 31, 42, 5, 26,
           39, 38, 37, 4, 44,
           2, 11, 34, 24,
           14, 29, 6,
           23, 22,
           27))
```

```

# provide a character vector of discipline names
disciplines <- c("Marketing", "Operations Research", "Mathematics",
  'Statistics', "Computer Science", 'Accounting', "Finance",
  "Management", "Data Science", "Economics")

attr(distance_structure, "Size") <- length(disciplines) # set size attribute

# check to see that the distance structure has been entered correctly
# by converting the distance structure to a distance matrix
# using the utility function make.distance.matrix, which we had defined
distance_matrix <- unlist(make.distance.matrix(distance_structure))
cat("\n","Distance Matrix of Academic Disciplines","\n")
print(distance_matrix)

if (USE_METRIC_MDS)
{
  # apply the metric multidimensional scaling algorithm and plot the map
  mds_solution <- cmdscale(distance_structure, k=2, eig=T)
  First_Dimension <- mds_solution$points[,1]
  Second_Dimension <- mds_solution$points[,2]
}

# apply the non-metric multidimensional scaling algorithm
# this is more appropriate for rank-order data
# and provides a more satisfactory solution here

if (!USE_METRIC_MDS)
{
  mds_solution <- isoMDS(distance_matrix, k = 2, trace = FALSE)
}

```

```

pdf(file = "plot_nonmetric.mds.data.science.pdf",
  width=11, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))

# original solution
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2]

# set up the plot but do not plot points... use names for points
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) # first page of pdf plots
# We plot the city names in the locations where points normally go.
text(First_Dimension, Second_Dimension, labels = disciplines,
     offset = 0.0, cex = 1.5)
title("Academic Disciplines (initial solution)")

# reflect the horizontal dimension
# multiply the first dimension by -1 to get reflected image
First_Dimension <- mds_solution$points[,1] * -1
Second_Dimension <- mds_solution$points[,2]
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) # second page of pdf plots
text(First_Dimension, Second_Dimension, labels = disciplines,
     offset = 0.0, cex = 1.5)
title("Academic Disciplines (horizontal reflection)")

# reflect the vertical dimension
# multiply the second dimension by -1 to get reflected image
First_Dimension <- mds_solution$points[,1]
Second_Dimension <- mds_solution$points[,2] * -1
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-25, 25), ylim = c(-25, 25)) # third page of pdf plots

```

```

text(First_Dimension, Second_Dimension, labels = disciplines,
    offset = 0.0, cex = 1.5)
title("Academic Disciplines (vertical reflection)")
# multiply the first and second dimensions by -1
# for reflection in both horizontal and vertical directions
First_Dimension <- mds_eclution$points[,1] * -1
Second_Dimension <- mds_solution$points[,2] * -1
plot(First_Dimension, Second_Dimension, type = "n", cex = 1.5,
      xlim = c(-26, 26), ylim = c(-26, 26)) # fourth page of pdf plots
text(First_Dimension, Second_Dimension, labels = disciplines,
    offset = 0.0, cex = 1.5)
title("Academic Disciplines (horizontal and vertical reflection)")
dev.off() # closes the pdf plotting device

pdf(file = "plot_pretty_criginal_mds_data_science.pdf",
     width=8.5, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] # no reflection
Second_Dimension <- mds_solution$points[,2] # no reflection
# wordcloud utility for plotting with no overlapping text
textplot(x = First_Dimension,
         y = Second_Dimension,
         words = disciplines,
         show.lines = FALSE,
         xlim = c(-28, 28), # extent of horizontal axis range
         ylim = c(-20, 20), # extent of vertical axis range
         xaxt = "n", # suppress tick marks
         yaxt = "n", # suppress tick marks
         cex = 1.25, # size of text points
         mgp = c(0.85, 1, 0.85), # position of axis labels

```

```
cex.lab = 1.6, # magnification of axis label text
xlab = "Methods . . . . . Applications",
ylab = "Mostly Numbers . . . Numbers and Words")
dev.off() # closes the pdf plotting device

pdf(file = "plot_reflected_mds_data_science.pdf",
     width=8.5, height=8.5) # opens pdf plotting device
# use par(mar = c(bottom, left, top, right)) to set up margins on the plot
par(mar=c(7.5, 7.5, 7.5, 5))
First_Dimension <- mds_solution$points[,1] * -1 # reflect horizontal
Second_Dimension <- mds_solution$points[,2] * -1 # reflect vertical
# wordcloud utility for plotting with no overlapping text
textplot(x = First_Dimension,
          y = Second_Dimension,
          words = disciplines,
          show.lines = FALSE,
          xlim = c(-28, 28), # extent of horizontal axis range
          ylim = c(-20, 20), # extent of vertical axis range
          xaxt = "n", # suppress tick marks
          yaxt = "n", # suppress tick marks
          cex = 1.25, # size of text points
          mgp = c(0.85, 1, 0.85), # position of axis labels
          cex.lab = 1.6, # magnification of axis label text
          xlab = "Applications . . . . . Methods",
          ylab = "Words and Numbers . . . Mostly Numbers")
dev.off() # closes the pdf plotting device
```
