

STUMBLING INTO DATA SCIENCE

STUMBLING INTO DATA SCIENCE

A game-based introduction

by Haim Bar, HaiYing Wang, and Jun Yan



**no starch
press**

San Francisco

STUMBLING INTO DATA SCIENCE. Copyright © 2021 by Haim Bar, HaiYing Wang, and Jun Yan

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

Printed in the United States of America

25 24 23 22 21 1 2 3 4 5 6 7 8 9

ISBN-13: 123-4-5678-9012-3 (print)

ISBN-13: 123-4-5678-9012-3 (ebook)

Publisher: Bill Pollock

Production Manager: Rachel Monaghan

Production Editor: E. D. Itor

Developmental Editor: D. E. Itor

Cover Illustrator: Illustrator

Interior Design: Calamari Studios

Technical Reviewer: R. E. Viewer

Copyeditor: C. O. Editor

Proofreader: P. R. Øfreader

Indexer: I. N. Dexer

For information on book distributors or translations, please contact No Starch Press, Inc. directly:

No Starch Press, Inc.

245 8th Street, San Francisco, CA 94103

phone: 415.863.9900; fax: 415.863.9950; info@nostarch.com; www.nostarch.com

Library of Congress Cataloging-in-Publication Data

Uthor, A. U. and Respondent, C. O. R. and Iter, W. R.

Pellentesque habitant morbi: Cum sociis natoque penatibus/

A. U. Thor, C. O. R. Respondent and W. R. Iter

p. cm.

Includes index.

ISBN-10: 1-23456-789-0

ISBN-13: 123-4-56789-012-3

1. Morbi (habitant). 2. Penatibus. I. Title.

XX2303.5.T324 2008

No Starch Press and the No Starch Press logo are registered trademarks of No Starch Press, Inc. Other product and company names mentioned herein may be the trademarks of their respective owners. Rather than use a trademark symbol with every occurrence of a trademarked name, we are using the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The information in this book is distributed on an “As Is” basis, without warranty. While every precaution has been taken in the preparation of this work, neither the author nor No Starch Press, Inc. shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in it.

[S]

Haim: To my family

HaiYing: To my family

Jun: To Jo, Bo, and Jiafeng, without whom the book would have been finished much sooner,
but life would be unimaginable.

About the Authors

Haim Bar is an Associate Professor in the Department of Statistics, University of Connecticut. Before joining UConn, he worked as a software engineer for Motorola, a director of software development in MicroPatent, LLC, a Principal Scientist at ATC-NY, and a statistician at Cornell's Statistical Consulting Unit. His research interests include Bayesian methods, statistical modeling and variable selection in high-dimensional data, especially in applications to genomics.

HaiYing Wang is an Associate Professor in the Department of Statistics, University of Connecticut. He was an Assistant Professor in the Department of Mathematics and Statistics at the University of New Hampshire and an cooling system engineer in the Midea Group. His research interests include informative subdata selection for big data, model selection, model averaging, measurement error models, optimal design, and semi-parametric regression.

Jun Yan is a Professor in the Department of Statistics at the University of Connecticut, a Research Fellow in the Center for Population Health at UConn Health. He received his PhD in Statistics from University of Wisconsin-Madison in 2003. After four years on the faculty of the Department of Statistics and Actuarial Science at the University of Iowa, he joined UConn in 2007. His methodological research interests include survival analysis, clustered data analysis, spatial extremes, and statistical computing. His application domains are public health, environmental sciences, and sports. With a special interest in making his statistical methods available via open source software, he and his coauthors developed and maintain a collection of R packages in the public domain. He is an Elected Member of the International Statistical Institute and a Fellow of the American Statistical Association.

All three authors have been members of the Computer Committee of the Department for years, during which time the idea of the book emerged.

BRIEF CONTENTS

| | |
|--------------------------------|-----|
| Foreword | xv |
| Acknowledgments | xix |
| Foreword by R. E. Viewer | xxi |

PART I: MI ALIQUAM DICTUM

| | |
|------------------------------------|---|
| Chapter 1: Getting to know R | 3 |
|------------------------------------|---|

CONTENTS IN DETAIL

| | |
|-----------------|-----------|
| FOREWORD | xv |
|-----------------|-----------|

| | |
|------------------------|------------|
| ACKNOWLEDGMENTS | xix |
|------------------------|------------|

| | |
|---------------------------------|------------|
| FOREWORD BY R. E. VIEWER | xxi |
|---------------------------------|------------|

PART I MI ALIQUAM DICTUM

| | |
|--------------------------|--|
| 1 | |
| GETTING TO KNOW R | 3 |
| 1.1 | Installing R and Rstudio 3 |
| 1.2 | Basic Operations in R 5 |
| 1.2.1 | Some Useful Functions 5 |
| 1.2.2 | Generating random numbers 10 |
| 1.2.3 | Summary statistics 13 |

FOREWORD

Morbi justo. Aenean nec dolor. In hac habitasse platea dictumst. Proin nonummy porttitor velit. Sed sit amet leo nec metus rhoncus varius. Cras ante. Vestibulum commodo sem tincidunt massa. Nam justo. Aenean luctus, felis et condimentum lacinia, lectus enim pulvinar purus, non porta velit nisl sed eros. Suspendisse consequat. Mauris a dui et tortor mattis pretium. Sed nulla metus, volutpat id, aliquam eget, ullamcorper ut, ipsum. Morbi eu nunc. Praesent pretium. Duis aliquam pulvinar ligula. Ut blandit egestas justo. Quisque posuere metus viverra pede.

Vivamus sodales elementum neque. Vivamus dignissim accumsan neque. Sed at enim. Vestibulum nonummy interdum purus. Mauris ornare velit id nibh pretium ultricies. Fusce tempor pellentesque odio. Vivamus augue purus, laoreet in, scelerisque vel, commodo id, wisi. Duis enim. Nulla interdum, nunc eu semper eleifend, enim dolor pretium elit, ut commodo ligula nisl a est. Vivamus ante. Nulla leo massa, posuere nec, volutpat vitae, rhoncus eu, magna.

Quisque facilisis auctor sapien. Pellentesque gravida hendrerit lectus. Mauris rutrum sodales sapien. Fusce hendrerit sem vel lorem. Integer pellentesque massa vel augue. Integer elit tortor, feugiat quis, sagittis et, ornare non, lacus. Vestibulum posuere pellentesque eros. Quisque venenatis ipsum dictum nulla. Aliquam quis quam non metus eleifend interdum. Nam eget sapien ac mauris malesuada adipiscing. Etiam eleifend neque sed quam. Nulla facilisi. Proin a ligula. Sed id dui eu nibh egestas tincidunt. Suspendisse arcu.

Maecenas dui. Aliquam volutpat auctor lorem. Cras placerat est vitae lectus. Curabitur massa lectus, rutrum euismod, dignissim ut, dapibus a, odio. Ut eros erat, vulputate ut, interdum non, porta eu, erat. Cras ferment-

tum, felis in porta congue, velit leo facilis odio, vitae consecetur lorem quam vitae orci. Sed ultrices, pede eu placerat auctor, ante ligula rutrum tellus, vel posuere nibh lacus nec nibh. Maecenas laoreet dolor at enim. Donec molestie dolor nec metus. Vestibulum libero. Sed quis erat. Sed tristique. Duis pede leo, fermentum quis, consecetur eget, vulputate sit amet, erat.

Donec vitae velit. Suspendisse porta fermentum mauris. Ut vel nunc non mauris pharetra varius. Duis consequat libero quis urna. Maecenas at ante. Vivamus varius, wisi sed egestas tristique, odio wisi luctus nulla, lobortis dictum dolor ligula in lacus. Vivamus aliquam, urna sed interdum porttitor, metus orci interdum odio, sit amet euismod lectus felis et leo. Praesent ac wisi. Nam suscipit vestibulum sem. Praesent eu ipsum vitae pede cursus venenatis. Duis sed odio. Vestibulum eleifend. Nulla ut massa. Proin rutrum mattis sapien. Curabitur dictum gravida ante.

Phasellus placerat vulputate quam. Maecenas at tellus. Pellentesque neque diam, dignissim ac, venenatis vitae, consequat ut, lacus. Nam nibh. Vestibulum fringilla arcu mollis arcu. Sed et turpis. Donec sem tellus, volutpat et, varius eu, commodo sed, lectus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque enim arcu, suscipit nec, tempus at, imperdiet vel, metus. Morbi volutpat purus at erat. Donec dignissim, sem id semper tempus, nibh massa eleifend turpis, sed pellentesque wisi purus sed libero. Nullam lobortis tortor vel risus. Pellentesque consequat nulla eu tellus. Donec velit. Aliquam fermentum, wisi ac rhoncus iaculis, tellus nunc malesuada orci, quis volutpat dui magna id mi. Nunc vel ante. Duis vitae lacus. Cras nec ipsum.

Morbi nunc. Aliquam consecetur varius nulla. Phasellus eros. Cras dapibus porttitor risus. Maecenas ultrices mi sed diam. Praesent gravida velit at elit vehicula porttitor. Phasellus nisl mi, sagittis ac, pulvinar id, gravida sit amet, erat. Vestibulum est. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur id sem elementum leo rutrum hendrerit. Ut at mi. Donec tincidunt faucibus massa. Sed turpis quam, sollicitudin a, hendrerit eget, pretium ut, nisl. Duis hendrerit ligula. Nunc pulvinar congue urna.

Nunc velit. Nullam elit sapien, eleifend eu, commodo nec, semper sit amet, elit. Nulla lectus risus, condimentum ut, laoreet eget, viverra nec, odio. Proin lobortis. Curabitur dictum arcu vel wisi. Cras id nulla venenatis tortor congue ultrices. Pellentesque eget pede. Sed eleifend sagittis elit. Nam sed tellus sit amet lectus ullamcorper tristique. Mauris enim sem, tristique eu, accumsan at, scelerisque vulputate, neque. Quisque lacus. Donec et ipsum sit amet elit nonummy aliquet. Sed viverra nisl at sem. Nam diam. Mauris ut dolor. Curabitur ornare tortor cursus velit.

Morbi tincidunt posuere arcu. Cras venenatis est vitae dolor. Vivamus scelerisque semper mi. Donec ipsum arcu, consequat scelerisque, viverra id, dictum at, metus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut pede sem, tempus ut, porttitor bibendum, molestie eu, elit. Suspendisse potenti. Sed id lectus sit amet purus faucibus vehicula. Praesent sed sem non dui pharetra interdum. Nam viverra ultrices magna.

Aenean laoreet aliquam orci. Nunc interdum elementum urna. Quisque erat. Nullam tempor neque. Maecenas velit nibh, scelerisque a, consequat

ut, viverra in, enim. Duis magna. Donec odio neque, tristique et, tincidunt eu, rhoncus ac, nunc. Mauris malesuada malesuada elit. Etiam lacus mauris, pretium vel, blandit in, ultricies id, libero. Phasellus bibendum erat ut diam. In congue imperdiet lectus.

Aenean scelerisque. Fusce pretium porttitor lorem. In hac habitasse platea dictumst. Nulla sit amet nisl at sapien egestas pretium. Nunc non tellus. Vivamus aliquet. Nam adipiscing euismod dolor. Aliquam erat volutpat. Nulla ut ipsum. Quisque tincidunt auctor augue. Nunc imperdiet ipsum eget elit. Aliquam quam leo, consectetur non, ornare sit amet, tristique quis, felis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque interdum quam sit amet mi. Pellentesque mauris dui, dictum a, adipiscing ac, fermentum sit amet, lorem.

Ut quis wisi. Praesent quis massa. Vivamus egestas risus eget lacus. Nunc tincidunt, risus quis bibendum facilisis, lorem purus rutrum neque, nec porta tortor urna quis orci. Aenean aliquet, libero semper volutpat luctus, pede erat lacinia augue, quis rutrum sem ipsum sit amet pede. Vestibulum aliquet, nibh sed iaculis sagittis, odio dolor blandit augue, eget mollis urna tellus id tellus. Aenean aliquet aliquam nunc. Nulla ultricies justo eget orci. Phasellus tristique fermentum leo. Sed massa metus, sagittis ut, semper ut, pharetra vel, erat. Aliquam quam turpis, egestas vel, elementum in, egestas sit amet, lorem. Duis convallis, wisi sit amet mollis molestie, libero mauris porta dui, vitae aliquam arcu turpis ac sem. Aliquam aliquet dapibus metus.

Vivamus commodo eros eleifend dui. Vestibulum in leo eu erat tristique mattis. Cras at elit. Cras pellentesque. Nullam id lacus sit amet libero aliquet hendrerit. Proin placerat, mi non elementum laoreet, eros elit tincidunt magna, a rhoncus sem arcu id odio. Nulla eget leo a leo egestas facilisis. Curabitur quis velit. Phasellus aliquam, tortor nec ornare rhoncus, purus urna posuere velit, et commodo risus tellus quis tellus. Vivamus leo turpis, tempus sit amet, tristique vitae, laoreet quis, odio. Proin scelerisque bibendum ipsum. Etiam nisl. Praesent vel dolor. Pellentesque vel magna. Curabitur urna. Vivamus congue urna in velit. Etiam ullamcorper elementum dui. Praesent non urna. Sed placerat quam non mi. Pellentesque diam magna, ultricies eget, ultrices placerat, adipiscing rutrum, sem.

R. E. Viewer
New York
December 2007

ACKNOWLEDGMENTS

Insert acknowledgments here

FOREWORD

Morbi justo. Aenean nec dolor. In hac habitasse platea dictumst. Proin nonummy porttitor velit. Sed sit amet leo nec metus rhoncus varius. Cras ante. Vestibulum commodo sem tincidunt massa. Nam justo. Aenean luctus, felis et condimentum lacinia, lectus enim pulvinar purus, non porta velit nisl sed eros. Suspendisse consequat. Mauris a dui et tortor mattis pretium. Sed nulla metus, volutpat id, aliquam eget, ullamcorper ut, ipsum. Morbi eu nunc. Praesent pretium. Duis aliquam pulvinar ligula. Ut blandit egestas justo. Quisque posuere metus viverra pede.

Vivamus sodales elementum neque. Vivamus dignissim accumsan neque. Sed at enim. Vestibulum nonummy interdum purus. Mauris ornare velit id nibh pretium ultricies. Fusce tempor pellentesque odio. Vivamus augue purus, laoreet in, scelerisque vel, commodo id, wisi. Duis enim. Nulla interdum, nunc eu semper eleifend, enim dolor pretium elit, ut commodo ligula nisl a est. Vivamus ante. Nulla leo massa, posuere nec, volutpat vitae, rhoncus eu, magna.

Quisque facilisis auctor sapien. Pellentesque gravida hendrerit lectus. Mauris rutrum sodales sapien. Fusce hendrerit sem vel lorem. Integer pellentesque massa vel augue. Integer elit tortor, feugiat quis, sagittis et, ornare non, lacus. Vestibulum posuere pellentesque eros. Quisque venenatis ipsum dictum nulla. Aliquam quis quam non metus eleifend interdum. Nam eget sapien ac mauris malesuada adipiscing. Etiam eleifend neque sed quam. Nulla facilisi. Proin a ligula. Sed id dui eu nibh egestas tincidunt. Suspendisse arcu.

Maecenas dui. Aliquam volutpat auctor lorem. Cras placerat est vitae lectus. Curabitur massa lectus, rutrum euismod, dignissim ut, dapibus a, odio. Ut eros erat, vulputate ut, interdum non, porta eu, erat. Cras fermentum, felis in porta congue, velit leo facilisis odio, vitae consectetur lorem quam vitae orci. Sed ultrices, pede eu placerat auctor, ante ligula rutrum tellus, vel posuere nibh lacus nec nibh. Maecenas laoreet dolor at enim. Donec molestie dolor nec metus. Vestibulum libero. Sed quis erat. Sed tristique. Duis pede leo, fermentum quis, consectetur eget, vulputate sit amet, erat.

Donec vitae velit. Suspendisse porta fermentum mauris. Ut vel nunc non mauris pharetra varius. Duis consequat libero quis urna. Maecenas at ante. Vivamus varius, wisi sed egestas tristique, odio wisi luctus nulla, lobortis dictum dolor ligula in lacus. Vivamus aliquam, urna sed interdum porttitor, metus orci interdum odio, sit amet euismod lectus felis et leo. Praesent ac wisi. Nam suscipit vestibulum sem. Praesent eu ipsum vitae pede cursus venenatis. Duis sed odio. Vestibulum eleifend. Nulla ut massa. Proin rutrum mattis sapien. Curabitur dictum gravida ante.

Phasellus placerat vulputate quam. Maecenas at tellus. Pellentesque neque diam, dignissim ac, venenatis vitae, consequat ut, lacus. Nam nibh. Vestibulum fringilla arcu mollis arcu. Sed et turpis. Donec sem tellus, volutpat et, varius eu, commodo sed, lectus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque enim arcu, suscipit nec, tempus at, imperdiet vel, metus. Morbi volutpat purus at erat. Donec dignissim, sem id semper tempus, nibh massa eleifend turpis, sed pellentesque wisi purus sed libero. Nullam lobortis tortor vel risus. Pellentesque consequat nulla eu tellus. Donec velit. Aliquam fermentum, wisi ac rhoncus iaculis, tellus nunc malesuada orci, quis volutpat dui magna id mi. Nunc vel ante. Duis vitae lacus. Cras nec ipsum.

Morbi nunc. Aliquam consectetur varius nulla. Phasellus eros. Cras dapibus porttitor risus. Maecenas ultrices mi sed diam. Praesent gravida velit at elit vehicula porttitor. Phasellus nisl mi, sagittis ac, pulvinar id, gravida sit amet, erat. Vestibulum est. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur id sem elementum leo rutrum hendrerit. Ut at mi. Donec tincidunt faucibus massa. Sed turpis quam, sollicitudin a, hendrerit eget, pretium ut, nisl. Duis hendrerit ligula. Nunc pulvinar congue urna.

Nunc velit. Nullam elit sapien, eleifend eu, commodo nec, semper sit amet, elit. Nulla lectus risus, condimentum ut, laoreet eget, viverra nec, odio. Proin lobortis. Curabitur dictum arcu vel wisi. Cras id nulla venenatis tortor congue ultrices. Pellentesque eget pede. Sed eleifend sagittis elit. Nam sed tellus sit amet lectus ullamcorper tristique. Mauris enim sem, tri-

stique eu, accumsan at, scelerisque vulputate, neque. Quisque lacus. Donec et ipsum sit amet elit nonummy aliquet. Sed viverra nisl at sem. Nam diam. Mauris ut dolor. Curabitur ornare tortor cursus velit.

Morbi tincidunt posuere arcu. Cras venenatis est vitae dolor. Vivamus scelerisque semper mi. Donec ipsum arcu, consequat scelerisque, viverra id, dictum at, metus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut pede sem, tempus ut, porttitor bibendum, molestie eu, elit. Suspendisse potenti. Sed id lectus sit amet purus faucibus vehicula. Praesent sed sem non dui pharetra interdum. Nam viverra ultrices magna.

Aenean laoreet aliquam orci. Nunc interdum elementum urna. Quisque erat. Nullam tempor neque. Maecenas velit nibh, scelerisque a, consequat ut, viverra in, enim. Duis magna. Donec odio neque, tristique et, tincidunt eu, rhoncus ac, nunc. Mauris malesuada malesuada elit. Etiam lacus mauris, pretium vel, blandit in, ultricies id, libero. Phasellus bibendum erat ut diam. In congue imperdiet lectus.

Aenean scelerisque. Fusce pretium porttitor lorem. In hac habitasse platea dictumst. Nulla sit amet nisl at sapien egestas pretium. Nunc non tellus. Vivamus aliquet. Nam adipiscing euismod dolor. Aliquam erat volutpat. Nulla ut ipsum. Quisque tincidunt auctor augue. Nunc imperdiet ipsum eget elit. Aliquam quam leo, consectetur non, ornare sit amet, tristique quis, felis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque interdum quam sit amet mi. Pellentesque mauris dui, dictum a, adipiscing ac, fermentum sit amet, lorem.

Ut quis wisi. Praesent quis massa. Vivamus egestas risus eget lacus. Nunc tincidunt, risus quis bibendum facilisis, lorem purus rutrum neque, nec porta tortor urna quis orci. Aenean aliquet, libero semper volutpat luctus, pede erat lacinia augue, quis rutrum sem ipsum sit amet pede. Vestibulum aliquet, nibh sed iaculis sagittis, odio dolor blandit augue, eget mollis urna tellus id tellus. Aenean aliquet aliquam nunc. Nulla ultricies justo eget orci. Phasellus tristique fermentum leo. Sed massa metus, sagittis ut, semper ut, pharetra vel, erat. Aliquam quam turpis, egestas vel, elementum in, egestas sit amet, lorem. Duis convallis, wisi sit amet mollis molestie, libero mauris porta dui, vitae aliquam arcu turpis ac sem. Aliquam aliquet dapibus metus.

Vivamus commodo eros eleifend dui. Vestibulum in leo eu erat tristique mattis. Cras at elit. Cras pellentesque. Nullam id lacus sit amet libero aliquet hendrerit. Proin placerat, mi non elementum laoreet, eros elit tincidunt magna, a rhoncus sem arcu id odio. Nulla eget leo a leo egestas facilisis. Curabitur quis velit. Phasellus aliquam, tortor nec ornare rhoncus, purus urna posuere velit, et commodo risus tellus quis tellus. Vivamus leo turpis, tempus sit amet, tristique vitae, laoreet quis, odio. Proin scelerisque bibendum ipsum. Etiam nisl. Praesent vel dolor. Pellentesque vel magna. Curabitur urna. Vivamus congue urna in velit. Etiam ullamcorper elementum dui. Praesent non urna. Sed placerat quam non mi. Pellentesque diam magna, ultricies eget, ultrices placerat, adipiscing rutrum, sem.

R. E. Viewer
New York
December 2007

PART I

MI ALIQUAM DICTUM

Vivamus adipiscing. Curabitur imperdiet tempus turpis. Vivamus sapien dolor, congue venenatis, euismod eget, porta rhoncus, magna. Proin condimentum pretium enim. Fusce fringilla, libero et venenatis facilisis, eros enim cursus arcu, vitae facilisis odio augue vitae orci. Aliquam varius nibh ut odio. Sed condimentum condimentum nunc. Pellentesque eget massa. Pellentesque quis mauris. Donec ut ligula ac pede pulvinar lobortis. Pellentesque euismod. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent elit. Ut laoreet ornare est. Phasellus gravida vulputate nulla. Donec sit amet arcu ut sem tempor malesuada. Praesent hendrerit augue in urna. Proin enim ante, ornare vel, consequat ut, blandit in, justo. Donec felis elit, dignissim sed, sagittis ut, ullamcorper a, nulla. Aenean pharetra vulputate odio.

Quisque enim. Proin velit neque, tristique eu, eleifend eget, vestibulum nec, lacus. Vivamus odio. Duis odio urna, vehicula in, elementum aliquam, aliquet laoreet, tellus. Sed velit. Sed vel mi ac elit aliquet interdum. Etiam sapien neque, convallis et, aliquet vel, auctor non, arcu. Aliquam suscipit aliquam lectus. Proin tincidunt magna sed wisi. Integer blandit lacus ut lorem. Sed luctus justo sed enim.

1

GETTING TO KNOW R




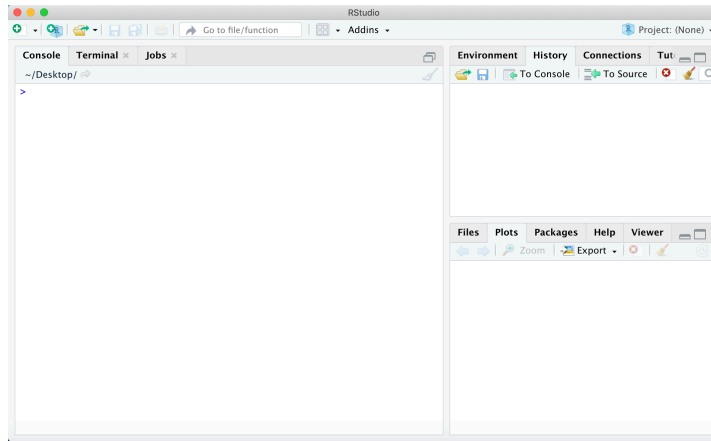
Just like a chef needs a set of tools, a kitchen with a convenient surface to work on, and a detailed cookbook with different recipes, a data scientist needs a powerful programming language, a convenient development environment, and good documentation. In this chapter we provide a brief introduction to the R language and the RStudio integrated development environment (IDE). Here, and in the remainder of the book we provide some essential ‘recipes’ for data scientists.

1.1 Installing R and Rstudio

To install R go to the website of the Comprehensive R Archive Network (<https://cran.r-project.org/>) and download the latest version of R for your operating system (Windows, MacOS, or Linux). Follow the installation instructions.

Next, go the Rstudio download website (<https://rstudio.com/products/rstudio/download/>) and get the Desktop version (open source license). Follow

the installation instruction. An icon that looks like this  will be on your computer's desktop. Double-click on this icon to start the R session. The Rstudio IDE will open, and will look like this:



The left-hand side of the screen contains the Console tab. Notice the `>` sign (called the 'prompt'). When you see this character, it means that R is ready for the next command. Put the cursor there, and then type

```
2 + 2
```

and hit Enter on the keyboard. You should get the following on the Console:

```
[1] 4
```

Notice the top-right part of the Rstudio window. You should see an 'Environment' and a 'History' tab. Click on History. All your previous input appears there. Try entering another calculation or command in the console and see that they appear in the session's history. For example, try entering the following

```
date()
```

Click on the Environment tab. It should be empty when you start R for the first time. In the Console, type

```
myFirstVariable <- factorial(5)
```

Notice that nothing was printed in the Console, but the Environment tab now contains a table with one row, with 'myFirstVariable' appearing in the cell on the left, and its value (120) on the right. Any object appearing in the Environment tab is available to you throughout your R session, and you don't have to redefine or recalculate it. For example, you try typing the following in the console:

```
myFirstVariable/6
```

The Console should now display 20.

The lower-right side of the IDE has a few tabs: Files provides a file browser, Packages provides information about installed packages (more about it later), Plots will contain any plot generated during the R session, and the Help tab is used to obtain information about built-in functions.

Finally, before we move on to the next section, in the Rstudio top menu, click on File, then on New File, and then on R Script. Alternatively, you can click on little green ‘+’ icon in the top-left part of the IDE. This will split the left side of the Rstudio IDE into two parts – the lower part will contain the Console, and the top part will contain a tab labeled ‘Untitled1’. This is where you can enter R code which you will save to a permanent file, and re-use later. For example, enter the following in the blank space in the Untitled1 tab:

```
# This is my first R program
cat("Hello, World!\n")
```

Then, from the main menu in Rstudio, click on File, then on Save, and in the ‘Save As’ box enter FirstProgram.R and click the Save button. Notice that the tab name is now FirstProgram.R.

In that part of the window, there should now be a small button called Source. Click on it. The program will be executed and the output will be shown in the Console. You can also execute individual lines in the source code. Just put the cursor anywhere in that line, and click on the Run button (which is near the Source button) or click Ctrl+Enter (or Command+Enter on a Mac).

That’s it. You are now ready to start learning programming in R.

1.2 Basic Operations in R

1.2.1 Some Useful Functions

R has many built-in functions, and many more in external packages. We will introduce them as we go, but let’s get started with some basic ones. The documentation on each function can be obtained by using `?func` or `help(func)` where `func` is some function. For example, to get the documentation about using the `date()` function, try the following:

```
1 help(date)
2 ?date
```

When we start an R session, it’s important to determine the ‘working directory’. This is the folder on our computer which R will use to search for data or code files and where it will save results (unless we explicitly specify another directory). To find out which folder is currently used, we use the following:

```
1 getwd()
```

Try it! We sometimes refer to the complete directory information as ‘path’, since it tells us how to get to a specific file, starting from the root directory (which is denoted by ‘/’ on Mac and Linux, or ‘\’ on Windows). The path corresponds to the hierarchical organization of folders (directories) on your computer.

If we want (and we often will), we may change the working directory by using `setwd()`. For example, try:

```
1 setwd("~/Desktop")
```

The `~/` notation is a shortcut to your home directory. Using this shortcut is convenient because if you are using different computers or you share code with others, the home directory may be different on each computer.

Data is stored in ‘*variables*’. We can get the data from a file (Excel, comma-separated values, etc.), the Internet, or we can generate it ourselves. Let’s start by generating some data. The simplest function to create data is called `c()`, which stands for ‘combine’.

```
1 courseNames <- c("Data Science", "Statistics", "Probability")
```

We created a variable called `courseNames`, and it contains three values. We assigned the values into the variable by using the `<-` operator. We can now do things with this variable. For example, we check the variable type, using the `class()` function:

```
1 class(courseNames)
```

In this case we see that this variable is used to store text (‘character’ class):

```
[1] "character"
```

We can also get a bit more detailed information by checking its structure, using the `str()` function:

```
1 str(courseNames)
```

which shows that the variable is of class ‘character’, and it is a vector with three values:

```
chr [1:3] "Data Science" "Statistics" "Probability"
```

We can also get the number of elements in a vector by using the `length()` function,

```
1 length(courseNames)
```

Pick good variable names. They should describe the meaning of the value, yet be short enough to type. Variable names can only contain letters, numbers, the dot character, or the underline character. Variable names can only begin with either a letter, or a dot as long as it is not followed by a number. They should not be the same as an existing function name or other reserved words in the language (like ‘while’, ‘if’, ‘quit’.)

We often have to generate variables which contain sequences of numbers. To do that, we use the `seq()` function. For example, we can create a variable which contains all the odd numbers between 1 and 20:

```
1 oddLT20 <- seq(1,20, by=2)
```

The variable `oddLT20` contains the following values: 1 3 5 7 9 11 13 15 17 19. To generate consecutive values, we can also use the colon operator. For example, try:

```
1 (firstThirteen <- 1:13)
```

Note that the parentheses surrounding the whole line cause R to print the content of the variable `firstThirteen` in the console.

Another useful function to generate data is `rep()`, which replicates values. We often have to generate a vector consisting of a single value (e.g., a vector of ones or zeros), and we can do it like in the following example. We use this opportunity to also introduce the sum and cumulative sum functions, `sum()` and `cumsum()`.

```
1 ones <- rep(1, 10)
2 sum(ones)
3 cumsum(ones)
```

The variable `ones` contains the values: 1 1 1 1 1 1 1 1 1 1, and the sum is 10. Their cumulative sum of ones is a vector of length 10, such that the first value is just the first element of `ones`, the second value is the sum of the first two elements in `ones`, and so forth: 1 2 3 4 5 6 7 8 9 10. Try the following: `cumsum(cumsum(ones))` (but first try to think what the result will be).

There are also many functions to handle text. The `paste()` function attaches two strings of characters together. The default is to concatenate strings with a space as a separator, but we can use the `collapse` option (also called an ‘argument’ of the function) to use a different separator. For example,

```

1 paste(courseNames)
2 paste(courseNames, collapse=", ")

```

The second line generates the following: Data Science, Statistics, Probability.

Let's combine what we've learned so far to create the 52 cards in a standard deck, which includes four suits: Club, Diamond, Heart, and Spade:

```

1 suits <- c(rep("C",13), rep("D",13), rep("H",13), rep("S",13))
2 cards <- paste0(suits, rep(1:13,4))

```

Run each line separately, and check the content of 'suits' and what `rep(1:13,4)` generates. Note that the function `paste0()` is the same as `paste(..., collapse="")`. The content of the cards variable is:

```

C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 C13 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 H1
↪ H2 H3 H4 H5 H6 H7 H8 H9 H10 H11 H12 H13 S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13

```

Let's 'deal' five cards to each player for a game of poker. We will use the `sample()` function.

```

1 set.seed(5252)
2 (pokerHand <- matrix(sample(cards,20,replace=FALSE), nrow=5, ncol=4))

```

The `sample()` function in this example is used to draw 20 cards at random from the deck, without replacement. Then, to divide it into four hands, we use the `matrix()` function, and specify the number of rows and the number of columns. The `sample()` function draws a different random sample each time unless we set the random 'seed' to a fixed value. The `set.seed()` function is used to ensure the reproducibility of the results. Try running the code without the `set.seed()` line, and see that you get different results each time.

```

      [,1] [,2] [,3] [,4]
[1,] "S8"  "S13" "S7"  "C4"
[2,] "D8"  "C1"  "H6"  "H11"
[3,] "D4"  "C11" "S6"  "S11"
[4,] "D12" "C13" "S4"  "S1"
[5,] "C6"  "S9"  "H12" "H9"

```

Check the class and the structure of the variable `pokerHand`, using the functions we've mentioned earlier.

We can save variables to a file in order to use them in a later session. For example:

```
1 save(pokerHand, file="tmp/pokerHand.RData")
```

If we don't specify the path, the file will be stored in the current working directory. Recall that you can find out which directory is used with the `getwd()` function, and set it to another directory with `setwd()`. Then, we may get the saved variables by using the following:

```
1 load("tmp/pokerHand.RData")
```

Variables which we do not save will not be available once we terminate the current R session. When we quit an R session we have an option to save the entire session's information. However, if there are variables, datasets, or functions which we have created and want to save, it's better to save them explicitly. After you use the `load()` function in the example above, the variable `pokerHand` will be available to use, even if you quit the R session in which the variable was created.

There are a few constants in R, including the letters of the alphabet (upper- and lower-case), the month names, and the number π . Try running the following:

```
1 cat(LETTERS)
2 cat(letters)
3 cat(month.abb)
4 cat(month.name)
5 cat(pi)
```

The output for the third line is:

```
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
```

The base distribution of R is very comprehensive, but there are thousands of contributed packages which are written by R users. We will use several such packages in the book, so let us demonstrate the basic usage. The package *lattice* provides 'elegant high-level data visualization system with an emphasis on multivariate data'. To install the package, we use the `install.packages()` function. In order to use the package, you have to load it using the `library()` function:

```
1 install.packages("lattice")
2 library("lattice")
```

Installing the package has to be done just once. Occasionally, you may be prompted to install updates, which can also be done by using the `update.packages()`

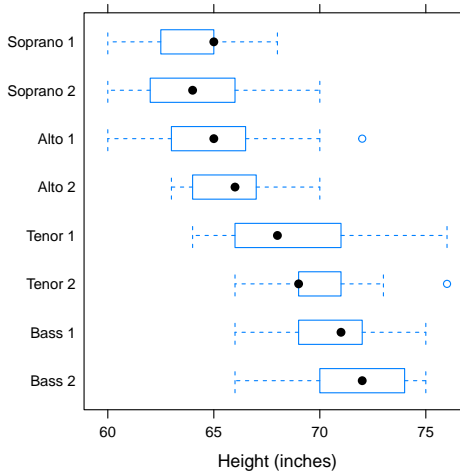


Figure 1-1: Using the lattice package to create boxplots.

function. Loading a package has to be done each time you begin a new R session.

In the following example we use a built-in dataset of opera singers, and we plot their heights by their vocal parts (Figure 1-1).

```
1 library("lattice")
2 bwplot(voice.part ~ height, data=singer, xlab="Height (inches)")
```

We will learn a lot more about data visualization in subsequent chapters. When you want to finish your session, just type `quit()`.

1.2.2 Generating random numbers

In this book much of the learning will be done by using simulated data, so let's start by learning how to generate data. The basic function to generate random data is `runif()` which is used to draw random numbers uniformly between 0 and 1. In the following example we create 10,000 random draws from a uniform distribution, we keep these numbers in a variable called `simData`, and then plot a histogram to show the distribution of the data we have generated. The `set.seed()` function is used to ensure that every time we run this code, we will get the same set of random numbers. This is called *reproducible code*.

```
1 # Generate 10,000 points from a uniform distribution
2 set.seed(210313)
3 n <- 10000
```

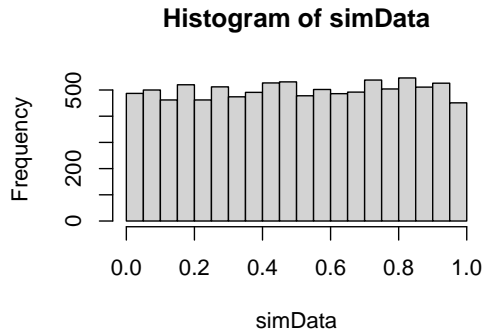


Figure 1-2: A histogram of a random sample from a standard uniform distribution.

```
4  simData <- runif(n)
5  hist(simData)
```

From the range and the flatness of the histogram in Figure 1-2 we can see that the generated data is indeed uniform on $[0,1]$. The `runif()` function can be used to draw random numbers uniformly on any finite interval. For example, if we want our random numbers to be in the interval $[1,5]$ we will run the following code:

```
1  n <- 10000
2  simData <- runif(n, min=1, max=5)
```

Try it, and draw the histogram as in the previous example. Try it with a fixed seed and verify that you get the same output each time. Then, run the code without a fixed seed and observe that you get a different distribution each time. Since the number of random draws is fairly large, the shape of the histogram will not change much.

From a random draw of a uniform distribution we can generate random numbers from other distributions. For example, we want to simulate (fair) coin flips, and count the number of Heads that we get. Let's say we want to simulate 200 coin tosses. We will draw 200 random numbers from a uniform distribution, and decide that we got Heads in the i -th toss if the i -th random number is less than 0.5, and Tails otherwise. Try the following code multiple times (do not use `set.seed()`). What do you observe? We will discuss it further in a different chapter.

```
1  set.seed(220513)
2  n.trials <- 200
3  cat("Number of Heads is: ", sum(runif(n.trials) < 0.5), "\n")
```

A few comments:

- A comment in R starts with #. Any text following the # sign is considered user-documentation and is not executed by R.
- We have used the `cat()` function to print (concatenate) text to the console. Fixed text appears in double (or single) quotes, but the content of variables or output from R functions should not be quoted. The `\n` symbol tells R to print a newline character at the end. Try to see what happens if you remove it.
- The expression within the `sum()` function produces TRUE/FALSE (Boolean) values. First, we draw `n.trials` random numbers from a uniform distribution. Then, each one is compared with 0.5. If the value is less than 0.5, the returned value is TRUE. Otherwise, it's FALSE. This demonstrates one of R's greatest features - allowing to run 'vectorized' code. In one line, we generated 200 random numbers and for each one we checked if it is less than 0.5, and as a result, we got 200 Boolean values which we have added together (TRUE counts as 1, and FALSE counts as 0.) So, the result in the sum is the number of Heads in 200 tosses.

The number of Heads we get (with this seed) is 92. Statistical inference is based on a mental exercise in which we ask, if we could repeat the same experiment infinitely many times, what would we see? With simulations, we can get a good approximation. For example, the 200 coin-tosses experiment can be repeated, say, 100 times. One way is to use loops, like in the following example:

```
1 set.seed(442886)
2 n.trials <- 200 # the number of coin-tosses in each experiment
3 reps <- 100 # the number of experiments
4 Heads <- rep(0, reps) # A vector to store the results (initialize with 0s)
5 for (i in 1:reps) {
6   Heads[i] <- sum((runif(n.trials) < 0.5))
7 }
8 hist(Heads, breaks=20)
```

We've used a `for()` loop, with an index variable called `i` which runs from 1 to 100. In each iteration we simulated `n.trials=200` coin tosses, as before. The result from each iteration is stored in the *i*-th position in the vector we called `Heads`. Figure 1-3 shows the distribution of 100 simulations, each of which consists of 200 coin tosses and from each of the 100 simulations we obtain the proportion of times we got Heads.

Using the uniform distribution, we have created a random draw from a different distribution, called the 'binomial'. The mathematical notation is $Bin(N, p)$ where N is the number of trials such that in each trial there can be exactly two outcomes (e.g., Heads or Tails in coin tosses), and p is the probability of the first possible outcome (e.g., Head), and $1 - p$ is the proba-

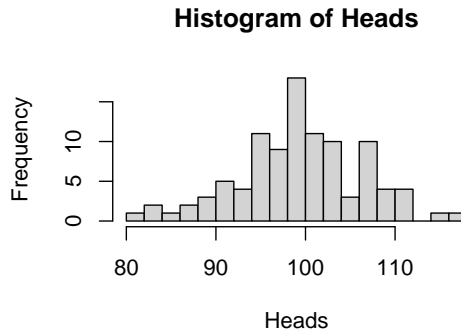


Figure 1-3: A histogram of 100 experiments in which we flip a coin 200 time and calculate the proportion of times we get Heads out of 200.

bility of the second possible outcome (e.g., Tail). R has a built-in function to generate random numbers from many different distributions, including the binomial so the code above can be replaced by the following, which uses the `rbinom()` function:

```
1 set.seed(442886)
2 reps <- 100
3 n.trials <- 200
4 Heads <- rbinom(reps, n.trials, 0.5)
5 hist(Heads, breaks=20)
```

Exercises:

1. Change the number of simulated coin-toss simulations (`reps`) to 1,000 and rerun the code. Then change it to 10,000 and run it again. What do you notice?
2. Change the probability of Heads to 0.2 and run the code again. Then, change it to 0.8. What do you observe?
3. Try other values of the number of trials and the probability of Heads.

1.2.3 Summary statistics

We will introduce some statistical functions to summarize data. To demonstrate some of these functions, let's first generate some data:

```
1 set.seed(321333)
2 n <- 10000
3 lambda <- 10
4 x <- -log(runif(n))/lambda
```

Note that in R the `log` function uses the natural logarithm by default. To use base 10 or base 2, use `log10` and `log2`, respectively. To specify another base, use the `base` argument. For example, try `log(16, base=4)`.

The most commonly used statistical summary is the mean (a.k.a. the average), \bar{x} :

$$\bar{x} = \frac{x_1 + \dots + x_n}{n}.$$

Other ways to estimate some sort of ‘central tendency’ of a distribution are:

- The trimmed mean, which is similar to the mean, except that the smallest and largest $p \cdot 100\%$ of the values are excluded from the computation. In our example, if we take $p = 0.1$ with the simulated data, only 8,000 data points will be used in the calculation of the trimmed mean;
- The median, which is a number (denoted here by $x_{0.5}$) such that half the data points are greater than $x_{0.5}$ and half are less than or equal to $x_{0.5}$.

Let’s compute the sample’s mean, trimmed mean, and median. Notice that in this example the mean is greater than trimmed mean, which is greater than the median. In general, the mean is not a great estimate of the ‘center’ of a non-symmetric distribution. We say that the mean is more ‘sensitive to extreme values’ than the median.

```
1 mean(x)
2 mean(x, trim=0.1)
3 median(x)
```

In this example with $n=10000$ random numbers we get that the mean is 2.5, the trimmed mean is 2.51, and the median is 2.5.

The formula we used, `x <- -log(runif(n))/lambda`, generates random number from the so-called exponential distribution with rate parameter $\lambda = 10$. We denote it by $X \sim \exp(\lambda)$. The exponential distribution is often used to model random waiting times, like the time between incoming text messages. We would usually generate it by using the `rexp()` function in R, like this: `x <- rexp(n, lambda)`. Mathematical analysis of the distribution leads to the fact that the expected value of an exponential random variable with rate λ , is $1/\lambda$. We see that the theoretical expected value of our example is 0.1, and the sample mean is very close to 0.1. This is no coincidence - we will discuss it in later chapter.

The distribution of `x` is shown in Figure 1-4 as a *box-and-whisker plot* (or simply, boxplot). This is a very simple representation of numeric data, which is constructed by summarizing the data using a few numeric characteristics. In Figure 1-4 the boxplot is drawn horizontally, and the vertical grey line inside the box is the median. Similar to the median, we find the first quartile - a point, $x_{0.25}$, such that 25% of the values are less than $x_{0.25}$ and 75% are greater than $x_{0.25}$; and the third quartile - a point, $x_{0.75}$, such that 75% of the values are less than $x_{0.75}$ and 25% are greater than $x_{0.75}$. The first and third quartiles are the vertical edges of the box, also called the lower

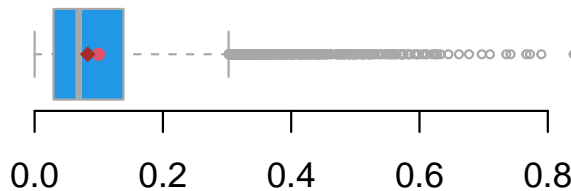


Figure 1-4: A boxplot of data generated from an exponential distribution with $\lambda = 10$.

and upper hinges. So, the box represents the middle 50% of the data. The range between the first and third quartiles is called the *Inter-Quartile Range*, or *IQR*, which is sometimes used to estimate the dispersion or spread of the data. The ‘whiskers’, which are the dashed grey lines, are constructed by adding $1.5 \cdot \text{IQR}$ to each side of the box. If the result is smaller than the minimum value (or greater than the maximum), then the whisker only extends to the minimum (maximum). Points within the range between the two whiskers are not plotted individually, since their distribution is summarized succinctly by the box-and-whiskers plot. Points outside the range between the two whiskers are considered ‘outliers’, or extreme values, and are shown explicitly.

The plot was generated with the following code. Try it, and try changing some of the parameters to understand their role. We will cover the topic of visualization in a different chapter.

```

1 boxplot(x, cex=0.5, col=4, border = "grey66", horizontal = T, axes=F,
  ↪ at=0.25)
2 axis(1, pos = 0)
3 points(mean(x), 0.25, col=2, pch=19, cex=0.7)
4 points(mean(x, trim=0.1), 0.25, col="brown", pch=18)

```

A boxplot does not include the mean, or the trimmed mean, but we have added them here as a red circle and brown diamond, respectively, in order to show that they are different than the median. The median is smaller

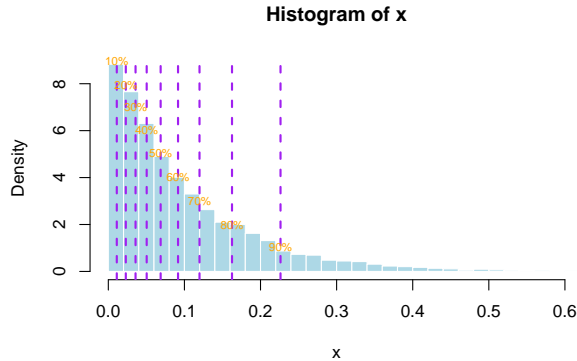


Figure 1-5: A histogram of a sample from an exponential distribution, with the 10, 20, 30,... percentiles.

than the mean in this case, so we say that the distribution is skewed to the left. The median does not depend on the scale of the data. It simply represents where half the data lies.

A more detailed summary of a sample can be obtained by calculating its quantiles. In the boxplot, only three quantiles (also called percentiles) are shown. We can show the quintiles (20, 40, 60, 80 percentiles), or deciles (10, 20,..., 90 percentiles) by using the `quantile()` function.

```

1 hist(x, breaks=50, border="white", col="lightblue", freq=FALSE,
   ↪ xlim=c(0,0.6))
2 deciles <- quantile(x, probs=seq(0.1,0.9,by=0.1))
3 abline(v=deciles, lty=2, col="purple", lwd=2)
4 text(deciles, dexp(deciles, 10), paste0(seq(10,90,by=10), "%"), cex=0.7,
   ↪ col="orange")

```

In addition to sample statistics which summarize some notion of the center of the distribution, we are often interested in estimating the dispersion of the data. The most commonly used measures of dispersion are the variance, and its square root - the standard deviation. The variance is defined as follows

$$Var(X) = \frac{(x_1 - \mu)^2 + \dots + (x_n - \mu)^2}{n}$$

where μ is the mean of the distribution of X . In words, the variance is the average of the squared deviations from the mean. Notice that in R the variance is computed with $n-1$ in the denominator (with $n-1$ we get an ‘unbiased estimator’ for the true variance).

The corresponding functions in R are `var()` and `sd()`. In the following code we also demonstrate the `IQR()` function.

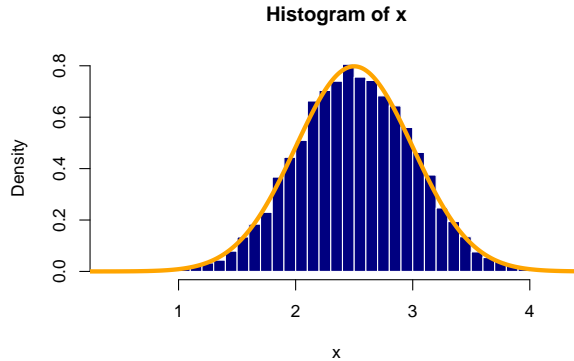


Figure 1-6: A histogram of a sample from a normal distribution, with mean=2.5 and variance=0.25.

```
1 var(x)
2 sd(x) # note that sd(x) is sqrt(sd(x))
3 IQR(x)
```

We will see in subsequent chapters that the normal distribution plays a major role in statistics. It is defined by a probability density function, with two parameters – μ and σ^2 :

$$\phi(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{(x - \mu)^2}{2\sigma^2} \right].$$

It is symmetric, ‘bell-shaped’, and centered around the mean, μ , as shown in Figure 1-6. The blue histogram is generated by drawing random numbers from a normal distribution, using the `rnorm()` function. Here, we use $\mu = 2.5$ and $\sigma^2 = 0.25$. The orange curve shows the function ϕ , by using the `dnorm()` function (to obtain the density function of x).

```
1 x <- rnorm(10000, mean=2.5, sd=0.5)
2 hist(x, breaks=30, border="white", col="navyblue", freq=FALSE)
3 xs <- seq(0, 5, length=500)
4 lines(xs, dnorm(xs, 2.5, 0.5), col="orange", lwd=4)
```

We can obtain more detailed summaries about the sample, by using the `summary()` function. `summary()` gives the minimum, maximum, the 25th, 50th, and 75th percentiles, as well as the mean of the sample. Using the random numbers from the normal distribution in the previous code:

```
1 print(summary(x))
```

| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|------|---------|--------|------|---------|------|
| 0.42 | 2.17 | 2.50 | 2.50 | 2.85 | 4.28 |

Another possibility (among many) to summarize data in R is the `describe()` function from the `psych` package,

```
1 print(psych::describe(x))
```

It gives the sample size, the standard deviation, the trimmed mean, the mean absolute deviation (`mad`), the range (maximum minus minimum), the skewness, and the kurtosis **will we use the kurtosis later? If so, we should add a ref.**

| | vars | n | mean | sd | median | trimmed | mad | min | max | range | skew | kurtosis | se |
|----|------|-------|------|-----|--------|---------|------|------|------|-------|-------|----------|----|
| X1 | 1 | 10000 | 2.5 | 0.5 | 2.5 | 2.51 | 0.51 | 0.42 | 4.28 | 3.86 | -0.03 | -0.05 | 0 |

We conclude this chapter with some functions which can be used to summarize discrete (categorical) data, where the mean, variance, quantiles, etc. are not applicable. We will revisit the topic of summarizing data in the chapter on **visualization**.

To simulate categorical data we can use the `rmultinom()` function, which simulates putting N objects in K bins with given probabilities for each bin. Another possibility is to use `cut()` function which divides a range of numbers into discrete ranges and creates discrete categories in a factor variable. For example, suppose there is a town with three hotels (Motel 6, Best Western, and Hilton) with 400, 300, and 300 rooms, and one auto rental company with only two makes of cars (700 Honda, and 300 Teslas). In the following code we simulate the allocation of 100 visitors to hotels and cars. We use the `table()` function to show the counts by hotel, by car, and by both.

```
1 hotelrooms <- cut(runif(100),breaks = c(0,0.4,0.7,1), include.lowest =
  ↳ TRUE)
2 levels(hotelrooms) <- c("Motel 6", "Best Western", "Hilton")
3 autorental <- cut(runif(100),breaks = c(0,0.7,1), include.lowest = TRUE)
4 levels(autorental) <- c("Honda", "Tesla")
5 (hoteltbl <- table(hotelrooms))
6 (autotbl <- table(autorental))
7 table(hotelrooms, autorental)
```

| hotelrooms | | |
|------------|--------------|--------|
| Motel 6 | Best Western | Hilton |
| 50 | 25 | 25 |

```

autorental
Honda Tesla
66 34

```

```

             autorental
hotelrooms   Honda Tesla
Motel 6      31 19
Best Western 17 8
Hilton       18 7

```

We can use the `max()` and `which.max()` functions to find the mode of the data (the most frequent value).

```

1 cat(levels(hotelrooms)[which.max(hoteltbl)], ":", max(hoteltbl), "\n")
2 cat(levels(autorental)[which.max(autotbl)], ":", max(autotbl), "\n")

```

The hotel with the largest number of guests is Motel 6 with 50 guests, and Honda is rented to 66 people.

UPDATES

Visit <http://borisv.lk.net/latex.html> for updates, errata, and other information.

The fonts used in *Stumbling into data science* are New Baskerville, Futura, The Sans Mono Condensed, and Dogma. The book was typeset with $\text{\LaTeX} 2_{\epsilon}$ package nostarch by Boris Veytsman (2008/06/06 v1.3 *Typesetting books for No Starch Press*).

The book was produced as an example of the package nostarch.