



C o m m u n i t y   E x p e r i e n c e   D i s t i l l e d

# Machine Learning with R

## *Second Edition*

Discover how to build machine learning algorithms, prepare data, and dig deep into data prediction techniques with R

**Brett Lantz**

**[PACKT]** open source\*  
PUBLISHING community experience distilled

# Machine Learning with R

## *Second Edition*

Discover how to build machine learning algorithms,  
prepare data, and dig deep into data prediction  
techniques with R

**Brett Lantz**

**[PACKT]** open source   
PUBLISHING community experience distilled

BIRMINGHAM - MUMBAI

# Machine Learning with R

## *Second Edition*

Copyright © 2015 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: October 2013

Second edition: July 2015

Production reference: 1280715

Published by Packt Publishing Ltd.  
Livery Place  
35 Livery Street  
Birmingham B3 2PB, UK.

ISBN 978-1-78439-390-8

[www.packtpub.com](http://www.packtpub.com)

# Credits

**Author**

Brett Lantz

**Project Coordinator**

Vijay Kushlani

**Reviewers**

Vijayakumar Nattamai Jawaharlal

Kent S. Johnson

Mzabalazo Z. Ngwenya

Anuj Saxena

**Proofreader**

Safis Editing

**Indexer**

Monica Ajmera Mehta

**Commissioning Editor**

Ashwin Nair

**Production Coordinator**

Arvindkumar Gupta

**Acquisition Editor**

James Jones

**Cover Work**

Arvindkumar Gupta

**Content Development Editor**

Natasha D'Souza

**Technical Editor**

Rahul C. Shah

**Copy Editors**

Akshata Lobo

Swati Priya

# About the Author

**Brett Lantz** has spent more than 10 years using innovative data methods to understand human behavior. A trained sociologist, he was first enchanted by machine learning while studying a large database of teenagers' social networking website profiles. Since then, Brett has worked on interdisciplinary studies of cellular telephone calls, medical billing data, and philanthropic activity, among others. When not spending time with family, following college sports, or being entertained by his dachshunds, he maintains <http://dataspelunking.com/>, a website dedicated to sharing knowledge about the search for insight in data.

---

This book could not have been written without the support of my friends and family. In particular, my wife, Jessica, deserves many thanks for her endless patience and encouragement. My son, Will, who was born in the midst of the first edition and supplied much-needed diversions while writing this edition, will be a big brother shortly after this book is published. In spite of cautionary tales about correlation and causation, it seems that every time I expand my written library, my family likewise expands! I dedicate this book to my children in the hope that one day they will be inspired to tackle big challenges and follow their curiosity wherever it may lead.

I am also indebted to many others who supported this book indirectly. My interactions with educators, peers, and collaborators at the University of Michigan, the University of Notre Dame, and the University of Central Florida seeded many of the ideas I attempted to express in the text; any lack of clarity in their expression is purely mine. Additionally, without the work of the broader community of researchers who shared their expertise in publications, lectures, and source code, this book might not have existed at all. Finally, I appreciate the efforts of the R team and all those who have contributed to R packages, whose work has helped bring machine learning to the masses. I sincerely hope that my work is likewise a valuable piece in this mosaic.

---

# About the Reviewers

**Vijayakumar Nattamai Jawaharlal** is a software engineer with an experience of 2 decades in the IT industry. His background lies in machine learning, big data technologies, business intelligence, and data warehouse.

He develops scalable solutions for many distributed platforms, and is very passionate about scalable distributed machine learning.

**Kent S. Johnson** is a software developer who loves data analysis, statistics, and machine learning. He currently develops software to analyze tissue samples related to cancer research. According to him, a day spent with R and ggplot2 is a good day. For more information about him, visit <http://kentsjohnson.com>.

---

I'd like to thank, Gile, for always loving me.

---

**Mzabalazo Z. Ngwenya** holds a postgraduate degree in mathematical statistics from the University of Cape Town. He has worked extensively in the field of statistical consulting, and currently works as a biometrician at a research and development entity in South Africa. His areas of interest are primarily centered around statistical computing, and he has over 10 years of experience with R for data analysis and statistical research. Previously, he was involved in reviewing *Learning RStudio for R Statistical Computing*, *R Statistical Application Development by Example Beginner's Guide*, *R Graph Essentials*, *R Object-oriented Programming*, *Mastering Scientific Computing with R*, and *Machine Learning with R*, all by Packt Publishing.

**Anuj Saxena** is a data scientist at IGATE Corporation. He has an MS in analytics from the University of San Francisco and an MSc in Statistics from the NMIMS University in India. He is passionate about data science and likes using open source languages such as R and Python as primary tools for data science projects. In his spare time, he participates in predictive analytics competitions on kaggle.com. For more information about him, visit <http://www.anuj-saxena.com>.

---

I'd like to thank my father, Dr. Sharad Kumar, who inspired me at an early age to learn math and statistics and my mother, Mrs. Ranjana Saxena, who has been a backbone throughout my educational life.

I'd also like to thank my wonderful professors at the University of San Francisco and the NMIMS University who triggered my interest in this field and taught me the power of data and how it can be used to tell a wonderful story.

---

# www.PacktPub.com

## Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit [www.PacktPub.com](http://www.PacktPub.com).

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at [www.PacktPub.com](http://www.PacktPub.com) and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at [service@packtpub.com](mailto:service@packtpub.com) for more details.

At [www.PacktPub.com](http://www.PacktPub.com), you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<https://www2.packtpub.com/books/subscription/packtlib>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

## Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

## Free access for Packt account holders

If you have an account with Packt at [www.PacktPub.com](http://www.PacktPub.com), you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.





# Table of Contents

<b>Preface</b>	<b>ix</b>
<b>Chapter 1: Introducing Machine Learning</b>	<b>1</b>
The origins of machine learning	2
<b>Uses and abuses of machine learning</b>	<b>4</b>
Machine learning successes	5
The limits of machine learning	5
Machine learning ethics	7
<b>How machines learn</b>	<b>9</b>
Data storage	10
Abstraction	11
Generalization	13
Evaluation	14
<b>Machine learning in practice</b>	<b>16</b>
Types of input data	17
Types of machine learning algorithms	19
Matching input data to algorithms	21
<b>Machine learning with R</b>	<b>22</b>
Installing R packages	23
Loading and unloading R packages	24
<b>Summary</b>	<b>25</b>
<b>Chapter 2: Managing and Understanding Data</b>	<b>27</b>
<b>R data structures</b>	<b>28</b>
Vectors	28
Factors	30
Lists	32
Data frames	35
Matrixes and arrays	37

<b>Managing data with R</b>	<b>39</b>
Saving, loading, and removing R data structures	39
Importing and saving data from CSV files	41
<b>Exploring and understanding data</b>	<b>42</b>
Exploring the structure of data	43
Exploring numeric variables	44
Measuring the central tendency – mean and median	45
Measuring spread – quartiles and the five-number summary	47
Visualizing numeric variables – boxplots	49
Visualizing numeric variables – histograms	51
Understanding numeric data – uniform and normal distributions	53
Measuring spread – variance and standard deviation	54
Exploring categorical variables	56
Measuring the central tendency – the mode	58
Exploring relationships between variables	59
Visualizing relationships – scatterplots	59
Examining relationships – two-way cross-tabulations	61
<b>Summary</b>	<b>64</b>
<b>Chapter 3: Lazy Learning – Classification Using Nearest Neighbors</b>	<b>65</b>
<b>Understanding nearest neighbor classification</b>	<b>66</b>
The k-NN algorithm	66
Measuring similarity with distance	69
Choosing an appropriate k	70
Preparing data for use with k-NN	72
Why is the k-NN algorithm lazy?	74
<b>Example – diagnosing breast cancer with the k-NN algorithm</b>	<b>75</b>
Step 1 – collecting data	76
Step 2 – exploring and preparing the data	77
Transformation – normalizing numeric data	79
Data preparation – creating training and test datasets	80
Step 3 – training a model on the data	81
Step 4 – evaluating model performance	83
Step 5 – improving model performance	84
Transformation – z-score standardization	85
Testing alternative values of k	86
<b>Summary</b>	<b>87</b>
<b>Chapter 4: Probabilistic Learning – Classification Using Naive Bayes</b>	<b>89</b>
<b>Understanding Naive Bayes</b>	<b>90</b>
Basic concepts of Bayesian methods	90
Understanding probability	91
Understanding joint probability	92

---

Computing conditional probability with Bayes' theorem	94
<b>The Naive Bayes algorithm</b>	<b>97</b>
Classification with Naive Bayes	98
The Laplace estimator	100
Using numeric features with Naive Bayes	102
<b>Example – filtering mobile phone spam with the Naive Bayes algorithm</b>	<b>103</b>
Step 1 – collecting data	104
Step 2 – exploring and preparing the data	105
Data preparation – cleaning and standardizing text data	106
Data preparation – splitting text documents into words	112
Data preparation – creating training and test datasets	115
Visualizing text data – word clouds	116
Data preparation – creating indicator features for frequent words	119
Step 3 – training a model on the data	121
Step 4 – evaluating model performance	122
Step 5 – improving model performance	123
<b>Summary</b>	<b>124</b>
<b>Chapter 5: Divide and Conquer – Classification Using Decision Trees and Rules</b>	<b>125</b>
<b>Understanding decision trees</b>	<b>126</b>
Divide and conquer	127
The C5.0 decision tree algorithm	131
Choosing the best split	133
Pruning the decision tree	135
<b>Example – identifying risky bank loans using C5.0 decision trees</b>	<b>136</b>
Step 1 – collecting data	136
Step 2 – exploring and preparing the data	137
Data preparation – creating random training and test datasets	138
Step 3 – training a model on the data	140
Step 4 – evaluating model performance	144
Step 5 – improving model performance	145
Boosting the accuracy of decision trees	145
Making mistakes more costlier than others	147
<b>Understanding classification rules</b>	<b>149</b>
Separate and conquer	150
The 1R algorithm	153
The RIPPER algorithm	155
Rules from decision trees	157
What makes trees and rules greedy?	158
<b>Example – identifying poisonous mushrooms with rule learners</b>	<b>160</b>
Step 1 – collecting data	160
Step 2 – exploring and preparing the data	161

---

Step 3 – training a model on the data	162
Step 4 – evaluating model performance	165
Step 5 – improving model performance	166
<b>Summary</b>	<b>169</b>
<b>Chapter 6: Forecasting Numeric Data – Regression Methods</b>	<b>171</b>
<b>Understanding regression</b>	<b>172</b>
Simple linear regression	174
Ordinary least squares estimation	177
Correlations	179
Multiple linear regression	181
<b>Example – predicting medical expenses using linear regression</b>	<b>186</b>
Step 1 – collecting data	186
Step 2 – exploring and preparing the data	187
Exploring relationships among features – the correlation matrix	189
Visualizing relationships among features – the scatterplot matrix	190
Step 3 – training a model on the data	193
Step 4 – evaluating model performance	196
Step 5 – improving model performance	197
Model specification – adding non-linear relationships	198
Transformation – converting a numeric variable to a binary indicator	198
Model specification – adding interaction effects	199
Putting it all together – an improved regression model	200
<b>Understanding regression trees and model trees</b>	<b>201</b>
Adding regression to trees	202
<b>Example – estimating the quality of wines with regression trees and model trees</b>	<b>205</b>
Step 1 – collecting data	205
Step 2 – exploring and preparing the data	206
Step 3 – training a model on the data	208
Visualizing decision trees	210
Step 4 – evaluating model performance	212
Measuring performance with the mean absolute error	213
Step 5 – improving model performance	214
<b>Summary</b>	<b>218</b>
<b>Chapter 7: Black Box Methods – Neural Networks and Support Vector Machines</b>	<b>219</b>
<b>Understanding neural networks</b>	<b>220</b>
From biological to artificial neurons	221
Activation functions	223

Network topology	225
The number of layers	226
The direction of information travel	227
The number of nodes in each layer	228
Training neural networks with backpropagation	229
<b>Example – Modeling the strength of concrete with ANNs</b>	<b>231</b>
Step 1 – collecting data	232
Step 2 – exploring and preparing the data	232
Step 3 – training a model on the data	234
Step 4 – evaluating model performance	237
Step 5 – improving model performance	238
<b>Understanding Support Vector Machines</b>	<b>239</b>
Classification with hyperplanes	240
The case of linearly separable data	242
The case of nonlinearly separable data	244
Using kernels for non-linear spaces	245
<b>Example – performing OCR with SVMs</b>	<b>248</b>
Step 1 – collecting data	249
Step 2 – exploring and preparing the data	250
Step 3 – training a model on the data	252
Step 4 – evaluating model performance	254
Step 5 – improving model performance	256
<b>Summary</b>	<b>257</b>
<b>Chapter 8: Finding Patterns – Market Basket Analysis Using Association Rules</b>	<b>259</b>
<b>Understanding association rules</b>	<b>260</b>
The Apriori algorithm for association rule learning	261
Measuring rule interest – support and confidence	263
Building a set of rules with the Apriori principle	265
<b>Example – identifying frequently purchased groceries with association rules</b>	<b>266</b>
Step 1 – collecting data	266
Step 2 – exploring and preparing the data	267
Data preparation – creating a sparse matrix for transaction data	268
Visualizing item support – item frequency plots	272
Visualizing the transaction data – plotting the sparse matrix	273
Step 3 – training a model on the data	274
Step 4 – evaluating model performance	277
Step 5 – improving model performance	280
Sorting the set of association rules	280
Taking subsets of association rules	281
Saving association rules to a file or data frame	283
<b>Summary</b>	<b>284</b>

<b>Chapter 9: Finding Groups of Data – Clustering with k-means</b>	<b>285</b>
<b>Understanding clustering</b>	<b>286</b>
Clustering as a machine learning task	286
The k-means clustering algorithm	289
Using distance to assign and update clusters	290
Choosing the appropriate number of clusters	294
<b>Example – finding teen market segments using k-means clustering</b>	<b>296</b>
Step 1 – collecting data	297
Step 2 – exploring and preparing the data	297
Data preparation – dummy coding missing values	299
Data preparation – imputing the missing values	300
Step 3 – training a model on the data	302
Step 4 – evaluating model performance	304
Step 5 – improving model performance	308
<b>Summary</b>	<b>310</b>
<b>Chapter 10: Evaluating Model Performance</b>	<b>311</b>
<b>Measuring performance for classification</b>	<b>312</b>
Working with classification prediction data in R	313
A closer look at confusion matrices	317
Using confusion matrices to measure performance	319
Beyond accuracy – other measures of performance	321
The kappa statistic	323
Sensitivity and specificity	326
Precision and recall	328
The F-measure	330
Visualizing performance trade-offs	331
ROC curves	332
<b>Estimating future performance</b>	<b>336</b>
The holdout method	336
Cross-validation	340
Bootstrap sampling	343
<b>Summary</b>	<b>344</b>
<b>Chapter 11: Improving Model Performance</b>	<b>347</b>
<b>Tuning stock models for better performance</b>	<b>348</b>
Using caret for automated parameter tuning	349
Creating a simple tuned model	352
Customizing the tuning process	355
<b>Improving model performance with meta-learning</b>	<b>359</b>
Understanding ensembles	359
Bagging	362
Boosting	366

---

Random forests	369
Training random forests	370
Evaluating random forest performance	373
<b>Summary</b>	<b>375</b>
<b>Chapter 12: Specialized Machine Learning Topics</b>	<b>377</b>
<b>Working with proprietary files and databases</b>	<b>378</b>
Reading from and writing to Microsoft Excel, SAS, SPSS, and Stata files	378
Querying data in SQL databases	379
<b>Working with online data and services</b>	<b>381</b>
Downloading the complete text of web pages	382
Scraping data from web pages	383
Parsing XML documents	387
Parsing JSON from web APIs	388
<b>Working with domain-specific data</b>	<b>392</b>
Analyzing bioinformatics data	393
Analyzing and visualizing network data	393
<b>Improving the performance of R</b>	<b>398</b>
Managing very large datasets	398
Generalizing tabular data structures with dplyr	399
Making data frames faster with data.table	401
Creating disk-based data frames with ff	402
Using massive matrices with bigmemory	404
Learning faster with parallel computing	404
Measuring execution time	406
Working in parallel with multicore and snow	406
Taking advantage of parallel with foreach and doParallel	410
Parallel cloud computing with MapReduce and Hadoop	411
GPU computing	412
Deploying optimized learning algorithms	413
Building bigger regression models with biglm	414
Growing bigger and faster random forests with bigrf	414
Training and evaluating models in parallel with caret	414
<b>Summary</b>	<b>416</b>
<b>Index</b>	<b>417</b>

---





# Preface

Machine learning, at its core, is concerned with the algorithms that transform information into actionable intelligence. This fact makes machine learning well-suited to the present-day era of big data. Without machine learning, it would be nearly impossible to keep up with the massive stream of information.

Given the growing prominence of R—a cross-platform, zero-cost statistical programming environment—there has never been a better time to start using machine learning. R offers a powerful but easy-to-learn set of tools that can assist you with finding data insights.

By combining hands-on case studies with the essential theory that you need to understand how things work under the hood, this book provides all the knowledge that you will need to start applying machine learning to your own projects.

## What this book covers

*Chapter 1, Introducing Machine Learning*, presents the terminology and concepts that define and distinguish machine learners, as well as a method for matching a learning task with the appropriate algorithm.

*Chapter 2, Managing and Understanding Data*, provides an opportunity to get your hands dirty working with data in R. Essential data structures and procedures used for loading, exploring, and understanding data are discussed.

*Chapter 3, Lazy Learning – Classification Using Nearest Neighbors*, teaches you how to understand and apply a simple yet powerful machine learning algorithm to your first real-world task—identifying malignant samples of cancer.

*Chapter 4, Probabilistic Learning – Classification Using Naïve Bayes*, reveals the essential concepts of probability that are used in the cutting-edge spam filtering systems. You'll learn the basics of text mining in the process of building your own spam filter.

*Chapter 5, Divide and Conquer – Classification Using Decision Trees and Rules*, explores a couple of learning algorithms whose predictions are not only accurate, but also easily explained. We'll apply these methods to tasks where transparency is important.

*Chapter 6, Forecasting Numeric Data – Regression Methods*, introduces machine learning algorithms used for making numeric predictions. As these techniques are heavily embedded in the field of statistics, you will also learn the essential metrics needed to make sense of numeric relationships.

*Chapter 7, Black Box Methods – Neural Networks and Support Vector Machines*, covers two complex but powerful machine learning algorithms. Though the math may appear intimidating, we will work through examples that illustrate their inner workings in simple terms.

*Chapter 8, Finding Patterns – Market Basket Analysis Using Association Rules*, exposes the algorithm used in the recommendation systems employed by many retailers. If you've ever wondered how retailers seem to know your purchasing habits better than you know yourself, this chapter will reveal their secrets.

*Chapter 9, Finding Groups of Data – Clustering with k-means*, is devoted to a procedure that locates clusters of related items. We'll utilize this algorithm to identify profiles within an online community.

*Chapter 10, Evaluating Model Performance*, provides information on measuring the success of a machine learning project and obtaining a reliable estimate of the learner's performance on future data.

*Chapter 11, Improving Model Performance*, reveals the methods employed by the teams at the top of machine learning competition leaderboards. If you have a competitive streak, or simply want to get the most out of your data, you'll need to add these techniques to your repertoire.

*Chapter 12, Specialized Machine Learning Topics*, explores the frontiers of machine learning. From working with big data to making R work faster, the topics covered will help you push the boundaries of what is possible with R.

## What you need for this book

The examples in this book were written for and tested with R version 3.2.0 on Microsoft Windows and Mac OS X, though they are likely to work with any recent version of R.

## Who this book is for

This book is intended for anybody hoping to use data for action. Perhaps you already know a bit about machine learning, but have never used R; or perhaps you know a little about R, but are new to machine learning. In any case, this book will get you up and running quickly. It would be helpful to have a bit of familiarity with basic math and programming concepts, but no prior experience is required. All you need is curiosity.

## Conventions

In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "The most direct way to install a package is via the `install.packages()` function."

A block of code is set as follows:

```
subject_name, temperature, flu_status, gender, blood_type
John Doe,      98.1,      FALSE,      MALE,      O
Jane Doe,      98.6,      FALSE,      FEMALE,     AB
Steve Graves,  101.4,     TRUE,      MALE,      A
```

Any command-line input or output is written as follows:

```
> summary(wbcd_z$area_mean)
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.4530 -0.6666 -0.2949  0.0000  0.3632  5.2460
```

**New terms** and **important words** are shown in bold. Words that you see on the screen, for example, in menus or dialog boxes, appear in the text like this: "The **Task Views** link on the left side of the CRAN page provides a curated list of packages."



Warnings or important notes appear in a box like this.



Tips and tricks appear like this.

## Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book – what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail [feedback@packtpub.com](mailto:feedback@packtpub.com), and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at [www.packtpub.com/authors](http://www.packtpub.com/authors).

## Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

## Downloading the example code

You can download the example code files from your account at <http://www.packtpub.com> for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

New to the second edition of this book, the example code is also available via GitHub at <https://github.com/dataspelunking/MLwR/>. Check here for the most up-to-date R code, as well as issue tracking and a public wiki. Please join the community!

## Downloading the color images of this book

We also provide you with a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from [http://www.packtpub.com/sites/default/files/downloads/Machine\\_Learning\\_With\\_R\\_Second\\_Edition\\_ColoredImages.pdf](http://www.packtpub.com/sites/default/files/downloads/Machine_Learning_With_R_Second_Edition_ColoredImages.pdf).

## Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to <https://www.packtpub.com/books/content/support> and enter the name of the book in the search field. The required information will appear under the **Errata** section.

## Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at [copyright@packtpub.com](mailto:copyright@packtpub.com) with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

## Questions

If you have a problem with any aspect of this book, you can contact us at [questions@packtpub.com](mailto:questions@packtpub.com), and we will do our best to address the problem.



# 1

## Introducing Machine Learning

If science fiction stories are to be believed, the invention of artificial intelligence inevitably leads to apocalyptic wars between machines and their makers. In the early stages, computers are taught to play simple games of tic-tac-toe and chess. Later, machines are given control of traffic lights and communications, followed by military drones and missiles. The machines' evolution takes an ominous turn once the computers become sentient and learn how to teach themselves. Having no more need for human programmers, humankind is then "deleted."

Thankfully, at the time of this writing, machines still require user input.

Though your impressions of machine learning may be colored by these mass media depictions, today's algorithms are too application-specific to pose any danger of becoming self-aware. The goal of today's machine learning is not to create an artificial brain, but rather to assist us in making sense of the world's massive data stores.

Putting popular misconceptions aside, by the end of this chapter, you will gain a more nuanced understanding of machine learning. You also will be introduced to the fundamental concepts that define and differentiate the most commonly used machine learning approaches.

You will learn:

- The origins and practical applications of machine learning
- How computers turn data into knowledge and action
- How to match a machine learning algorithm to your data

The field of machine learning provides a set of algorithms that transform data into actionable knowledge. Keep reading to see how easy it is to use R to start applying machine learning to real-world problems.



## The origins of machine learning

Since birth, we are inundated with data. Our body's sensors – the eyes, ears, nose, tongue, and nerves – are continually assailed with raw data that our brain translates into sights, sounds, smells, tastes, and textures. Using language, we are able to share these experiences with others.

From the advent of written language, human observations have been recorded. Hunters monitored the movement of animal herds, early astronomers recorded the alignment of planets and stars, and cities recorded tax payments, births, and deaths. Today, such observations, and many more, are increasingly automated and recorded systematically in the ever-growing computerized databases.

The invention of electronic sensors has additionally contributed to an explosion in the volume and richness of recorded data. Specialized sensors see, hear, smell, taste, and feel. These sensors process the data far differently than a human being would. Unlike a human's limited and subjective attention, an electronic sensor never takes a break and never lets its judgment skew its perception.

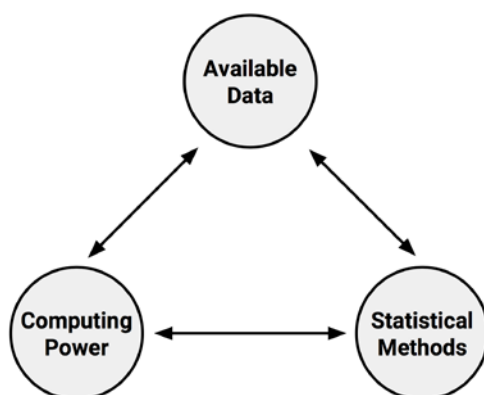


Although sensors are not clouded by subjectivity, they do not necessarily report a single, definitive depiction of reality. Some have an inherent measurement error, due to hardware limitations. Others are limited by their scope. A black and white photograph provides a different depiction of its subject than one shot in color. Similarly, a microscope provides a far different depiction of reality than a telescope.

Between databases and sensors, many aspects of our lives are recorded. Governments, businesses, and individuals are recording and reporting information, from the monumental to the mundane. Weather sensors record temperature and pressure data, surveillance cameras watch sidewalks and subway tunnels, and all manner of electronic behaviors are monitored: transactions, communications, friendships, and many others.

This deluge of data has led some to state that we have entered an era of **Big Data**, but this may be a bit of a misnomer. Human beings have always been surrounded by large amounts of data. What makes the current era unique is that we have vast amounts of *recorded* data, much of which can be directly accessed by computers. Larger and more interesting data sets are increasingly accessible at the tips of our fingers, only a web search away. This wealth of information has the potential to inform action, given a systematic way of making sense from it all.

The field of study interested in the development of computer algorithms to transform data into intelligent action is known as **machine learning**. This field originated in an environment where available data, statistical methods, and computing power rapidly and simultaneously evolved. Growth in data necessitated additional computing power, which in turn spurred the development of **statistical methods** to analyze large datasets. This created a cycle of advancement, allowing even larger and more interesting data to be collected.



A closely related sibling of machine learning, **data mining**, is concerned with the generation of novel insights from large databases. As the implies, data mining involves a systematic hunt for nuggets of actionable intelligence. Although there is some disagreement over how widely machine learning and data mining overlap, a potential point of distinction is that machine learning focuses on teaching computers how to use data to solve a problem, while data mining focuses on teaching computers to identify patterns that humans then use to solve a problem.

Virtually all data mining involves the use of machine learning, but not all machine learning involves data mining. For example, you might apply machine learning to data mine automobile traffic data for patterns related to accident rates; on the other hand, if the computer is learning how to drive the car itself, this is purely machine learning without data mining.



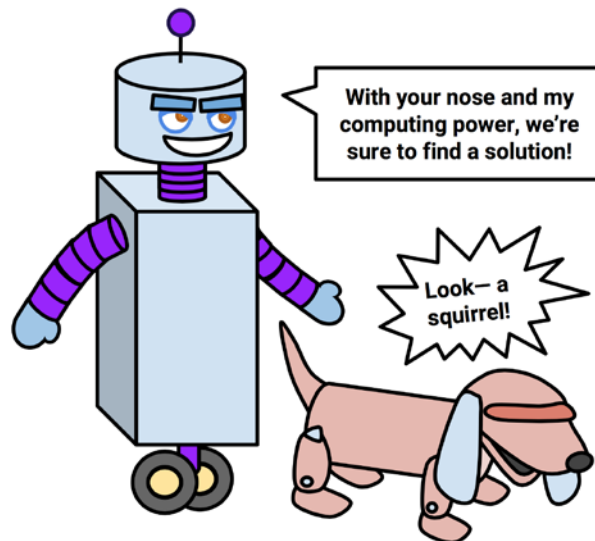
The phrase "data mining" is also sometimes used as a pejorative to describe the deceptive practice of cherry-picking data to support a theory.

## Uses and abuses of machine learning

Most people have heard of the chess-playing computer **Deep Blue**—the first to win a game against a world champion—or **Watson**, the computer that defeated two human opponents on the television trivia game show Jeopardy. Based on these stunning accomplishments, some have speculated that computer intelligence will replace humans in many information technology occupations, just as machines replaced humans in the fields, and robots replaced humans on the assembly line.

The truth is that even as machines reach such impressive milestones, they are still relatively limited in their ability to thoroughly understand a problem. They are pure intellectual horsepower without direction. A computer may be more capable than a human of finding subtle patterns in large databases, but it still needs a human to motivate the analysis and turn the result into meaningful action.

Machines are not good at asking questions, or even knowing what questions to ask. They are much better at answering them, provided the question is stated in a way the computer can comprehend. Present-day machine learning algorithms partner with people much like a bloodhound partners with its trainer; the dog's sense of smell may be many times stronger than its master's, but without being carefully directed, the hound may end up chasing its tail.



To better understand the real-world applications of machine learning, we'll now consider some cases where it has been used successfully, some places where it still has room for improvement, and some situations where it may do more harm than good.

## Machine learning successes

Machine learning is most successful when it augments rather than replaces the specialized knowledge of a subject-matter expert. It works with medical doctors at the forefront of the fight to eradicate cancer, assists engineers and programmers with our efforts to create smarter homes and automobiles, and helps social scientists build knowledge of how societies function. Toward these ends, it is employed in countless businesses, scientific laboratories, hospitals, and governmental organizations. Any organization that generates or aggregates data likely employs at least one machine learning algorithm to help make sense of it.

Though it is impossible to list every use case of machine learning, a survey of recent success stories includes several prominent applications:

- Identification of unwanted spam messages in e-mail
- Segmentation of customer behavior for targeted advertising
- Forecasts of weather behavior and long-term climate changes
- Reduction of fraudulent credit card transactions
- Actuarial estimates of financial damage of storms and natural disasters
- Prediction of popular election outcomes
- Development of algorithms for auto-piloting drones and self-driving cars
- Optimization of energy use in homes and office buildings
- Projection of areas where criminal activity is most likely
- Discovery of genetic sequences linked to diseases

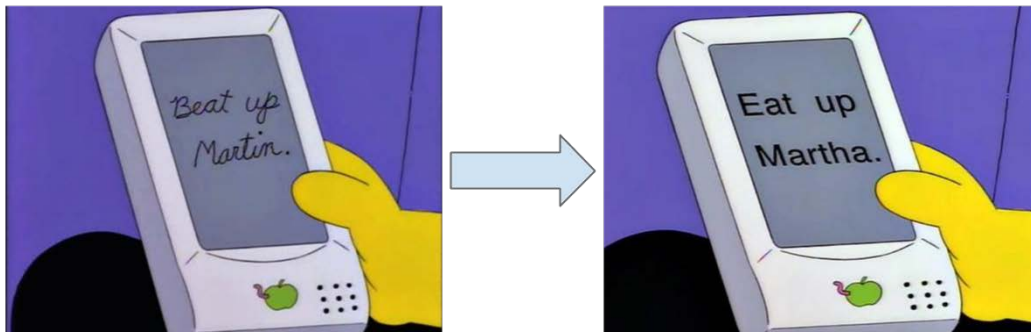
By the end of this book, you will understand the basic machine learning algorithms that are employed to teach computers to perform these tasks. For now, it suffices to say that no matter what the context is, the machine learning process is the same. Regardless of the task, an algorithm takes data and identifies patterns that form the basis for further action.

## The limits of machine learning

Although machine learning is used widely and has tremendous potential, it is important to understand its limits. Machine learning, at this time, is not in any way a substitute for a human brain. It has very little flexibility to extrapolate outside of the strict parameters it learned and knows no common sense. With this in mind, one should be extremely careful to recognize exactly what the algorithm has learned before setting it loose in the real-world settings.

Without a lifetime of past experiences to build upon, computers are also limited in their ability to make simple common sense inferences about logical next steps. Take, for instance, the banner advertisements seen on many web sites. These may be served, based on the patterns learned by data mining the browsing history of millions of users. According to this data, someone who views the websites selling shoes should see advertisements for shoes, and those viewing websites for mattresses should see advertisements for mattresses. The problem is that this becomes a never-ending cycle in which additional shoe or mattress advertisements are served rather than advertisements for shoelaces and shoe polish, or bed sheets and blankets.

Many are familiar with the deficiencies of machine learning's ability to understand or translate language or to recognize speech and handwriting. Perhaps the earliest example of this type of failure is in a 1994 episode of the television show, *The Simpsons*, which showed a parody of the Apple Newton tablet. For its time, the Newton was known for its state-of-the-art handwriting recognition. Unfortunately for Apple, it would occasionally fail to great effect. The television episode illustrated this through a sequence in which a bully's note to **Beat up Martin** was misinterpreted by the Newton as **Eat up Martha**, as depicted in the following screenshots:



Screenshots from "Lisa on Ice" *The Simpsons*, 20th Century Fox (1994)

Machines' ability to understand language has improved enough since 1994, such that Google, Apple, and Microsoft are all confident enough to offer virtual concierge services operated via voice recognition. Still, even these services routinely struggle to answer relatively simple questions. Even more, online translation services sometimes misinterpret sentences that a toddler would readily understand. The predictive text feature on many devices has also led to a number of humorous *autocorrect fail* sites that illustrate the computer's ability to understand basic language but completely misunderstand context.

Some of these mistakes are to be expected, for sure. Language is complicated with multiple layers of text and subtext and even human beings, sometimes, understand the context incorrectly. This said, these types of failures in machines illustrate the important fact that machine learning is only as good as the data it learns from. If the context is not directly implicit in the input data, then just like a human, the computer will have to make its best guess.

## Machine learning ethics

At its core, machine learning is simply a tool that assists us in making sense of the world's complex data. Like any tool, it can be used for good or evil. Machine learning may lead to problems when it is applied so broadly or callously that humans are treated as lab rats, automata, or mindless consumers. A process that may seem harmless may lead to unintended consequences when automated by an emotionless computer. For this reason, those using machine learning or data mining would be remiss not to consider the ethical implications of the art.

Due to the relative youth of machine learning as a discipline and the speed at which it is progressing, the associated legal issues and social norms are often quite uncertain and constantly in flux. Caution should be exercised while obtaining or analyzing data in order to avoid breaking laws, violating terms of service or data use agreements, and abusing the trust or violating the privacy of customers or the public.



The informal corporate motto of Google, an organization that collects perhaps more data on individuals than any other, is "don't be evil." While this seems clear enough, it may not be sufficient. A better approach may be to follow the *Hippocratic Oath*, a medical principle that states "above all, do no harm."

Retailers routinely use machine learning for advertising, targeted promotions, inventory management, or the layout of the items in the store. Many have even equipped checkout lanes with devices that print coupons for promotions based on the customer's buying history. In exchange for a bit of personal data, the customer receives discounts on the specific products he or she wants to buy. At first, this appears relatively harmless. But consider what happens when this practice is taken a little bit further.

One possibly apocryphal tale concerns a large retailer in the U.S. that employed machine learning to identify expectant mothers for coupon mailings. The retailer hoped that if these mothers-to-be received substantial discounts, they would become loyal customers, who would later purchase profitable items like diapers, baby formula, and toys.

Equipped with machine learning methods, the retailer identified items in the customer purchase history that could be used to predict with a high degree of certainty, not only whether a woman was pregnant, but also the approximate timing for when the baby was due.

After the retailer used this data for a promotional mailing, an angry man contacted the chain and demanded to know why his teenage daughter received coupons for maternity items. He was furious that the retailer seemed to be encouraging teenage pregnancy! As the story goes, when the retail chain's manager called to offer an apology, it was the father that ultimately apologized because, after confronting his daughter, he discovered that she was indeed pregnant!

Whether completely true or not, the lesson learned from the preceding tale is that common sense should be applied before blindly applying the results of a machine learning analysis. This is particularly true in cases where sensitive information such as health data is concerned. With a bit more care, the retailer could have foreseen this scenario, and used greater discretion while choosing how to reveal the pattern its machine learning analysis had discovered.

Certain jurisdictions may prevent you from using racial, ethnic, religious, or other protected class data for business reasons. Keep in mind that excluding this data from your analysis may not be enough, because machine learning algorithms might inadvertently learn this information independently. For instance, if a certain segment of people generally live in a certain region, buy a certain product, or otherwise behave in a way that uniquely identifies them as a group, some machine learning algorithms can infer the protected information from these other factors. In such cases, you may need to fully "de-identify" these people by excluding any *potentially* identifying data in addition to the protected information.

Apart from the legal consequences, using data inappropriately may hurt the bottom line. Customers may feel uncomfortable or become spooked if the aspects of their lives they consider private are made public. In recent years, several high-profile web applications have experienced a mass exodus of users who felt exploited when the applications' terms of service agreements changed, and their data was used for purposes beyond what the users had originally agreed upon. The fact that privacy expectations differ by context, age cohort, and locale adds complexity in deciding the appropriate use of personal data. It would be wise to consider the cultural implications of your work before you begin your project.




The fact that you *can* use data for a particular end does not always mean that you *should*.



## How machines learn

A formal definition of machine learning proposed by computer scientist Tom M. Mitchell states that a machine learns whenever it is able to utilize its an experience such that its performance improves on similar experiences in the future. Although this definition is intuitive, it completely ignores the process of exactly how experience can be translated into future action – and of course learning is always easier said than done!

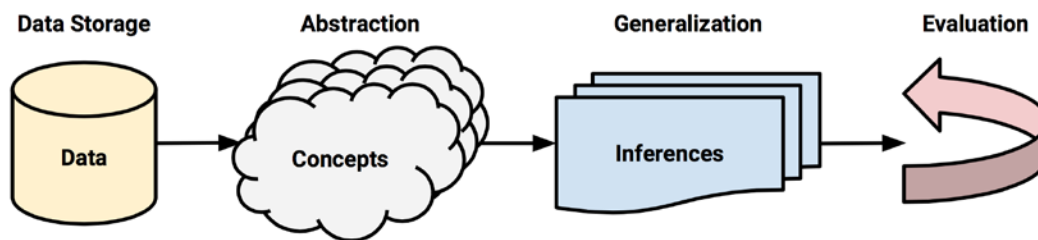
While human brains are naturally capable of learning from birth, the conditions necessary for computers to learn must be made explicit. For this reason, although it is not strictly necessary to understand the theoretical basis of learning, this foundation helps understand, distinguish, and implement machine learning algorithms.

[  As you compare machine learning to human learning, you may discover yourself examining your own mind in a different light. ]

Regardless of whether the learner is a human or machine, the basic learning process is similar. It can be divided into four interrelated components:

- **Data storage** utilizes observation, memory, and recall to provide a factual basis for further reasoning.
- **Abstraction** involves the translation of stored data into broader representations and concepts.
- **Generalization** uses abstracted data to create knowledge and inferences that drive action in new contexts.
- **Evaluation** provides a feedback mechanism to measure the utility of learned knowledge and inform potential improvements.

The following figure illustrates the steps in the learning process:





Keep in mind that although the learning process has been conceptualized as four distinct components, they are merely organized this way for illustrative purposes. In reality, the entire learning process is inextricably linked. In human beings, the process occurs subconsciously. We recollect, deduce, induct, and intuit with the confines of our mind's eye, and because this process is hidden, any differences from person to person are attributed to a vague notion of subjectivity. In contrast, with computers these processes are explicit, and because the entire process is transparent, the learned knowledge can be examined, transferred, and utilized for future action.

## Data storage

All learning must begin with data. Humans and computers alike utilize **data storage** as a foundation for more advanced reasoning. In a human being, this consists of a brain that uses electrochemical signals in a network of biological cells to store and process observations for short- and long-term future recall. Computers have similar capabilities of short- and long-term recall using hard disk drives, flash memory, and random access memory (RAM) in combination with a central processing unit (CPU).

It may seem obvious to say so, but the ability to store and retrieve data alone is not sufficient for learning. Without a higher level of understanding, knowledge is limited exclusively to recall, meaning exclusively what is seen before and nothing else. The data is merely ones and zeros on a disk. They are stored memories with no broader meaning.

To better understand the nuances of this idea, it may help to think about the last time you studied for a difficult test, perhaps for a university final exam or a career certification. Did you wish for an eidetic (photographic) memory? If so, you may be disappointed to learn that perfect recall is unlikely to be of much assistance. Even if you could memorize material perfectly, your rote learning is of no use, unless you know in advance the exact questions and answers that will appear in the exam. Otherwise, you would be stuck in an attempt to memorize answers to every question that could conceivably be asked. Obviously, this is an unsustainable strategy.

Instead, a better approach is to spend time selectively, memorizing a small set of representative ideas while developing strategies on how the ideas relate and how to use the stored information. In this way, large ideas can be understood without needing to memorize them by rote.

## Abstraction

This work of assigning meaning to stored data occurs during the **abstraction** process, in which raw data comes to have a more abstract meaning. This type of connection, say between an object and its representation, is exemplified by the famous René Magritte painting *The Treachery of Images*:



Source: <http://collections.lacma.org/node/239578>

The painting depicts a tobacco pipe with the caption *Ceci n'est pas une pipe* ("this is not a pipe"). The point Magritte was illustrating is that a representation of a pipe is not truly a pipe. Yet, in spite of the fact that the pipe is not real, anybody viewing the painting easily recognizes it as a pipe. This suggests that the observer's mind is able to connect the *picture* of a pipe to the *idea* of a pipe, to a memory of a *physical* pipe that could be held in the hand. Abstracted connections like these are the basis of **knowledge representation**, the formation of logical structures that assist in turning raw sensory information into a meaningful insight.

During a machine's process of knowledge representation, the computer summarizes stored raw data using a **model**, an explicit description of the patterns within the data. Just like Magritte's pipe, the model representation takes on a life beyond the raw data. It represents an idea greater than the sum of its parts.

There are many different types of models. You may be already familiar with some. Examples include:

- Mathematical equations
- Relational diagrams such as trees and graphs
- Logical if/else rules
- Groupings of data known as clusters

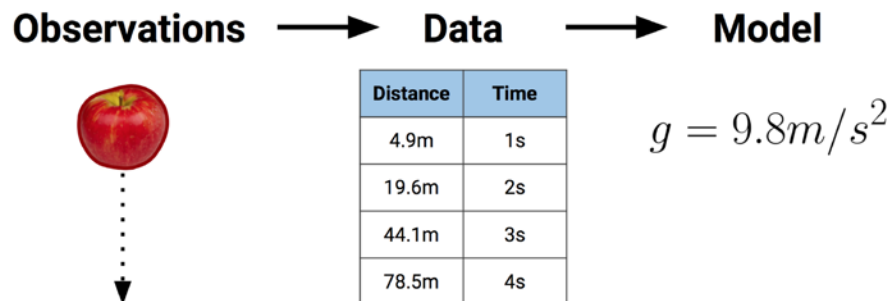
The choice of model is typically not left up to the machine. Instead, the learning task and data on hand inform model selection. Later in this chapter, we will discuss methods to choose the type of model in more detail.

The process of fitting a model to a dataset is known as **training**. When the model has been trained, the data is transformed into an abstract form that summarizes the original information.



You might wonder why this step is called training rather than learning. First, note that the process of learning does not end with data abstraction; the learner must still generalize and evaluate its training. Second, the word training better connotes the fact that the human teacher trains the machine student to understand the data in a specific way.

It is important to note that a learned model does not itself provide new data, yet it does result in new knowledge. How can this be? The answer is that imposing an assumed structure on the underlying data gives insight into the unseen by supposing a concept about how data elements are related. Take for instance the discovery of gravity. By fitting equations to observational data, Sir Isaac Newton inferred the concept of gravity. But the force we now know as gravity was always present. It simply wasn't recognized until Newton recognized it as an abstract concept that relates some data to others – specifically, by becoming the  $g$  term in a model that explains observations of falling objects.




Most models may not result in the development of theories that shake up scientific thought for centuries. Still, your model might result in the discovery of previously unseen relationships among data. A model trained on genomic data might find several genes that, when combined, are responsible for the onset of diabetes; banks might discover a seemingly innocuous type of transaction that systematically appears prior to fraudulent activity; and psychologists might identify a combination of personality characteristics indicating a new disorder. These underlying patterns were always present, but by simply presenting information in a different format, a new idea is conceptualized.

## Generalization

The learning process is not complete until the learner is able to use its abstracted knowledge for future action. However, among the countless underlying patterns that might be identified during the abstraction process and the myriad ways to model these patterns, some will be more useful than others. Unless the production of abstractions is limited, the learner will be unable to proceed. It would be stuck where it started – with a large pool of information, but no actionable insight.

The term **generalization** describes the process of turning abstracted knowledge into a form that can be utilized for future action, on tasks that are similar, but not identical, to those it has seen before. Generalization is a somewhat vague process that is a bit difficult to describe. Traditionally, it has been imagined as a search through the entire set of models (that is, theories or inferences) that could be abstracted during training. In other words, if you can imagine a hypothetical set containing every possible theory that could be established from the data, generalization involves the reduction of this set into a manageable number of important findings.

In generalization, the learner is tasked with limiting the patterns it discovers to only those that will be most relevant to its future tasks. Generally, it is not feasible to reduce the number of patterns by examining them one-by-one and ranking them by future utility. Instead, machine learning algorithms generally employ shortcuts that reduce the search space more quickly. Toward this end, the algorithm will employ **heuristics**, which are educated guesses about where to find the most useful inferences.

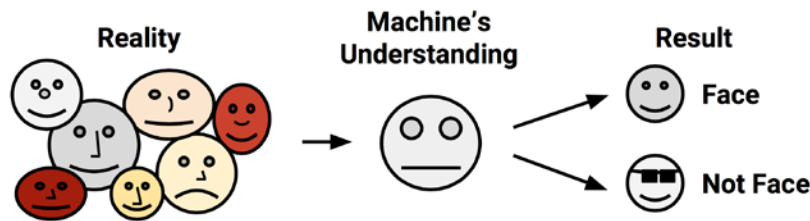
 Because heuristics utilize approximations and other rules of thumb, they do not guarantee to find the single best model. However, without taking these shortcuts, finding useful information in a large dataset would be infeasible.

Heuristics are routinely used by human beings to quickly generalize experience to new scenarios. If you have ever utilized your gut instinct to make a snap decision prior to fully evaluating your circumstances, you were intuitively using mental heuristics.

The incredible human ability to make quick decisions often relies not on computer-like logic, but rather on heuristics guided by emotions. Sometimes, this can result in illogical conclusions. For example, more people express fear of airline travel versus automobile travel, despite automobiles being statistically more dangerous. This can be explained by the availability heuristic, which is the tendency of people to estimate the likelihood of an event by how easily its examples can be recalled. Accidents involving air travel are highly publicized. Being traumatic events, they are likely to be recalled very easily, whereas car accidents barely warrant a mention in the newspaper.

The folly of misapplied heuristics is not limited to human beings. The heuristics employed by machine learning algorithms also sometimes result in erroneous conclusions. The algorithm is said to have a **bias** if the conclusions are systematically erroneous, or wrong in a predictable manner.

For example, suppose that a machine learning algorithm learned to identify faces by finding two dark circles representing eyes, positioned above a straight line indicating a mouth. The algorithm might then have trouble with, or be *biased against*, faces that do not conform to its model. Faces with glasses, turned at an angle, looking sideways, or with various skin tones might not be detected by the algorithm. Similarly, it could be *biased toward* faces with certain skin tones, face shapes, or other characteristics that do not conform to its understanding of the world.



In modern usage, the word bias has come to carry quite negative connotations. Various forms of media frequently claim to be free from bias, and claim to report the facts objectively, untainted by emotion. Still, consider for a moment the possibility that a little bias might be useful. Without a bit of arbitrariness, might it be a bit difficult to decide among several competing choices, each with distinct strengths and weaknesses? Indeed, some recent studies in the field of psychology have suggested that individuals born with damage to portions of the brain responsible for emotion are ineffectual in decision making, and might spend hours debating simple decisions such as what color shirt to wear or where to eat lunch. Paradoxically, bias is what blinds us from some information while also allowing us to utilize other information for action. It is how machine learning algorithms choose among the countless ways to understand a set of data.

## Evaluation

Bias is a necessary evil associated with the abstraction and generalization processes inherent in any learning task. In order to drive action in the face of limitless possibility, each learner must be biased in a particular way. Consequently, each learner has its weaknesses and there is no single learning algorithm to rule them all. Therefore, the final step in the generalization process is to **evaluate** or measure the learner's success in spite of its biases and use this information to inform additional training if needed.



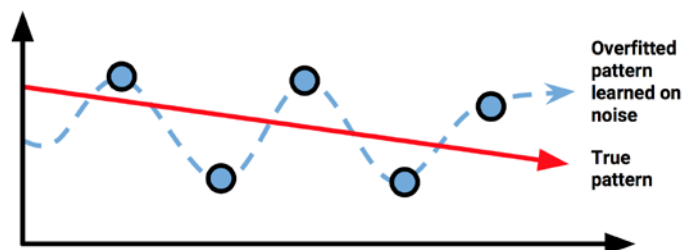
Once you've had success with one machine learning technique, you might be tempted to apply it to everything. It is important to resist this temptation because no machine learning approach is the best for every circumstance. This fact is described by the *No Free Lunch* theorem, introduced by David Wolpert in 1996. For more information, visit: <http://www.no-free-lunch.org>.

Generally, evaluation occurs after a model has been trained on an initial training dataset. Then, the model is evaluated on a new test dataset in order to judge how well its characterization of the training data generalizes to new, unseen data. It's worth noting that it is exceedingly rare for a model to perfectly generalize to every unforeseen case.

In parts, models fail to perfectly generalize due to the problem of **noise**, a term that describes unexplained or unexplainable variations in data. Noisy data is caused by seemingly random events, such as:

- Measurement error due to imprecise sensors that sometimes add or subtract a bit from the readings
- Issues with human subjects, such as survey respondents reporting random answers to survey questions, in order to finish more quickly
- Data quality problems, including missing, null, truncated, incorrectly coded, or corrupted values
- Phenomena that are so complex or so little understood that they impact the data in ways that appear to be unsystematic

Trying to model noise is the basis of a problem called **overfitting**. Because most noisy data is unexplainable by definition, attempting to explain the noise will result in erroneous conclusions that do not generalize well to new cases. Efforts to explain the noise will also typically result in more complex models that will miss the true pattern that the learner tries to identify. A model that seems to perform well during training, but does poorly during evaluation, is said to be overfitted to the training dataset, as it does not generalize well to the test dataset.



Solutions to the problem of overfitting are specific to particular machine learning approaches. For now, the important point is to be aware of the issue. How well the models are able to handle noisy data is an important source of distinction among them.

## Machine learning in practice

So far, we've focused on how machine learning works in theory. To apply the learning process to real-world tasks, we'll use a five-step process. Regardless of the task at hand, any machine learning algorithm can be deployed by following these steps:

1. **Data collection:** The data collection step involves gathering the learning material an algorithm will use to generate actionable knowledge. In most cases, the data will need to be combined into a single source like a text file, spreadsheet, or database.
2. **Data exploration and preparation:** The quality of any machine learning project is based largely on the quality of its input data. Thus, it is important to learn more about the data and its nuances during a practice called data exploration. Additional work is required to prepare the data for the learning process. This involves fixing or cleaning so-called "messy" data, eliminating unnecessary data, and recoding the data to conform to the learner's expected inputs.
3. **Model training:** By the time the data has been prepared for analysis, you are likely to have a sense of what you are capable of learning from the data. The specific machine learning task chosen will inform the selection of an appropriate algorithm, and the algorithm will represent the data in the form of a model.
4. **Model evaluation:** Because each machine learning model results in a biased solution to the learning problem, it is important to evaluate how well the algorithm learns from its experience. Depending on the type of model used, you might be able to evaluate the accuracy of the model using a test dataset or you may need to develop measures of performance specific to the intended application.
5. **Model improvement:** If better performance is needed, it becomes necessary to utilize more advanced strategies to augment the performance of the model. Sometimes, it may be necessary to switch to a different type of model altogether. You may need to supplement your data with additional data or perform additional preparatory work as in step two of this process.

After these steps are completed, if the model appears to be performing well, it can be deployed for its intended task. As the case may be, you might utilize your model to provide score data for predictions (possibly in real time), for projections of financial data, to generate useful insight for marketing or research, or to automate tasks such as mail delivery or flying aircraft. The successes and failures of the deployed model might even provide additional data to train your next generation learner.

## Types of input data

The practice of machine learning involves matching the characteristics of input data to the biases of the available approaches. Thus, before applying machine learning to real-world problems, it is important to understand the terminology that distinguishes among input datasets.

The phrase **unit of observation** is used to describe the smallest entity with measured properties of interest for a study. Commonly, the unit of observation is in the form of persons, objects or things, transactions, time points, geographic regions, or measurements. Sometimes, units of observation are combined to form units such as person-years, which denote cases where the same person is tracked over multiple years; each person-year comprises of a person's data for one year.



The unit of observation is related, but not identical, to the **unit of analysis**, which is the smallest unit from which the inference is made. Although it is often the case, the observed and analyzed units are not always the same. For example, data observed from people might be used to analyze trends across countries.

Datasets that store the units of observation and their properties can be imagined as collections of data consisting of:

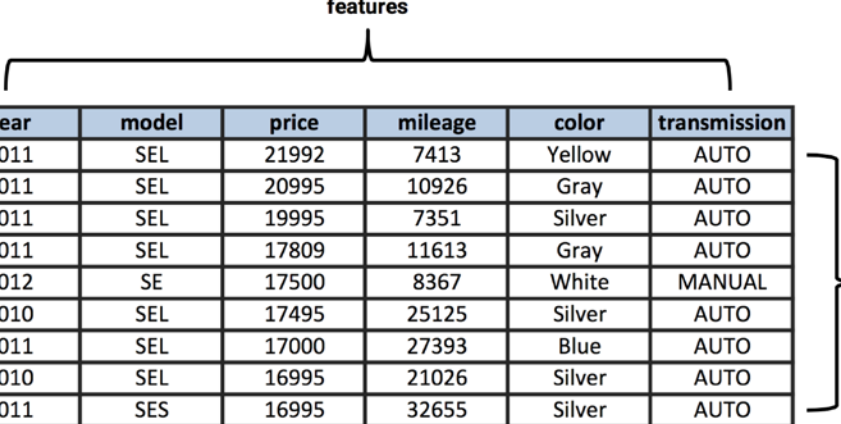
- **Examples:** Instances of the unit of observation for which properties have been recorded
- **Features:** Recorded properties or attributes of examples that may be useful for learning

It is easiest to understand features and examples through real-world cases. To build a learning algorithm to identify spam e-mail, the unit of observation could be e-mail messages, the examples would be specific messages, and the features might consist of the words used in the messages. For a cancer detection algorithm, the unit of observation could be patients, the examples might include a random sample of cancer patients, and the features may be the genomic markers from biopsied cells as well as the characteristics of patient such as weight, height, or blood pressure.



While examples and features do not have to be collected in any specific form, they are commonly gathered in **matrix format**, which means that each example has exactly the same features.

The following spreadsheet shows a dataset in matrix format. In matrix data, each row in the spreadsheet is an example and each column is a feature. Here, the rows indicate examples of automobiles, while the columns record various each automobile's features, such as price, mileage, color, and transmission type. Matrix format data is by far the most common form used in machine learning. Though, as you will see in the later chapters, other forms are used occasionally in specialized cases:



year	model	price	mileage	color	transmission
2011	SEL	21992	7413	Yellow	AUTO
2011	SEL	20995	10926	Gray	AUTO
2011	SEL	19995	7351	Silver	AUTO
2011	SEL	17809	11613	Gray	AUTO
2012	SE	17500	8367	White	MANUAL
2010	SEL	17495	25125	Silver	AUTO
2011	SEL	17000	27393	Blue	AUTO
2010	SEL	16995	21026	Silver	AUTO
2011	SES	16995	32655	Silver	AUTO

Features also come in various forms. If a feature represents a characteristic measured in numbers, it is unsurprisingly called **numeric**. Alternatively, if a feature is an attribute that consists of a set of categories, the feature is called **categorical** or **nominal**. A special case of categorical variables is called **ordinal**, which designates a nominal variable with categories falling in an ordered list. Some examples of ordinal variables include clothing sizes such as small, medium, and large; or a measurement of customer satisfaction on a scale from "not at all happy" to "very happy." It is important to consider what the features represent, as the type and number of features in your dataset will assist in determining an appropriate machine learning algorithm for your task.

## Types of machine learning algorithms

Machine learning algorithms are divided into categories according to their purpose. Understanding the categories of learning algorithms is an essential first step towards using data to drive the desired action.

A **predictive model** is used for tasks that involve, as the name implies, the prediction of one value using other values in the dataset. The learning algorithm attempts to discover and model the relationship between the **target** feature (the feature being predicted) and the other features. Despite the common use of the word "prediction" to imply forecasting, predictive models need not necessarily foresee events in the future. For instance, a predictive model could be used to predict past events, such as the date of a baby's conception using the mother's present-day hormone levels. Predictive models can also be used in real time to control traffic lights during rush hours.


Because predictive models are given clear instruction on what they need to learn and how they are intended to learn it, the process of training a predictive model is known as **supervised learning**. The supervision does not refer to human involvement, but rather to the fact that the target values provide a way for the learner to know how well it has learned the desired task. Stated more formally, given a set of data, a supervised learning algorithm attempts to optimize a function (the model) to find the combination of feature values that result in the target output.

The often used supervised machine learning task of predicting which category an example belongs to is known as **classification**. It is easy to think of potential uses for a classifier. For instance, you could predict whether:

- An e-mail message is spam
- A person has cancer
- A football team will win or lose
- An applicant will default on a loan


In classification, the target feature to be predicted is a categorical feature known as the **class**, and is divided into categories called **levels**. A class can have two or more levels, and the levels may or may not be ordinal. Because classification is so widely used in machine learning, there are many types of classification algorithms, with strengths and weaknesses suited for different types of input data. We will see examples of these later in this chapter and throughout this book.

Supervised learners can also be used to predict numeric data such as income, laboratory values, test scores, or counts of items. To predict such numeric values, a common form of **numeric prediction** fits linear regression models to the input data. Although regression models are not the only type of numeric models, they are, by far, the most widely used. Regression methods are widely used for forecasting, as they quantify in exact terms the association between inputs and the target, including both, the magnitude and uncertainty of the relationship.

 Since it is easy to convert numbers into categories (for example, ages 13 to 19 are teenagers) and categories into numbers (for example, assign 1 to all males, 0 to all females), the boundary between classification models and numeric prediction models is not necessarily firm.

A **descriptive model** is used for tasks that would benefit from the insight gained from summarizing data in new and interesting ways. As opposed to predictive models that predict a target of interest, in a descriptive model, no single feature is more important than any other. In fact, because there is no target to learn, the process of training a descriptive model is called **unsupervised learning**. Although it can be more difficult to think of applications for descriptive models – after all, what good is a learner that isn't learning anything in particular – they are used quite regularly for data mining.

For example, the descriptive modeling task called **pattern discovery** is used to identify useful associations within data. Pattern discovery is often used for **market basket analysis** on retailers' transactional purchase data. Here, the goal is to identify items that are frequently purchased together, such that the learned information can be used to refine marketing tactics. For instance, if a retailer learns that swimming trunks are commonly purchased at the same time as sunglasses, the retailer might reposition the items more closely in the store or run a promotion to "up-sell" customers on associated items.

 Originally used only in retail contexts, pattern discovery is now starting to be used in quite innovative ways. For instance, it can be used to detect patterns of fraudulent behavior, screen for genetic defects, or identify hot spots for criminal activity.

The descriptive modeling task of dividing a dataset into homogeneous groups is called **clustering**. This is sometimes used for **segmentation analysis** that identifies groups of individuals with similar behavior or demographic information, so that advertising campaigns could be tailored for particular audiences. Although the machine is capable of identifying the clusters, human intervention is required to interpret them. For example, given five different clusters of shoppers at a grocery store, the marketing team will need to understand the differences among the groups in order to create a promotion that best suits each group.

Lastly, a class of machine learning algorithms known as **meta-learners** is not tied to a specific learning task, but is rather focused on learning how to learn more effectively. A meta-learning algorithm uses the result of some learnings to inform additional learning. This can be beneficial for very challenging problems or when a predictive algorithm's performance needs to be as accurate as possible.

## Matching input data to algorithms

The following table lists the general types of machine learning algorithms covered in this book. Although this covers only a fraction of the entire set of machine learning algorithms, learning these methods will provide a sufficient foundation to make sense of any other method you may encounter in the future.

Model	Learning task	Chapter
<b>Supervised Learning Algorithms</b>		
Nearest Neighbor	Classification	3
Naive Bayes	Classification	4
Decision Trees	Classification	5
Classification Rule Learners	Classification	5
Linear Regression	Numeric prediction	6
Regression Trees	Numeric prediction	6
Model Trees	Numeric prediction	6
Neural Networks	Dual use	7
Support Vector Machines	Dual use	7
<b>Unsupervised Learning Algorithms</b>		
Association Rules	Pattern detection	8
k-means clustering	Clustering	9
<b>Meta-Learning Algorithms</b>		
Bagging	Dual use	11
Boosting	Dual use	11
Random Forests	Dual use	11

To begin applying machine learning to a real-world project, you will need to determine which of the four learning tasks your project represents: classification, numeric prediction, pattern detection, or clustering. The task will drive the choice of algorithm. For instance, if you are undertaking pattern detection, you are likely to employ association rules. Similarly, a clustering problem will likely utilize the k-means algorithm, and numeric prediction will utilize regression analysis or regression trees.

For classification, more thought is needed to match a learning problem to an appropriate classifier. In these cases, it is helpful to consider various distinctions among algorithms – distinctions that will only be apparent by studying each of the classifiers in depth. For instance, within classification problems, decision trees result in models that are readily understood, while the models of neural networks are notoriously difficult to interpret. If you were designing a credit-scoring model, this could be an important distinction because law often requires that the applicant must be notified about the reasons he or she was rejected for the loan. Even if the neural network is better at predicting loan defaults, if its predictions cannot be explained, then it is useless for this application.

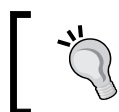
To assist with the algorithm selection, in every chapter, the key strengths and weaknesses of each learning algorithm are listed. Although you will sometimes find that these characteristics exclude certain models from consideration, in many cases, the choice of algorithm is arbitrary. When this is true, feel free to use whichever algorithm you are most comfortable with. Other times, when predictive accuracy is primary, you may need to test several algorithms and choose the one that fits the best, or use a meta-learning algorithm that combines several different learners to utilize the strengths of each.

## **Machine learning with R**

Many of the algorithms needed for machine learning with R are not included as part of the base installation. Instead, the algorithms needed for machine learning are available via a large community of experts who have shared their work freely. These must be installed on top of base R manually. Thanks to R's status as free open source software, there is no additional charge for this functionality.

A collection of R functions that can be shared among users is called a **package**. Free packages exist for each of the machine learning algorithms covered in this book. In fact, this book only covers a small portion of all of R's machine learning packages.

If you are interested in the breadth of R packages, you can view a list at **Comprehensive R Archive Network (CRAN)**, a collection of web and FTP sites located around the world to provide the most up-to-date versions of R software and packages. If you obtained the R software via download, it was most likely from CRAN at <http://cran.r-project.org/index.html>.



If you do not already have R, the CRAN website also provides installation instructions and information on where to find help if you have trouble.

The **Packages** link on the left side of the page will take you to a page where you can browse packages in an alphabetical order or sorted by the publication date. At the time of writing this, a total 6,779 packages were available—a jump of over 60% in the time since the first edition was written, and this trend shows no sign of slowing!

The **Task Views** link on the left side of the CRAN page provides a curated list of packages as per the subject area. The task view for machine learning, which lists the packages covered in this book (and many more), is available at <http://cran.r-project.org/web/views/MachineLearning.html>.

## Installing R packages

Despite the vast set of available R add-ons, the package format makes installation and use a virtually effortless process. To demonstrate the use of packages, we will install and load the `RWeka` package, which was developed by Kurt Hornik, Christian Buchta, and Achim Zeileis (see *Open-Source Machine Learning: R Meets Weka* in *Computational Statistics* 24: 225-232 for more information). The `RWeka` package provides a collection of functions that give R access to the machine learning algorithms in the Java-based Weka software package by Ian H. Witten and Eibe Frank. More information on Weka is available at <http://www.cs.waikato.ac.nz/~ml/weka/>.



To use the `RWeka` package, you will need to have Java installed (many computers come with Java preinstalled). Java is a set of programming tools available for free, which allow for the use of cross-platform applications such as Weka. For more information, and to download Java on your system, you can visit <http://java.com>.