

机器学习与数据科学 (基于R的统计学习方法)

[美] Daniel D • Gutierrez 著 施翊 译



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

目 录

[版权信息](#)

[版权声明](#)

[内容提要](#)

[前言](#)

[第1章 机器学习综述](#)

[1.1 机器学习的分类](#)

[1.2 机器学习的实际案例](#)

[1.2.1 预测回头客挑战赛](#)

[1.2.2 Netflix公司](#)

[1.2.3 算法交易挑战赛](#)

[1.2.4 Heritage健康奖](#)

[1.3 机器学习的过程](#)

[1.4 机器学习背后的数学](#)

[1.5 成为一名数据科学家](#)

[1.6 统计计算的R工程](#)

[1.7 RStudio](#)

[1.8 使用R包](#)

[1.9 数据集](#)

[1.10 在生产中使用R](#)

[1.11 小结](#)

[第2章 连接数据](#)

[2.1 管理你的工作目录](#)

[2.2 数据文件的种类](#)

[2.3 数据的来源](#)

[2.4 从网络中下载数据集](#)

[2.5 读取CSV文件](#)

[2.6 读取Excel文件](#)

[2.7 使用文件连接](#)
[2.8 读取JSON文件](#)
[2.9 从网站中抓取数据](#)
[2.10 SQL数据库](#)
[2.11 R中的SQL等价表述](#)
[2.12 读取Twitter数据](#)
[2.13 从谷歌分析中读取数据](#)
[2.14 写数据](#)
[2.15 小结](#)

[第3章 数据处理](#)

[3.1 特征工程](#)
[3.2 数据管道](#)
[3.3 数据采样](#)
[3.4 修正变量名](#)
[3.5 创建新变量](#)
[3.6 数值离散化](#)
[3.7 日期处理](#)
[3.8 将类变量二值化](#)
[3.9 合并数据集](#)
[3.10 排列数据集](#)
[3.11 重塑数据集](#)
[3.12 使用dplyr进行数据操作](#)
[3.13 处理缺失数据](#)
[3.14 特征缩放](#)
[3.15 降维](#)
[3.16 小结](#)

[第4章 探索性数据分析](#)

[4.1 数据统计](#)
[4.2 探索性可视化](#)
[4.3 直方图](#)
[4.4 箱形图](#)
[4.5 条形图](#)
[4.6 密度图](#)
[4.7 散点图](#)

[4.8 QQ图](#)

[4.9 热图](#)

[4.10 缺失值的图表](#)

[4.11 解释性图表](#)

[4.12 小结](#)

[第5章 回归](#)

[5.1 一元线性回归](#)

[5.2 多元线性回归](#)

[5.3 多项式回归](#)

[5.4 小结](#)

[第6章 分类](#)

[6.1 一个简单的例子](#)

[6.2 逻辑回归](#)

[6.3 分类树](#)

[6.4 朴素贝叶斯](#)

[6.5 K-最近邻](#)

[6.6 支持向量机](#)

[6.7 神经网络](#)

[6.8 集成](#)

[6.9 随机森林](#)

[6.10 梯度提升机](#)

[6.11 小结](#)

[第7章 评估模型性能](#)

[7.1 过拟合](#)

[7.2 偏差和方差](#)

[7.3 干扰因子](#)

[7.4 数据泄漏](#)

[7.5 测定回归性能](#)

[7.6 测定分类性能](#)

[7.7 交叉验证](#)

[7.8 其他机器学习诊断法](#)

[7.8.1 获取更多的训练观测数据](#)

[7.8.2 特征降维](#)

[7.8.3 添加新特征](#)

[7.8.4 添加多项式特征](#)

[7.8.5 对正则化参数进行微调](#)

[7.9 小结](#)

[第8章 非监督学习](#)

[8.1 聚类](#)

[8.2 模拟聚类](#)

[8.3 分级聚类](#)

[8.4 K-均值聚类](#)

[8.5 主成分分析](#)

[8.6 小结](#)

[术语表](#)

[欢迎来到异步社区！](#)

版权信息

书名：机器学习与数据科学（基于R的统计学习方法）

ISBN：978-7-115-45240-5

本书由人民邮电出版社发行数字版。版权所有，侵权必究。

您购买的人民邮电出版社电子书仅供您个人使用，未经授权，不得以任何方式复制和传播本书内容。

我们愿意相信读者具有这样的良知和觉悟，与我们共同保护知识产权。

如果购买者有侵权行为，我们可能对该用户实施包括但不限于关闭该帐号等维权措施，并可能追究法律责任。

• 著 [美] Daniel D. Gutierrez

译 施 翳

责任编辑 陈冀康

• 人民邮电出版社出版发行 北京市丰台区成寿寺路11号

邮编 100164 电子邮件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

• 读者服务热线：(010)81055410

反盗版热线: (010)81055315

版权声明

Simplified Chinese translation copyright ©2017 by Posts and Telecommunications Press

ALL RIGHTS RESERVED

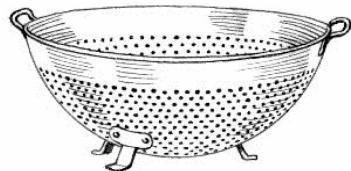
Machine Learning and Data Science, an Introduction to Statistical Learning Methods with R, by Daniel D. Gutierrez ISBN 9781634620963

Copyright © 2016 by Technics Publications, LLC

本书中文简体版由Technics Publications授权人民邮电出版社出版。
未经出版者书面许可，对本书的任何部分不得以任何方式或任何手段复制和传播。

版权所有，侵权必究。

内容提要

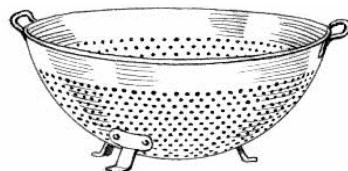


当前，机器学习和数据科学都是很重要和热门的相关学科，需要深入地研究学习才能精通。

本书试图指导读者掌握如何完成涉及机器学习的数据科学项目。本书将为数据科学家提供一些在统计学习领域会用到的工具和技巧，涉及数据连接、数据处理、探索性数据分析、监督机器学习、非监督机器学习和模型评估。本书选用的是R统计环境，书中所有代码示例都是用R语言编写的，涉及众多流行的R包和数据集。

本书适合数据科学家、数据分析师、软件开发者以及需要了解数据科学和机器学习方法的科研人员阅读参考。

前言



在我的童年时代，我十分喜爱著名科幻作家、教授艾萨克·阿西莫夫（Asimov Isaac）的《基地三部曲》。故事的主角叫作Hari Seldon，他是一位开创了“心理历史学”的数学教授，这门学科涉及历史学、社会学和数理统计，可以用来预测未来事件发生的概率。因此，我从小就迷上了预测这一概念。很自然地，我长大后成为了一名数据科学家。我把机器学习类比为Seldon的素数辐射法（Prime Radiant），是一个存储“心理历史学方程”的工具，可以用来展示人类未来发展前景。

远在“数据科学”这一概念问世之前，我就已经成为（或者假装成为）一名数据科学家（data scientist）很多年了。“数据科学家”这一头衔经历了数十年职业演化才建立，对此我表示十分欣喜。最近在业内论坛上，人们针对“数据科学”是否能恰当描述这一领域展开了激烈的辩论。我认为这个术语确实做出了了不起的贡献，因为数据科学家所做的事情实际上大多都是实验，这对我日复一日的基础工作毫无疑问是有效的科学方法（scientific method）。我个人认为“数据科学”比“数据挖掘”（data mining）或者“商业智能”（business intelligence）更精确、描述得更好。随着时间的流逝，后两种表述经历了严格的技术成熟度曲线。我对“数据科学家”这一称呼很满意，因为我真切地感受到自己是一个用数据进行实验的科学家。

这里讲讲我是如何理解数据科学用科学方法来解决问题的。

- 提出一个问题：问题可以是针对一个具体观察结果的解释。例如，

是不是给目标顾客打的电话越多，销售团队能成交的单量也越多？这一阶段牵涉到寻找能够为解决问题提供线索的数据集。当在数据科学中使用科学研究方法时，确定一个好的问题相当不易，并且问题的好坏会直接影响到研究的最终结果。

- 作出一个假设（**hypothesis**）：假设就是一个可能解释观测结果的猜想。这个假设是在提出问题时，基于现有的知识做出的。一个典型的假设表述形式是：是否批准一个房屋贷款，决定因素可能是房主的收入水平预期和信用评分。
- 预测（**prediction**）：这一步骤牵涉到确定假设的逻辑结论，使用数据科学意味着选择一个合适的机器学习算法来解决这个问题。在理想状态下，预测必须把假设和其他的可能原因区分出来；如果两个假设做出了一样的预测，观测到预料中的结果就并不能说明其中一个假设是正确的。这就是为什么某些领域的机器学习需要用相同数据集、不同算法来做实验，来看最终的结果如何。这一步也需要用有限数据集来“训练”算法。
- 测试（**testing**）：这一步骤是考察实际结果是否像假设预测的那样。作为一名数据科学家，你需要在训练过程中保留一份数据集，来评估预测的准确性。这一实验的目的是判断基于真实世界的观测与基于假设的预测是否一致。如果一致，该假设的置信度提高；否则，置信度降低。然而，一致性并不能确保假说的正确，更深入的实验可能会揭示其他问题。
- 分析（**analysis**）：这一阶段需要确定实验得出的结论是什么，并决定下一步需要做什么。通过数据可视化，你可能发现之前在机器学习中使用的数据不足以预测得出你需要的结果。所以你回退到前面，重新审视提出问题那一步。你可能希望用不同的数据集重复实验，来观察是否能得到相同的结果。一旦一个假说得到了数据的强烈支持，可以在同一主题下提出一个新的问题，来寻求更深入的了解。在这种情况下，科学方法是一个迭代的过程，它不断重复，直至发展出一个能继续前进的强大“理论”。

机器学习是数据科学家用来做预测和检验假设有效性的基本工具集。让我们继续简要地了解一下机器学习是什么、数据科学家用它来做什么。“机器学习”这一表述代表了多学科的融合：计算机科学（computer science）、数理统计、概率论和数据可视化。在接下去的章节中，我们将会看到机器学习有两大基本类型：监督学习（supervised learning）用于预测，非监督学习（unsupervised learning）用于发现。如

果你真的想深入理解各种机器学习算法的奥秘，必须明白多个数学领域的原理，例如数理统计、概率论（probability theory）、计算学、线性代数（linear algebra）、偏微分方程（partial differential equations）和组合数学（combinatorics）。好在，在本书中我们使用了R语言，所以无需钻研算法的基本原理。我们只需要学习如何使用它们。

本书是如何编排的

本书希望能带领读者走进一个涉及机器学习的数据科学项目。并不是说我在提供的是学习机器学习的唯一方法，而是我认为这是数据科学家工作的典型方式。这一方法多年来对我十分受用，我希望通过本书把我的经验传授给大家。以下是本书的分章介绍。

- 第1章：机器学习综述。这一章包含数据科学概论和企业对这一领域日益关注的原因。我们也会对机器学习做个简要介绍，包括它是如何在数据科学中扮演一个不可或缺的角色的。然后我们将回顾机器学习的不同类型，每种类型都提供示例，并提取机器学习过程的大纲。最后，我们将讨论在实验性机器学习中，R环境如何通过使用众多的R包（R package）发挥重要的作用。
- 第2章：连接数据。机器学习的第一步是连接到一个合适的数据集，在R环境下得到数据内容，然后开始对其进行分析。在这一章中，我们使用R来连接数据，使用不同数据源（逗号分割文件格式CSV、Excel、JSON、Twitter和谷歌分析）用多种方式连接。我们也会铺设一条在SQL数据库中连接数据的通路。一旦数据连接到R环境中，我们就能开始学习如何搭建一个用于数据分析和机器学习的开发环境了。
- 第3章：数据处理。在开始一个机器学习项目的初期，一个冗长乏味但又不可或缺的步骤是“数据处理”，也称为“数据清洗”或是“数据转化”。换句话说，检查并精炼数据集以便进行更深入的分析。在这一章中，我们将着眼于创造一个数据处理工具箱，其中包括多种技术：(修正变量名、创造新变量、数值离散化、日期处理、变量二分法、合并\按顺序排列\重塑数据集、使用dplyr进行数据整理以及处理缺漏数据和特征缩放。其他主题包括特征工程、数据采样和数据管道)。最后，我们会一起学习主成分分析是如何做到有效降维的。
- 第4章：探索性数据分析。一旦数据整理成合适的格式，下一步要做的就是熟悉数据，以便想出如何在机器学习中使用它们。在这一章中，我们会使用探索性和解释性数据可视化来理解数据的属性，寻找数据的特征，推荐建模策略。我们会从使用R的统计功能开

始，包括数字摘要、因子变量水平、平均数\中位数\众数、分位数、标准差和变化率。我们也会使用R的绘图功能：直方图、箱线图、条形图、密度图、散点图、分位数图和热图。

- 第5章：回归。在本章中，我们将介绍机器学习最常见的形式：监督学习。我们会仔细检视用于预测分析的主力工具：线性回归。也会学习如何在R环境下建立一个线性模型，并计算出一条用于预测的回归线。单变量和多变量回归以及多项式回归都会在本章中进行演示。
- 第6章：分类。在本章中，我们会介绍监督学习的另一种常见形式：分类。我们将使用大量有用的R包来考察各种分类算法，包括逻辑回归、分类树、朴素贝叶斯分类器、K最近邻、支持向量机和神经网络。本章也会考虑集成方法，例如流行的随机森林算法。最后，我们会学习梯度提升机，它在机器学习比赛中十分流行。
- 第7章：评估模型性能。本章会讨论如何挑选模型，并且评估它的预测水平。我们还会讨论统计学习中影响表现的方面，比如过度拟合、偏差和方差的平衡、混杂因素和数据泄漏。同时，定义了衡量回归和分类模型准确度的标准。最后，我们将展示使模型泛化误差达到最小的交叉检验过程。
- 第8章：非监督学习。本章将会介绍使用两种聚类技术的非监督机器学习：分级聚类和K-均值聚类。在分级聚类算法的帮助下，用聚合法得到一个树状图或树形结构图，来展示元素之间的关系。然后运用K-均值聚类，使用迭代分割法来估计聚类的中心，并把每个数据点分配到聚类节点中。最后，我们会快速地看一下另一个流行的非监督工具——主成分分析。

在介绍机器学习的过程中，为了让读者的学习过程尽可能简单和直截了当，我确定了几个基本原则。

- 我不会在代码示例中使用复杂的（或者容易混淆的）R编程技术。当然，使用嵌入式的函数调用，一行语句就能解决一个程序问题，但是理解编程语句将会与我们的学习目的背道而驰，特别如果这是你第一次接触到R，所以本书让一切都将保持简单。
- 在本书中，我将尽量不用到流行的ggplot2图形包。作为替代，我们将选择使用基础的R的图形函数。毫无疑问，使用基础的R函数会更加直截了当。

- 我们会努力将R包和数据集的数目降到最低，针对每章的主题都会专注于最常用的程序包，加上一些能让过程更简便的支持包。

本书的目标读者

本书的目标受众相当广泛。如果你是一名分析师，不论在私人企业还是公众部门，需要通过从一些工具（如Excel）中得到的特征集来扩展你的分析技巧，那么这本书适合你；如果你是一位软件开发者，需要在代码中实现机器学习，那么这本书适合你；如果你是一名学术科研人员，需要了解数据科学和机器学习方法的最新进展，那么这本书适合你。这些细分读者的共同点是：诚心诚意地想要学习这一领域基础知识，并想快速地做出一些成绩。我希望各行各业的读者都能在本书中有所收获，因此书里使用的案例涉及各个领域。

我假设你已经了解了R程序设计，或者通过本书给出的一些材料能快速的学会它。我们不教授R语言，而是把R作为一个快速上手机器学习的工具。好消息是本书只使用了一些很基本的R语言；坏消息是，众所周知，R语言对初学者来说十分晦涩难懂。书中使用的大部分R代码脚本十分直白，在有必要的情况下，我会在代码中添加注释来解释。我不会浪费时间用复杂棘手的代码来介绍机器学习的概念。我希望你有足够的动力来面对快速了解机器学习这一挑战。本书会提供学习的大纲，同时下面也会给出很多附加的学习资源来帮助你完成这一过程。

你需要什么

本书不需要任何其他附加的硬件或者软件，很显然，你需要R统计编程环境。好在，它是开源的，可以免费使用。你可以通过访问www.r-project.org来获取R软件。它可以在各种UNIX平台、Windows和MacOS环境下安装运行。当你在访问www.r-project.org网站时，请尽可能利用上面所有的学习资源，包括R手册、R期刊、图书和其他关于R的文档。

在学习本书的过程中，另一个强烈推荐使用的软件是RStudio集成开发环境（IDE）。访问www.rstudio.com来下载RStudio。RStudio是一个功能强大的R用户界面，免费开源，并且在Windows、Mac和Linux上都有很好的表现。在编写这本书的过程中，我频繁使用了RStudio，也推荐你这样做。虽然你可以使用R自带的基本编程环境来工作，但是RStudio包含了很多对程序员来说很有吸引力的特色：

- 语法高亮显示，代码补全和智能排版；
- 工作窗口和数据查看器；
- 历史曲线，缩放，灵活的图片、pdf导出；
- 集成的R帮助和文档；
- 可搜索的命令历史；
- 直接从源编辑器执行R代码；
- 便于管理在用项目的多个工作目录。

在学习本书的过程中，你也需要一些额外的R包（拓展R的统计环境）。这些R包也是开源的，并且能在R内部进行下载和安装。当具体案例出现时，我会指导你如何下载、使用R包。

同时，我也有意地避免读者去寻找、下载和安装本书案例中用到的数据集。在大多数情况下，我尽量使用R自带的数据集；在一些情况下，我们可能使用特定的R包带有的数据集；在少数其他情况下，我使用了R之外的数据集，但是我会指导你如何连接这些数据集。

R代码和图表

你会发现本书包含了很多R编程代码的示例，以及使用特定命令后R环境所返回的结果。为了在本书中展示代码，我将把熟悉的“>”符号放在R控制台输入的所有命令之前。我们也会使用一种特殊的“代码字体”（courier new）以便和本书正文区分。此外，R的输出也会用同样的代码字体显示，但是前面没有“>”标识。在阅读本书的时候，我鼓励你们自行在R环境中输入所有的代码示例，以便获得使用这一环境的语感。改变不同的元素来实验每个代码示例，观察输出的不同结果，这同样也是一个不错的学习方法。

为了让你的学习经历尽可能愉快，我在出版社网站
 (<https://technicspub.com/Analytics/>) 上传了本书中使用的所有R源代码和注释。同时也收录了所有的图表（很多是彩色的），它们在正文中是以灰度图的形式存在的，这会使一些图表更易于理解。日后你可以访问这一资源库得到最新的代码。

超越本书

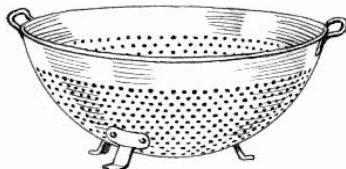
一旦你读完了本书，你将需要一些指导，关于如何更进一步学习机器学习。好在，机器学习在过去的几年里逐渐演化完整，该领域正日益受到关注。因此，有很多资源能帮助你扩充知识面。学无止境，你只需要考虑自己想钻研到多深入。为了帮助你开始学习，这里有一个简短的资源列表。

- 免费的在线课程：在大规模在线开放课程（MOOC，massive open online course）时代，你可以找到大量关于机器学习和相关领域的优质免费课程。我最喜欢的两个MOOC平台是Coursera (www.coursera.org) 和edX (www.edx.org)。相比之下，我更偏爱Coursera的产品，因为我试学过上面很多关于机器学习的课程，并且作为社区助教工作过一段时间。
- 博客：很多优秀的博主提供的优质文章能帮助你更好地在这个领域学习。好在，有一个特别好的网站聚合了很多流行的R博客（r-bloggers）的内容：www.r-bloggers.com。常常造访这个网站或者订阅它的每日邮件推送，你就能紧跟这个快速发展的领域的潮流了。
- **Meetup群组：**寻找你们当地聚焦于数据科学的**Meetup群组**，并选择其中一个或者多个你感兴趣的群组加入，是一种学习的好方法。在我的家乡洛杉矶，就有许多优秀的机器学习群组和R群组。
- **Twitter信息流：**就我自己而言，我从Twitter关注的人那里学习到了很多知识。一个人不可能监控整个业界的动向，所以我的Twitter朋友提供了非常有价值的服务。他们提醒我业界的新动向、文章、研究方法、产品、服务项目、会议等等。马上开始，搜索最相关的标签：#MachineLearning、#DataScience、#R和#BigData来找到一些你喜爱的用户。

联系作者

你可以通过多种方式找到我。访问我的咨询公司网站
www.amuletanalytics.com或者在LinkedIn上都可以。然而可能最容易
找到我的地方是在Twitter上（@AMULETAnalytics）。在那里你可以订
阅我，我会发布关于数据科学、机器学习和大数据的一些感想。

第1章 机器学习综述



机器学习（Machine Learning）可以看成是从观察自然界来推断结果和提取灵感的一套工具和方法。举个生活中常见的例子，你想通过房间数、卫生间数、建筑面积和地皮尺寸来预测一栋房子的价格，可以使用一个简单的机器学习算法（例如，线性回归），从现有的真实房地产销售数据集中学习，通过机器学习数据集中每栋房子的售价，可以预测尚未出售的房屋售价。事实上，这种预测需要海量数据（通常规模在 TB以上）的支撑。同时，数据的质量对预测结果准确度起着十分重要的作用，就像数据科学界的一句耳熟能详的话说的那样：好算法不如大数据。

近年来，机器学习已经发展成一门很成熟的学科。它逐渐成为数据科学领域的促进者，反过来，数据科学的发展也带动了大数据（Big Data）的发展。然而，机器学习并不是一门全新的学科，它的基本原理在相当长时间前就已经深入人心了，只是使用了不同的名称，例如，“数据挖掘”“在数据库中的知识挖掘（knowledge discovery）”和“商业智能”，这些术语都是机器学习的传统叫法。在此之前，“统计”和“数据分析”都用来描述从数据中收集信息的过程。我相信机器学习是现在描述这个领域的最好术语。Machine Learning也因为大量引用而成为Twitter圈中的热门标签。考虑到通过数据对系统进行的建设和研究，机器学习也被看成是人工智能（artificial intelligence）的一个分支。现如今，机器学习的应用大多依赖云存储硬件和性能优异的并行框架，如Apache公司的Hadoop和AMP实验室的Spark。

“机器学习”第一次正式使用是在1959年，当时在IBM公司工作的

Arthur Samuel把机器学习描述成赋予未设定程序的计算机学习能力。很快，到了1998年，卡耐基梅隆（Carnegie Mellon）大学机器学习系的系主任Tom Mitchell给学习程序下了一个定义：

如果一个计算机程序针对某类任务T的性能用P衡量，且根据经验E来自我完善，那么我们称这个计算机程序在从经验E中学习，针对某类任务T，它的性能用P来衡量。

Mitchell这个广为人知的定义适用范围非常广泛，能概括我们通常所说的大多数“学习”任务。在这一定义下，我们举一个机器学习问题的例子：考虑任务T是把垃圾邮件做分类，性能指标P是被正确分类的垃圾邮件的百分比，和训练集E是已经分好类（垃圾邮件或正常邮件）的邮件数据集。垃圾邮件分类器是机器学习解决现实商业问题的首批应用之一，如今它也应用在绝大部分邮件软件中。

启动一个新的机器学习项目时，另一条需要时刻谨记在心的公理是美国数学家John Tukey提出的，他因为在统计方式上的贡献和1977年开创性的著作《Exploratory Data Analysis》而受到统计学圈子的推崇：

拥有数据和对结果的渴求并不能确保从已知的数据中得到一个合理的结果。

这一准则意味着一个合格的机器学习从业者需要知道什么时候应该放弃，什么时候你拥有的数据不足以得出需要的答案。另一条耳熟能详的格言“输入无用数据，就会输出无用数据”同样也适用于机器学习领域。

1.1 机器学习的分类

本书会向读者介绍机器学习的基本原理。作为数据科学和大数据产业的主要推动力，机器学习在众多行业中广受关注，它可以为企业提供使公司数据资产增值的新方法。在本书中，我们会基于R语言统计环境学习机器学习算法的原理，包括两种基本类型：监督学习和非监督学习。

监督机器学习（Supervised machine learning）通常与预测有关，与每个观测值（也称为特征变量，feature variable）对应，都有一个结果值。监督学习的训练目标是根据响应模型准确预测未来的观测值对应的响应结果。很多传统的学习算法，诸如线性回归或逻辑回归，都属于监督学习的领域。

非监督机器学习（Unsupervised machine learning）是更开放性的一种类型。它不使用标记好的数据集，而是一套应用于程序上的统计工具，在大量的观测中只测量其中一组特征变量。在这种情况下，预测不再是学习的目标，因为数据集没有被标记，不存在可以监督分析行为的响应变量。事实上，非监督学习的目标是通过对特征变量的观测来挖掘一些有趣的事情。例如，你可以找到一个数据形象化的展现方式，或者发现数据集中隐藏的子群。

非监督学习技术的一个常用的场景是K-均值聚类，即在数据点集中找出“聚类”。另一种常用技术叫做主成分分析（PCA），用于降维，也就是说，在保持数据多样性的同时，减少特征变量，来简化学习算法中的数据复杂度，加快数据处理效率，并能降低所需的内存占用。

1.2 机器学习的实际案例

在本节内容中，我将向大家展示一些使用机器学习解决日常问题的例子。在学习这些案例之前，查看项目最初的需求，回顾数据集和每一个特征变量，并搞清楚如何判断解决方案是否成功，会让你的阅读事半功倍。阅读完本书之后，你甚至可以尝试自己的解决方案。为了做这些事情，我将特别推荐Kaggle (www.kaggle.com) 的数据挑战赛，它的金钱奖励已经吸引了全世界成千上万的数据科学家进行挑战。

在这些数据科学挑战中，参赛者为了寻找成功的解决方案，需要考虑以下问题。

1. 为谁解决问题？解决什么问题？
2. 现在这个问题是怎么解决的（如果已经有解决方案的话）？
3. 针对这个问题，可用的数据集是什么？这些数据集的来源是哪里？
4. 解决方案的结果如何展现（如商业智能仪表盘、在线应用中引入算法、一份静态管理报告等等）？
5. 这是什么类型的问题：为了降低收入流失（节流）还是增加收入（开源）？

在不同的比赛中，算法的评估方法也不尽相同。最常用的方法是将均方根误差（RMSE）的值降到最小，这一数值用于评价测试集的预测结果是否准确。RMSE评价法会在第7章进行更深入的解释。另一种常用的评估方法是AUC，即ROC曲线下的面积。

Kaggle挑战赛的一些赞助商也会参与比赛，有些为了获得产品或服务的新发展方向，有些为了招聘比赛中的赢家。对于你，一个上升期的数据科学家来说，这些挑战赛可以当做是磨炼技能的优秀训练场。

1.2.1 预测回头客挑战赛

品牌商为了吸引新客户购买商品，经常举办打折活动。在最初的激励性购买后，重复购买商品的回头客对品牌方来说是最有价值的。有足够的购物记录之后，可以预测在一次促销活动中，哪些顾客会再次购买商品。然而，在第一次购物之前就鉴别哪些顾客会成为忠诚的买主，这是一项更有挑战性的工作。

预测回头客挑战赛（Acquire Valued Shoppers Challenge）需要挑战者预测哪些顾客最可能成为回头客。为了增强算法的准确性，参赛的数据科学家得到了参与促销活动的大量顾客的完整购物记录，包括购买产品的类型和价格。比赛组织者也为顾客提供了初次购物和再次购物的激励。这个比赛的奖金是30000美元。

这个项目提供了超过30万名匿名顾客的将近3亿5千行交易记录数据，解压后达到22GB，大大超过了普通笔记本电脑的容量。在这一量级下，这个项目可以看做是“大数据”项目，所以，一个著名的供应商为参赛者提供了一个特别的工具。Revolution Analytics（Microsoft）公司为参赛者免费提供了亚马逊云服务（AWS，Amazon Web Services）下的R集成开发环境。有了AWS下的Linux机器的助力，内存容量可达到64GB，参赛者可以使用R集成开发环境下的并行外部存储算法。这个挑战赛是现实生活中数据项目的大小指数型增长带来的挑战的一个体现。

在图1-1的散点图中，展示了客户重复去商店的次数，x轴是商品的报价，数据点通过商场地理位置的不同来进行着色。

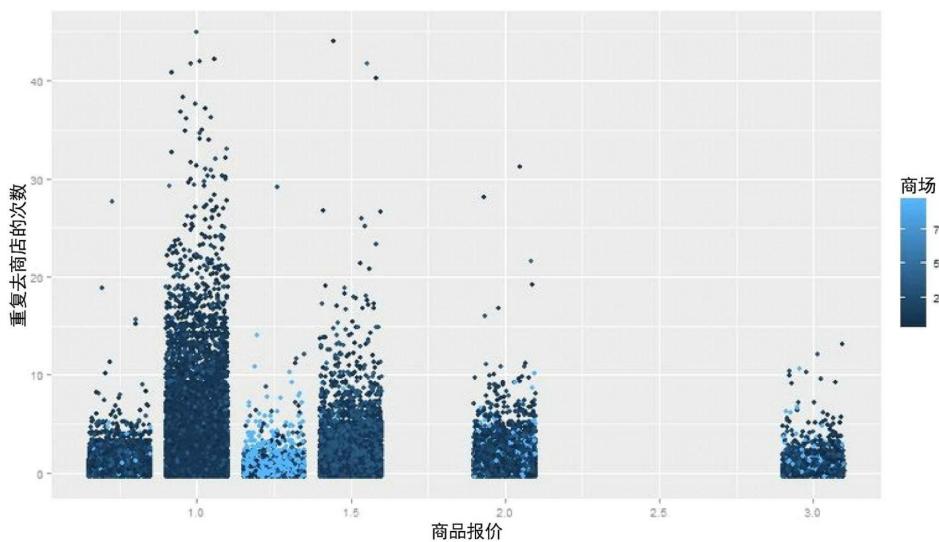


图1-1 使用预测回头客挑战赛数据绘制的探索图

1.2.2 Netflix公司

第一个著名的数据科学比赛可能是Netflix公司赞助的。这个著名的Netflix奖力图大幅度地提升预测的精确度：基于用户的观影偏好，预测他喜爱一部电影的程度。在2009年9月21日，Netflix公司给BellKor's Pragmatic Chaos队授予了100万美元的特等奖。BellKor队险胜The Ensemble队，他们的获胜作品在这个接近3年的比赛结束前的24分钟才提交。

用来解决Netflix奖的机器学习算法以均方根误差（RMSE, root mean square error）来衡量算法的好坏。均方根误差除了是一个众所周知的度量标准和数值之外，还有一个有用的性质：它可以放大极端误差带来的影响，无论是假阳性还是假阴性。这是理解任何推荐系统的重要性质。当然，简单的预测精度指标并不能概括在做推荐时很多其他重要的方面。举个例子，它不能预测推荐的顺序。不过，该队使用了特别的措施来降低均方根误差来赢得比赛。最终，获胜的解决方案将Netflix的推荐算法精确度提升了10%。

BellKor队的Netflix奖的解决方案涉及了很多机器学习的高新技术：协同滤波、用时间动力学进行矩阵分解、受限的玻尔兹曼机（RBM, Restricted Boltzmann Machines）、一个基于梯度提升决策树（GBDT, Gradient Boosted Decision Trees）的混合算法和一些用于特征工程的持久算法。这些方法超出了本书的讨论范围，但是通过看入围方案，你能学到很多。你可以在www.netflixprize.com下载介绍前4名入围方案的论文。

有趣的是，获得Netflix奖的解决方案事实上没有部署成功。因为在生产环境下，该算法的伸缩性不好。它无法及时处理庞大的Netflix数据集以获得结果。但是Netflix奖的无功而返让我们对一个可行的机器学习解决方案的组成要素有了更深刻的理解——虽然精确度显然是很重要的，但它必须在生产中衡量才有意义。最后，公司发现为了提升的精度将这一算法部署到生产环境中并不值得。

1.2.3 算法交易挑战赛

算法交易挑战赛（Algorithmic Trading Challenge）是一个预测比赛，致力于发掘新模型来预测股票市场在大额交易后的短期反应。参赛者被要求使用经验模型来预测在例如“流动性冲击”之后的买入和卖出行。通过提高模拟“回溯测试”的实时性，对市场弹性进行建模，旨在提高交易策略的评估方式。现在的方案都是基于市场没有弹性的假设。这一挑战赛的解决方案需要对模型中各种各样的市场动态有明确的认识。

这个58天的挑战赛，在2012年1月结束，吸引了111支队伍争夺10 000美元奖金。这个挑战赛是资本市场合作研究中心赞助的，这是一个驻澳大利亚的集科研人员、高校师生和工业界合作伙伴于一体的集团。

✓ 参赛者为了寻找最优解使用了各种各样的机器学习算法：线性回归、K-最近邻、支持向量机、随机森林、k-均值和各种组合算法。参赛算法使用均方根误差对预测的性能进行评估，计算在流动性冲击之后，每次买入和卖出报价。获胜的是在整个预测集中累计的均方根误差最低的一个算法。获奖者叫Ildefons Magrans，是一名来自西班牙的机器学习博士后研究员，拥有电子工程博士学位。他使用了R中的随机森林来降低均方根误差指标。

1.2.4 Heritage健康奖

这个机器学习难题可能是所有Kaggle挑战赛的“祖师爷”，因为参赛的队伍数目和获胜的奖金数目都十分惊人。HPN（Heritage Provider Network）赞助了该项目，旨在创造一个能帮助降低医疗服务成本的算法——显然这是全人类永恒的崇高追求。

根据美国医疗学会的最新调查显示，在美国，每年有超过7 100万人次住院。研究推断，仅在2006年一年中，就有超过300亿美元被花费在不必要的住院治疗中。有更好的方式吗？是否能够提早判断出高危患者，确保他们得到所需要的治疗？HPN相信这是可行的，并赞助了Kaggle挑战赛，以开发一个基于现有的患者数据来预测和减少不必要的住院治疗的机器学习算法。

参赛者被要求创造一个算法来预测患者在一年中会住院多久。一旦知道这个，医疗服务人员就能建立新的关怀计划，在紧急事件出现之前就对病人实施医疗帮助，从而减少非必要的住院时间。这将会提高患者

的健康水平，降低医疗服务费用。简言之，获胜方案将改变我们熟知的医疗保健服务，从救治病人转变为真正意义上的保持人类健康。

从2011年4月到2013年4月，这个比赛持续了两年。该比赛的第一名是一支名为POWERDOT的队伍，他们被授予了500 000美元的奖励。你可以在HPN的网站上找到这个比赛www.heritagehealthprize.com。在那里你可以下载到病人资料数据集以及一些具有重要意义的方案论文。最成功的解决方案涉及随机森林的集合，外加对特征变量的创造性选择。

除了上面介绍的Kaggle比赛的案例之外，下面还列举了许多能用机器学习解决的其他领域的问题（依然有很多没有提到的）。

1. 市场

- 预测生涯价值（LTV， Predict Lifetime Value）：预测高生涯价值的顾客特征。这将帮助进行客户细分，识别推销机会并为其他营销活动提供支持。
- 客户流失（churn）：判断流失的客户特征（例如，客户叛逃）。公司能够根据这一特征开发调整在线算法，使他们能应对流失的客户。
- 客户细分（customer segmentation）：通过让客户回答：因为什么原因会购买这件商品、因为什么原因不会购买这件商品等问题，使你能够定性地区分不同的客户群。
- 产品组合：产品优惠活动怎么组合能带来最低的客户流失率？例如，住宅和汽车的保险组合打折出售能有效降低客户的流失。
- 交叉销售（cross-selling）/提升销售（up-selling）额和推荐算法：给出一个顾客的浏览历史、购物历史和其他行为特征，预测他在未来可能想购买（或者升级）什么产品。
- 折扣目标：给出一个折扣能在多大程度上激发顾客的购买欲望？
- 重复购买（reactivation）的可能性：对于一个已经购买了商品的顾客，他重复购买的可能性有多高？
- 谷歌关键字（Google Adwords）优化和广告购买：决定不同的搜索关键字和广告位的最佳购买价格。

2. 销售

- 主导优先级（lead prioritization）：判断一个销售机会最后能成交的概率。
- 销售预测（sales forecasting）：提供销售策略，并深入了解销售预测的过程。

3. 供应链

- 需求预测（demand forecasting）：决定不同配送中心的最佳库存量，精益库存管理并防止缺货的情况发生。

4. 风险管理

- 欺诈（fraud）侦测：预测一个交易是否应该被锁定，因为它涉嫌某种欺诈，例如，信用卡欺诈或是医保欺诈。
- 应付账款回收：根据借款人和借款的特征，预测债务能回收的可能性。

5. 客户支持

- 呼叫中心管理（call center management）：呼叫中心的流量预测（例如，预测呼叫量以便合理配备接线人员）。
- 呼叫路由选择：根据呼叫者的呼叫历史、时刻、拥有的产品、客户流失风险、客户生涯价值等因素决定呼叫的等待时间。

6. 人力资源

- 人才管理：建立客观衡量员工绩效的指标。
- 人才流失：预测哪些员工最可能辞职。
- 简历筛选：根据过去的面试和聘用的结果为申请人的简历进行打分。
- 培训建议：根据雇员的绩效评估数据为其安排特定的培训计划。

7. 谷歌流感趋势

到现在为止，我们已经看了不少机器学习的成功应用和潜在应用，让我们简要地回顾一个失败的机器学习案例，并思考它为什么没有成功。在2013年2月，谷歌流感趋势（GFT，Google Flu Trends）上了头条——但这不是谷歌高管或者这个流感跟踪系统的创造者所希望的事。

《自然》杂志报道，GFT预测的感染者数量是美国疾控预防中心（Centers for Disease Control and Prevention, CDC）预测的数量的两倍多，而后者是基于全美实验室的检测报告来估算的。尽管GFT的创建目标是预测CDC的报告数据，这个令人大跌眼镜的情况还是发生了。鉴于GFT经常被认为是大数据应用的典范，我们可以从这个错误中吸取什么教训？

GFT试图根据人们在谷歌中使用的搜索关键字，如“我咳嗽了”，来预测某个时间地点的流感患者的比例。在这项研究的早期，预测的精度很好，很多人对GFT的未来十分乐观，因为使用谷歌的技术比CDC的方法划算多了。当时问题还没有显现，人们搜索的关键字会随着时间变化，此外，谷歌的搜索算法发生了改变，这些偏差影响了算法的表现并逐渐导致了预测精度跳水。

1.3 机器学习的过程

当在一个数据科学项目中工作时，总览机器学习的整个工作流程来决定应遵循的技术路线是十分有用的。在本节中，我会提供一个详细的工作流程，你可以在机器学习项目中使用它，它适用于几乎所有项目。如果你基于某些特定领域的需要，想对流程进行增改，请随意创造属于你的流程，以便未来重复使用。图1-2的简图描绘了一般情形下的工作流程。

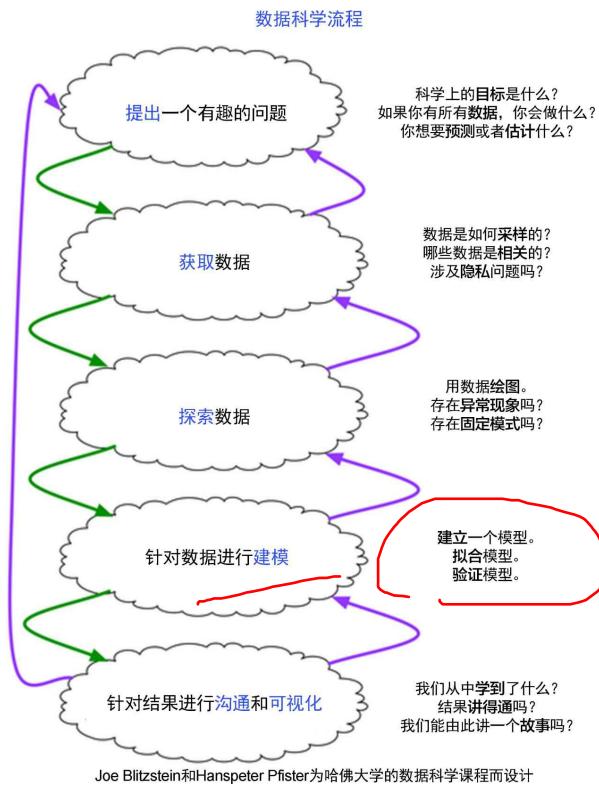


图1-2 机器学习项目中的数据科学流程

鉴于这是本书后续章节的模板，让我们一步步检查数据科学的流程。当你获得更多经验后，你会想要在此基础模型中增加更多细微的环节，但即使经过调整，流程在大体上还是一致的。

1. 理解问题。深入理解问题范畴，寻求该领域专家的帮助，如解释商业考量、提供数据集、确定关键变量等。最重要的是明确项目的目标。例如，要预测或是发现什么。任何机器学习的项目，如果没有清晰

的目标的话，将会后患无穷。如果项目的甲方只是简单地跟你说：“这里有一些数据，现在你可以用它来露一手了”。你可能会被吓跑，因为根据如此不现实的表述来提供结果是很困难的。相反，花时间执行流程的所有步骤，并且确保每一位项目相关人员都能参与进来，会使项目顺利进行。根据项目的不同，你可能需要与从事市场、金融、实业、IT甚至人力资源行业的人打交道。你将经常需要从多个部门获得资源。正确的接近数据科学的方法是，从一个对你的商业有着根深蒂固影响的问题开始，基于问题，反过来考虑数据分析和所需要的数据集。洞察力不会凭空产生，它们伴随着努力分析数据和为解决真实问题建模而到来。有时，商业问题和数据科学应用之间的关系十分清晰，就像在正确辨别信用卡欺诈交易的场景中一样。在其他情况下，分离商业问题和数据科学应用可能有多个步骤。一个数据科学工程的原始和最终驱动力，通常基于部门级需求（例如，金融、销售、市场等），并且不是源于IT部门的需求，虽然工厂部门可能需要向你提供用来做机器学习的数据集。

2. 获得数据集。用于分析的数据可以从数据库或者数据集中获得。在一些情况下，数据可能从生产系统中提取，例如，一个电子商务应用。最近越来越多的机器学习项目中使用的数据来自多种数据源，包括非结构化数据，例如社交媒体甚至是电子邮件。这就是为什么数据采集过程需要你具有创造力，并且有相关领域的专家提供帮助（诸如一个能给你提供销售数据或者薪资数据的主管）。这一步可能涉及IT部门，需要数据库工程师为你切分提取出数据。

3. 处理数据。在项目的起始阶段，将数据进行清洗，并将原始数据转换成更适合机器学习的数据格式等数据处理任务是十分重要的。对于一些企业数据的状态来说（脏数据、数据不一致、数据缺失等），这一步可能要花费相当多的时间和精力。

4. 使用探索性数据分析。使用统计方法和图表来发掘数据中有趣的特征和模式。有时，对原始数据（或者原始数据的采样）进行绘图，就能揭示出重要的信息，它能帮助你决定项目的研究方向，或者至少能在解释机器学习工程的结果时提供一个关键的思路。

5. 执行特征工程。使用探索性数据分析决定最优的特征变量，在你准备使用的机器学习算法中应用。这个步骤可能需要与相关专家讨论在什么层面上进行探索性数据分析。当然，你需要明确地知道每个特征

变量为算法预测精确度带来的提升程度。

6. 选择一个模型。选择一个适合解决现有问题的机器学习算法，并把你的数据集分为训练集、交叉验证集和测试集（详细内容见第7章）。在现阶段，你需要确定将要使用的机器学习的类型。你想做一个定量的预测、定性的分类，还是仅仅使用聚类技术进行探索？当你在机器学习领域积累一些经验后，你将能更容易判断在某个应用中使用什么算法最合适。

7. 验证模型。没有一个算法，在使用任何数据集的情况下都能打败其他任何算法。对于一个特定的数据集来说，某个算法可能有最好的结果，然而使用不同的数据集之后，其他算法可能效果更好。因此，对于任何给定的数据集，判断哪个方案能产生最好的结果，是一个非常重要的任务。在实践中，选择一个最好的方案是机器学习中最具有挑战性的部分之一。因此，对模型的性能评估（performance evaluation）是项目成功的关键。我们需要测定预测和实测数据的拟合程度，也就是说，我们需要将预测结果值和真实的响应值的接近程度进行量化。我们还需要确定过度拟合的程度，例如，给出的一个算法产生了少量的训练结果错误，但有大量的测试结果错误。特别地，我们还需要评估模型的适用性。更进一步，我们需要决定偏向减少方差还是减少偏差（这是第7章的一个主题）。在第7章中，我们将寻找同时满足低方差和低偏差的算法。

8. 数据可视化（**data visualization**）。一个机器学习项目的最终结果用一个制作精良的可视化（能准确传达算法告诉我们关于数据的本质）来表示最易被理解。传达合适信息的数据可视化不容易创造，可能需要多次尝试才能成功。事实上，建立有效的可视化需要一定程度的创造力和艺术天分。好在，互联网充满了大量有效可视化的例子，你可以用它们来想出你自己的方向。

9. 交流结果。一旦你完成了数据科学部分的处理，你要准备和决策者讨论结果。为了使这一切行之有效，你需要成为一个“数据说书人”，基于数据展示的内容，能讲一个令人信服的故事。要知道大多数经理人没有在数据分析或统计方面的背景知识，你的工作是把结果展现成一个典型的商人可以理解的形式。为了能针对结果采取行动，决策者必须明确知道正在讨论什么——不是在数据科学意义上讲，而是在可行

的商业智慧的意义上。这很不容易，但是讲数据的故事是成为一个数据科学家不可或缺的一部分。

10. 部署到生产环境。一旦项目完成，下一步重要的工作是将机器学习解决方案部署到生产环境中。这个环境采用的形式取决于这个工程的目标。如果你在创造一个推荐系统，那么部署可能意味着给电商网站增加功能；如果你在创造一个客户流失率预测器，那么部署可能意味着营销系统的一个补充；或者如果你致力于一个帮助完成销售交易的网站，那么可能没有任何实际意义上的部署，而是在销售部门标准作业程序（SOP, Standard Operating Procedure）文件上的改变（例如，在7次电话后继续给潜在客户打电话，会带来收益递减）。在生产环境中部署基于R的机器学习解决方案，你也需要特殊的工具。本章后面会讨论这个内容。

在上面这些指引之后，对任何机器学习项目来说都有一些重要的目标。第一，算法的结果应该是可解释的；换言之，组织中的领域专家应该能理解（当然，在你的帮助下）该算法展示了数据的什么内容。接下来，结果需要足够简单，能够解释清楚，与一个极度复杂的解决方案相反，这其中的很大部分取决于你把技术翻译成可行的商业智能的能力。你经常要对精确度进行折衷，换句话说，你会牺牲一些精确度来提高可解释性和简单性吗？如果你牺牲了太多的精度，那么你可能需要重新考虑其他影响因素（例如，牺牲一些简单性换取精确度可能是值得的）。算法也必须快速。这意味着建立模型和用小样本测试它都应该是简易的。最后，算法应该是可伸缩的，这意味着它很快速或是能并行处理多个样本，易于应用在海量数据集中。

1.4 机器学习背后的数学

为了让这本书的受众尽可能广泛，我做了一个慎重的决定——省去数学。我对这个决定并不感到特别兴奋，因为我坚信彻底理解机器学习需要深入理解每个算法的数学原理。虽然在学习的初始阶段你不用数学也可以通过，但是我强烈期待在某些情况下，你想要回来重新学习每个算法，并钻研它背后的数学原理。依我看来，数学基础和机器学习方面的专业知识都是一个数据科学家必须的。在数学领域，你需要知道的包括数理统计（mathematical statistics）、概率论、微积分（calculus）、线性代数、偏微分方程和组合数学。

如果你在大学时没有数学背景，什么时候来打基础都不算晚。现如今有很多可利用的资源来帮助你了解机器学习中数学的最新进展：诸如 Coursera、edX、MIT公开课（MIT OpenCourseWare）和可汗学院（Kahn Academy）这样的MOOC。在YouTube上搜索上面提到的每个数学领域，会找到很多高质量的讲座以进行关注和学习。

一旦你获得了数学上的观点，你将能够回答现在越来越多突然出现的问题：“机器学习真的是科学吗”？我在这个问题上的观点是坚决肯定的！数据科学是多学科的融合，每一学科都是基于之前提到的数学领域。利用这些原理，你可以有效地应用我在前言中提到的“科学方法”。

1.5 成为一名数据科学家

评估你在机器学习领域的职业规划时，你需要考虑成为一名数据科学家所需要的知识。你只需要查看这一职位最近的招聘公告就能理解这需要多方面的能力。很多公司在寻找在数学、统计建模、算法设计与验证、构建生产系统的软件工程、系统架构和管理、分布式计算机系统（例如：Hadoop和Spark）、云解决方案、存储系统和具有行业经验的候选人。再加上，这个人需要具有和首席高管讨论实验结果的能力。这样的候选人被称为“独角兽”（一种现实中不存在的神兽），或者，根据当前讨论的话题，称之为一名数据科学的超级英雄。事实上，这些职位说明描述了一个团队而不是一个个体，但是很多公司坚持数月甚至超过一年来寻找他们的“独角兽”并不肯妥协。

上面提到的各个领域的数据科学需要截然不同的技能、教育和训练。例如，假设一家公司想在上百万的保险索赔案例中定位异常值来鉴定欺诈，他们会需要一位在抽样和多变量统计分析上有经验的人。这家公司也可能想设计并实现一个软件解决方案为生产系统所用。这两种工作能由一人承担吗？这个人有统计的博士学位，同时也能写代码并且设计/配置硬件架构吗？或者反过来说，要让一个软件工程师做数理统计吗？不太可能——这些职责范围通常是互斥的。

对你来说更现实的是预先确定你想成为一个理论家（theorist）还是实验家（experimentalist）。理论家利用中等大小的数据集（或者从海量数据存储中抽样）执行标准的数据分析、数据清理、探索性数据分析、选择模型并验证，来建立项目的理论基础。这个人应该对机器学习及其背后数学原理有着深刻的理解。~~另一方面，实验家更多的是一名软件工程师或者在建立生产软件应用方面有经验的人。这个人应该理解机器学习的结构，以及如何把它部署到生产系统中。这个人可以建立Hadoop集群、写MapReduce代码、管理数据存储硬件、写高可用产品代码并处理对可扩展硬件架构的需求。~~

数据科学定义中包含以上所有特点，这使得雇主和应聘者的生活混乱不清。如果一家公司招聘数据科学家，他们期望找到一个能做复杂统计分析、处理海量数据并设计/建立可扩展软件系统的人吗？很多想要

招聘数据科学家的公司确实坚持要找这样的“独角兽”，但是这就像是招聘一个建筑师，不但能设计房子，还需要他倒水泥并干一些钉钉子的活。

一个解决方案是教育雇主和他们的HR团队：他们真的需要雇佣一个数据科学团队，把“数据科学家”这个职位拆分成专业特长，例如：

- 数学/统计学；
- 计算机科学；
- 软件工程；
- 特定领域的经验。

图1-3展示的数据科学韦恩图2.0并不是对重叠部分的精确描述，而只是一个形象化的指南。注意到中心的独角兽代表了所有学科的非常稀有的交叉点。

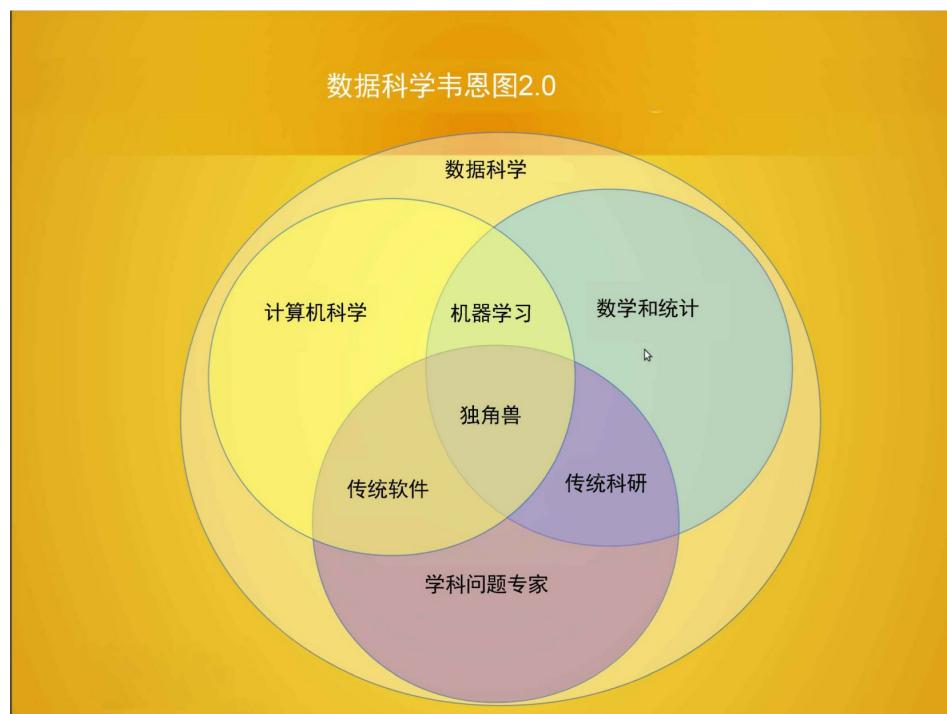


图1-3 数据科学韦恩图2.0

1.6 统计计算的R工程

在过去的几年里，R统计环境在机器语言社区里面逐渐获得了相当突出的重要性。虽然有很多其他选择来执行与数据分析、数据建模和机器学习有关的任务，R成为了今天数据科学家最喜欢的工具。这归功于R在学术界的广泛应用，而不是像SAS和SPSS等商业化产品。最近，R用户社区和SAS及Python社区展开了激烈的辩论，焦点是对数据科学家来说，哪个是最好的工具。R有令人信服的理由，包括：免费、开源、可用性强；广泛使用的可扩展语言；在CRAN中大约有7000个R包用以拓展R功能；一流的可视化功能；一个繁荣的用户社区以及博客集合（例如r-bloggers.com）。

这里有一些关于R的简短事实，来证明它的普及和成长：

- R是薪酬最高的IT技能（Dice.com调查，2014年1月）；
- R是在SQL之后最广泛使用的数据科学语言（O'Reilly调查，2014年1月）；
- 70%的数据挖掘师（data miner）使用R（Rexer调查，2013年10月）；
- R在所有的编程语言中排名15（RedMonk编程语言排名，2014年1月）；
- R比其他任何数据科学语言都发展得快（KD Nuggets调查，2013年8月）；
- R在谷歌搜索先进的分析软件中排名第一（Google Trends，2014年3月）；
- R在全世界有超过两百万用户(Oracle预测，2012年2月)。

1.7 RStudio

在这里，给出RStudio IDE的一个简要介绍是有益的。我强烈建议你使用RStudio来使用R开展机器学习工作。图1-4展示了一个典型的RStudio会话窗口。

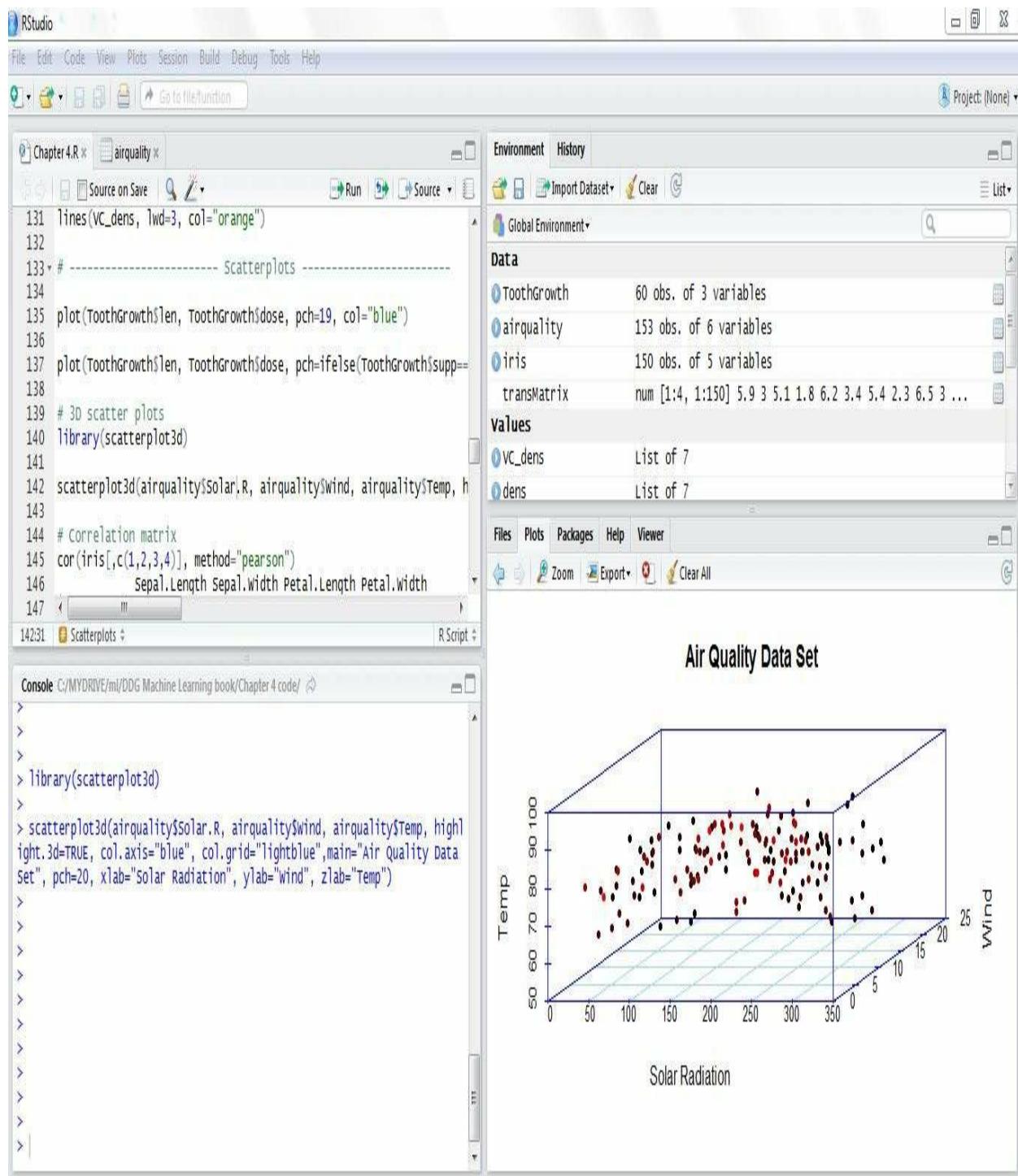


图1-4 Rstudio IDE

RS的屏幕被分割为4个窗格。左上角是R脚本编辑器，你可以将你的R编程代码输入到不同的代码文件中。你可以同时打开多个代码文件，以便从一个脚本中复制代码到另一个脚本中。为了执行R代码行，

你需要做的只是用鼠标把代码行进行高亮并在编辑器的工具栏单击Run。在执行代码时，这些行会显示在左下角的窗格中，也就是R控制台。代码的每一行都以“>”符号开始。在这里，你也能看到运行的代码的效果。举个例子，如果在控制台中输入了以下内容，你马上就能看到计算的反馈。

```
> mean(iris$Sepal.Length)
```

```
[1] 5.843333
```

在右上角，你能看到Environment标签，它包含两个部分：Data展示现在使用所有数据集，Variables展示你的R脚本创建的所有程序变量。注意，如果点击数据集的名称，一个新的数据浏览器会在左上角出现，这样就可以滚动查看数据集中的所有数据项了。

在右下方的窗格，你会看到许多不同的标签：Files、Plots、Packages、Help和Viewer。所有的图表特征结果都显示在这个窗格中，帮助信息也指向这里。你也能在这里找到R环境中所有在用的R包。

1.8 使用R包

R以一种非常积极的方式拥抱开源，通过使用所谓的“包”来扩展软件的基本功能。有大量的通用包（当前大约是7000个），其中很多涉及有用的统计方法，也有特定领域的包：金融、天文学、分子生物学、生态学等。你可以用下面给出的R脚本来找到当下可用的R包数目：

```
> dim(available.packages())
```

R包贡献在综合R归档网（CRAN, The Comprehensive R Archive Network）中，这是一个由世界各地的专家维护的资源库。为了找到离你地理位置最近的CRAN镜像，请访问www.r-project.org，点击CRAN链接，选择你的国家/位置。在那里，你可以看到一页按名称排列的可用CRAN包。这个网页非常长，所以你需要使用浏览器的Find工具来通过关键词来搜索。例如，有6个R包的名字中含有关键词“金融（finance）”。也有一些在CRAN上找不到的私有R包，你可以用谷歌来①查找并下载安装到你本地配置中。

一旦找到了满足你需求的R包，你需要在本地配置中②安装它。例如，这是安装lubridate包的命令：

```
> install.packages("lubridate")
```

一个包只能安装一次。安装完成后，你需要做的是用library()把它加载到内存中。library()函数用来加载基础R配置中未包括的函数库（函数和数据集的集合）。

```
> library(lubridate)
```

一旦成功安装了一个新的包，你应该去访问CRAN上该R包的页面来下载参考手册和任何可能有帮助的简介。参考手册包括了简介和如何使用包中每个函数的示例，包含每个函数完整的参数列表。不过，手册只是用于参考，通常不适合作为学习材料。简介更像是一份学习指南或者在大学教育中使用的实际案例，因此在学习包的特征和使用方法时是

非常有用的。遗憾的是，并不是所有的包都有简介。

对于本书中更深入的主题来说，R是机器学习资源中的一个宝库。在基础R环境以及许多针对机器学习算法的附加包中，你可以找到许多机器学习功能。举个例子，基础R包含有stats包，里面有常用算法，例如：lm()用来拟合一个简单的线性回归模型，glm()用来拟合广义的线性模型；如逻辑回归：hclust()用来做聚类分析，kmeans()用来做k均值聚类，prcomp()用来做基本的组成成分分析；还有其他很多功能。

除此之外，还有许多机器学习的附加包可以补充基础R包的功能。例如，class包中的knn()用来做k最近邻算法，tree包中的tree()用于拟合分类树或是回归树，randomForest包中的randomForest()用来实现随机树算法，e1071包中的svm()用来实现支持向量机，还有很多其他功能。

为了找到其他满足机器学习需求的R包，你可以使用谷歌。例如，你想寻找用来实现进化算法的包，可以搜索“R中的进化算法”。结果会告诉你参考DEoptim包，这个包中含有你需要的机器学习算法。

1.9 数据集

本书尽力教大家机器学习的方法，因为机器学习是关于数据的，所以我们需要大量的样本数据集以供在例子中使用。为了让事情简单一点（并且不需要你花费大量时间去寻找数据集），书中使用的大多数数据集都是R软件在安装时自带的。使用如下命令，查看可用的数据集列表：

```
> data()
```

你所能看到的列出的数据集，取决于你安装了什么包和你在内存中加载了什么包。R包中通常包含可以用来检验函数功能的数据集。可以使用如下命令，查看特定R包中包含的数据集：

```
> data(package="plyr")
```

要查看某个数据集中的更多内容，你可以使用在数据集名称前面加?的命令，就像下面展示的这样。R会给出数据集的简短介绍、观测（例子）的数目、变量名（特征）列表、代码示例和在很多情况下都有的每个变量的描述。图1-5展示了R给出的关于airquality数据集的帮助信息。本书会使用许多著名数据集，强烈建议你熟悉每个数据集中的每个变量。

```
> ? airquality
```

你可以使用以下命令来将某个数据集加载到内存中：

```
> data(iris)
```

你将在RStudio的Workspace标签页看到数据集的名称。

New York Air Quality Measurements

Description

Daily air quality measurements in New York, May to September 1973.

Usage

```
airquality
```

Format

A data frame with 154 observations on 6 variables.

```
[,1] Ozone    numeric Ozone (ppb)
[,2] Solar.R numeric Solar R (lang)
[,3] Wind     numeric Wind (mph)
[,4] Temp     numeric Temperature (degrees F)
[,5] Month    numeric Month (1-12)
[,6] Day      numeric Day of month (1-31)
```

Details

Daily readings of the following air quality values for May 1, 1973 (a Tuesday) to September 30, 1973.

- **Ozone:** Mean ozone in parts per billion from 1300 to 1500 hours at Roosevelt Island
- **Solar.R:** Solar radiation in Langleys in the frequency band 4000–7700 Angstroms from 0800 to 1200 hours at

图1-5 关于airquality数据集的R帮助信息

1.10 在生产中使用R

当你用R完成了一个机器学习项目时，接踵而来的问题就是能否将解决方案部署到生产环境中去。答案通常是否定的。因为性能和内存限制，开源的R语言对生产系统来说通常不是一个好的选择。对于需要连接到特别大数据集的海量数据机器学习项目来说，性能和内存特别重要，所以除了使用其他语言，例如C++或者Python之外，还有其他选择吗？你可以从使用bigmemory包开始，它能帮助你创建、存储、连接和操纵大量矩阵。然后你可以通过添加biganalytics、bigalgebra、bigtabulate和bigpca包来获得更高级的功能。这些包可以在建立一个可行的生产系统之路上帮助你走得更远，但你依然可能在路上遭遇绊脚石。好在，有几个商业版本的R环境适合企业级应用。其中一个是Revolution Analytics公司提供的Revolution R Enterprise（RRE）。另一个是TIBCO公司的TIBCO Enterprise Runtime for R（TERR）。这两个商业版本都能提供适合R机器学习算法的最佳版本，用以快速处理海量数据。

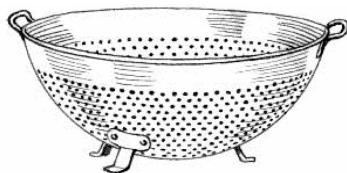
1.11 小结

在本章中，我们对机器学习领域（也被称为统计学习）进行了介绍。另外，我们做好了开始一个数据科学项目的准备，并提供了一个路线图来作为本书的向导。建立一个机器语言解决方案的第1步是探索以各种常用格式连接数据集的方法。在第2章中，你将看到很多连接数据的例子，可以加入到你的数据科学工具箱中。

下面是本章知识点小结：

- 简要介绍了接下去要学习的两种机器学习类型：监督学习和非监督学习。
- 使用Kaggle数据科学挑战赛网站，介绍几个用机器学习解决实际问题的案例以及一些其他领域未来可能解决的问题。
- 学习了数据科学处理时涉及的步骤。
- 讨论了如何才能成为一名资深的数据科学家，以及需要逐渐掌握的数学基本功。
- 本书基于R统计环境编写，所以我们讨论了这个重要的工具，以及如何在RStudio环境下使用它。
- 讨论了在机器学习中R包的威力，并介绍如何使用多种R包来处理数据集。
- 最后，我们讨论了如何在生产系统中使用R，这也是很多项目的终极目标。

第2章 连接数据



✓ 数据科学和它的使能技术机器学习一样，都是关于数据的，即使用海量数据训练算法，对未来事件作出预测；也会对存储的数据进行筛查，发现对商业有战略价值的模式。所以很自然地，机器学习过程中一个重要的部分就是从与待解决的问题相关的分散资源中访问数据内容！很多机器学习和数据分析讨论的前提是你已经有干净的数据，可以直接把它们应用在探索性数据分析工具中，然后选择一个合适的机器学习模型。遗憾的是，这种情况很少发生，更多的时候，你需要定位数据，确定它使用了哪种格式，找到一个有连接数据功能的R包，最后，连接数据并把它读入R数据框（data frame）中——这就是本章的全部主题。这一过程为下一个重要的步骤（数据处理）打下了基础，而后者是本书第3章的主题。

让我们后退一步，为“数据”下一个定义：

✓ 数据是属于同一群体的定性或定量的变量的值；是你感兴趣的一组对象的集合，其中的一个变量是对一项的特征的度量。

群体的例子是一种产品的销售数据、广告效果数据和制造过程数据。定性的变量的例子是销售价格、某个广告的点击数和一小时内一种产品的产量。在机器学习项目中，连接数据阶段的目标是定位并获取能为问题域提供帮助的数据源。

数据连接在机器学习项目中的重要性不容小觑。在和其他组织一起从事机器学习项目时，我学到的最重要的知识就是，人们天生愿意把数

据存储在多个位置，并以多种格式存储，而这些数据都可以用于机器学习。从某种意义上来说，缺乏合适的数据方法决定了我们需要具备从各种数据源中灵活获取数据的能力。无论你在初创公司、大型企业还是科研机构工作，你都需要掌握多种连接数据集的方法。

在可用数据没这么多且我们从数据中学习的能力没有这么强时，这些都不是问题。所谓的“大数据”产业，就是以有效的方式利用不同来源的数据。数据源的种类和深度在不断提升，这个方面一个很好的例子是，非结构化社会媒体的数据使用量在不断增加。利用这些数据可以得到人气和信誉值，并结合交易数据集来达到空前的预测能力。

本章的目标是为你提供一个有用的连接数据的工具包，以便你在后续的机器学习项目中重复使用。在这里学到的连接数据的步骤，将成为你构建数据管道的第一步，数据管道将在第3章讨论。当一名数据科学家遇到一类全新的数据源时，他必须研究引入机制（和这里展示的相似），并把数据加入到本章提供的工具中。逐渐地，你将拥有一个不断完善工具包，它的功能不断增强，并能连接常见的数据源。像R这种开源工具的优势是，给一些时间，就有人能开发出新的包来处理种类不断增加的数据源。]实际上，这就是Twitter上所发生的事情。

在本章中，我们会学习一些连接各种类型的数据内容并将其引入到R环境中作为学习算法输入源的方法。这一步代表了机器学习方程的第一步。以下是本章的主题列表：

- 管理你的R数据工作环境；
- 数据集的种类和来源；
- 从网络上下载数据集；
- 读取数据文件；
- 抓取网页中的数据；
- 使用SQL连接数据；
- R中的SQL等价表述；
- 读取Twitter数据；
- 读取谷歌分析数据；
- 写数据。

2.1 管理你的工作目录

在R环境中开始一个新的机器学习项目时，一个基本的步骤是决定在哪里存储你的工作内容。工作目录用于储存各种组成你项目的文件，包括数据文件、R脚本、图表文件、RDATA文件，也包含你分析得出的文件（Word、PowerPoint等格式）。有些人使用工作目录下的“data”子文件夹来存储这些数据文件。

一旦创建了工作目录，你就需要在R环境中管理它的位置。R有两条命令来管理工作目录：getwd()用于检索目前的工作目录，setwd()用于创建新的工作目录。在打开RStudio时，你可以在R控制台中用setwd()函数来创建工作目录，下面是使用相对路径的方法：

```
> setwd("./MYPROJECT")
> getwd()
[1] "C:/Users/Dan/MYPROJECT"
```

使用绝对路径的方法，你可以使用以下语法：

```
> setwd("/Users/dan/MYPROJECT")
> getwd()
[1] "C:/Users/dan/MYPROJECT"
```

如果你和我一样，也是Windows用户，希望使用Windows的语法，即使用反斜杠 (\) 指定路径名，那么你需要使用特殊的双反斜杠 (\)，如下所示：

```
> setwd("C:\\\\Users\\\\Dan\\\\MYPROJECT")
> getwd()
[1] "C:/Users/Dan/MYPROJECT"
```

管理工作目录的另一个方法是使用RStudio的功能：Session -> Set Working Directory -> Choose Directory来指向你需要的目录。

当你在进行一个机器学习项目时，定期保存工作目录下的工作文件是很有必要的。最简单的方式是在RStudio中使用Session -> Save

Workspace As, 然后为你的工作文件输入一个文件名, 通常是你项目的名称, 例如MYPROJECT.RData。工作空间的文件保存了目前工作环境、数据值和数据集的状态。以后, 当重新加载RStudio时, 你可以打开同一个工作文件来重载到之前的状态, 然后你就可以从上次结束的地方继续工作了。你进行的每一个项目都应该有单独的工作目录和工作文件。

2.2 数据文件的种类

有很多类型的文件可以用做机器学习的数据集。数据科学家的工作是在R环境下，创造工具将来自不同数据源的数据集导入，并把它们合并成一致的结构。本章后面的小节会聚焦于特定的数据源类型，并演示如何将数据以R数据框的形式导入内存。一旦数据进入数据框，通常漫长的数据处理过程就开始了。下面是我们接下来会谈到的数据文件类型的列表：

- 逗号分隔值（CSV, comma separated value）文件；
- Excel文件；
- JSON文件；
- HTML网页；
- SQL数据库；
- Twitter；
- 谷歌分析。

2.3 数据的来源

你会发现用于机器学习项目的数据文件有多种可能来源。大多数情况下，你将从所效力的公司的领域专家那里得到数据集。给你提供数据集的可能是一个IT技术人员，可能是财务部门掌管公司Excel数据仓库的人，也可能是一个为公司管理社会媒体效果的咨询师。以下是其他的一些来源的列表。

- 网络上的机器学习数据资源库：一些广为人知的资源库包括加州大学欧文分校的机器学习资源库 (archive.ics.uci.edu/ml)、政府的数据网站（例如data.gov），还有机器学习挑战赛的网站Kaggle (www.kaggle.com)。
- 从应用程序接口（API）中获得：社交媒体数据最常用的API是 Twitter API，但是除此之外还有很多其他API。在谷歌的帮助下，你可以轻而易举地调查出是否有其他社交媒体平台提供R可用的API（通过特殊的包）。
- 抓取网页：几乎你访问的所有网站都能作为数据源使用，特别是那些有组织地显示数据内容的网站。这里唯一的提醒是，你需要一个可以在R中使用的URL，用于呈现包含数据的网页。2.9节会展示从网页中抓取数据的方法。

当你继续学习机器学习的原理时，你应该时刻注意寻找新的数据来源，并考虑：应该如何在机器学习的帮助下使用这个数据集提取知识，从而创造价值？

2.4 从网络中下载数据集

就像在前面提到的，网络上有很多可供下载的机器学习数据源。其中对公众开放的一种数据类型是政府数据。为了演示怎样从网络中下载数据，如图2-1所示，访问San Francisco Data网站（data.sfgov.org），上面包含了大量的市政数据集。要用到的Parking Meter数据集，包含了多方面的仪表特征。目标是直接从网站上下载CSV文件格式的数据集。为此，我们将使用download.file()指令。这里需要注意的是，并不是网络上提供的所有数据集都以一个很好的结构化格式存在。例如，每一行包含与单次观察相关的多个变量。通常情况下，你在网络上找到的数据需要进行大量的再加工，将它转换成适用于数据分析和机器学习的格式。

SF OpenData

About Data Developers Showcase Help

Data catalog

Showing All types in category All categories (clear filters)

parking

Most Relevant

Name	Popularity	Type	RSS
Parking  Publicly available off-street parking in San Francisco as of September 2011. Find attached a data dictionary, a version of the map as a PDF (symbolized to show garage type and capacity of garage), and a shapefile and CSV of the data. This dataset includes location, capacities, and related details of public parking garages and lots; four parking categories: paid, publicly available (PPA); drive up and pay, typically by the hour or by the day; customer parking only (CPO); typically for businesses or religious institutions; permit holder only (PHO, CGO); i.e. employees only, students only, monthly only; free publicly available (FPA); free off-street parking. The data does not contain off-street residential parking spaces or other unmarked ♦♦♦private♦♦♦ parking.	159 views		
Bicycle Parking (Public)  Bicycle parking installed throughout San Francisco that is free and available for public use. Next Update will be December 2014 or January 2015.	6,504 views		
Parking meters  All parking meters owned by the SFMTA and the Port of San Francisco as of March 2014. Find attached a data dictionary, a version of the map as a PDF, and a shapefile of the data. This data includes locations of meters; meter characteristics, such as single-space or multi-space meter; smart meter status; sensor status; owner; on or off-street location; rate areas; cap colors (use type and restrictions): Black: Motorcycle parking; Brown: Tour bus parking; Green: Short term parking; Grey: General metered parking; Purple: Boat trailer parking; Red: Six-wheeled commercial vehicle parking; Yellow: Commercial vehicle parking. Please direct any data inquiries to info@sfpark.org	1,130 views		



图2-1 数据源实例——San Francisco Data, data.sfgov.org

下面的代码定义了一个变量fileURL, 并且将数据集的URL(网址)分配给它。URL是从San Francisco Data网站上获取的。下一步, download.file()命令用于下载文件, 将它存放在工作目录下的data子目录中。接下来使用list.files()函数来确认下载已经完成。注意相对路径./data的使用, 这指的是当前工作目录下的子目录。

```
> fileURL      <-      "https://data.sfgov.org/api/views/7egw-qt89/rows.csv?accessType=DOWNLOAD"
> download.file(fileURL, destfile=".data/SFParkingMeters.csv")
> list.files("./data")
[1] "SFParkingMeters.csv"
```

下一步是使用read.table()函数将数据集读取到数据框中。然后我们可以使用head()来看看前几行数据，检查数据格式是否正确，同时也能熟悉数据元素：

```
> SFParkingMeters <- read.table("./data/SFParkingMeters.csv", sep=", ", header=TRUE)
> head(SFParkingMeters)
  POST_ID    MS_ID MS_SPACEID CAP_COLOR METER_TYPE SMART_METER ACTIVESENS
1 354-20160     -      0.0     Grey       SS          Y           Y
2 354-21030     -      0.0   Green       SS          Y           Y
3 354-21160     -      0.0  Yellow       SS          Y           Y
4 363-05250     -      0.0     Grey       SS          N           N
5 363-05270     -      0.0     Grey       SS          N           N
6 464-04120     -      0.0     Grey       SS          Y           Y

  JURISDICTION ON_OFF_STR OSP_ID STREET_NUM STREETNAME STREET_SEG RATEAREA
1 SFMTA        ON      0.0    2016.0   CHESTNUT ST 39770000.0 Area 5
2 SFMTA        ON      0.0    2103.0   CHESTNUT ST 39790000.0 Area 5
3 SFMTA        ON      0.0    2116.0   CHESTNUT ST 39790000.0 Area 5
4 SFMTA        ON      0.0      525.0  COLUMBUS AVE 42950000.0 Area 3
5 SFMTA        ON      0.0      527.0  COLUMBUS AVE 42950000.0 Area 3
6 SFMTA        ON      0.0      412.0    HAYES ST  68160000.0 Area 5

  SFPARKAREA      LOCATION
1 Marina      (37.800798, -122.43687)
2 Marina      (37.800522, -122.438067)
3 Marina      (37.800589, -122.438525)
4            (37.800053, -122.409985)
5            (37.800088, -122.410035)
6 Civic Center (37.776878, -122.423512)
```

2.5 读取CSV文件

可能你遇到的最常见的数据文件类型是逗号分隔值（CSV）文件类型。这是因为CSV是数据科学社区的通用语言，并且很多软件应用导出的数据格式是CSV。同样地，大多数软件应用和环境（如R）能够读取CSV文件。如果你不熟悉一个CSV文件的样子，只要在诸如Windows记事本（Notepad）这样的工具中打开它即可。CSV文件的格式很简单：文件中的每一行代表了一个观测值，每一列代表一个变量（潜在的特征变量）。R能处理第一行包含一个变量名列表的情况，也能处理第一行丢失的情况（在这种情况下，R会任意分配变量名，你可以在之后重新命名变量）。

一旦你得到了CSV文件，第一步就是把它放进工作目录中。为了将CSV的内容读进内存中以便后续在R中使用，你可以用read.table()函数，这是R提供的把文件读入成为表格形式的一般方法，不单单适用于CSV格式。read.csv()的功能基本和read.table()相同，只不过它只能读取CSV格式，而这种格式通常是由Excel这样的电子表格应用导出的。不论使用哪个函数，文件都被读入数据框对象中。为了演示这部分内容，我们将读入前面的停车计时器数据集。

```
> SFParkingMeters <- read.csv("./data/SFParkingMeters.csv")
```

成功将文件读入之后，你可以用两种方法让内容显示在RStudio中，一种是在Workspace窗格中单击数据框的名称SFParkingMeters；另一种在控制台中输入指令view(SFParketingMeters)。图2-2显示了你将看见的结果。你可以像在电子表格中那样浏览数据，不同的是，这里不允许编辑。我们注意到，这个数据集有29 253条观测值和116个变量。如果你只想看这个数据框的前6行，也可以使用head(SFParkingMeters)。

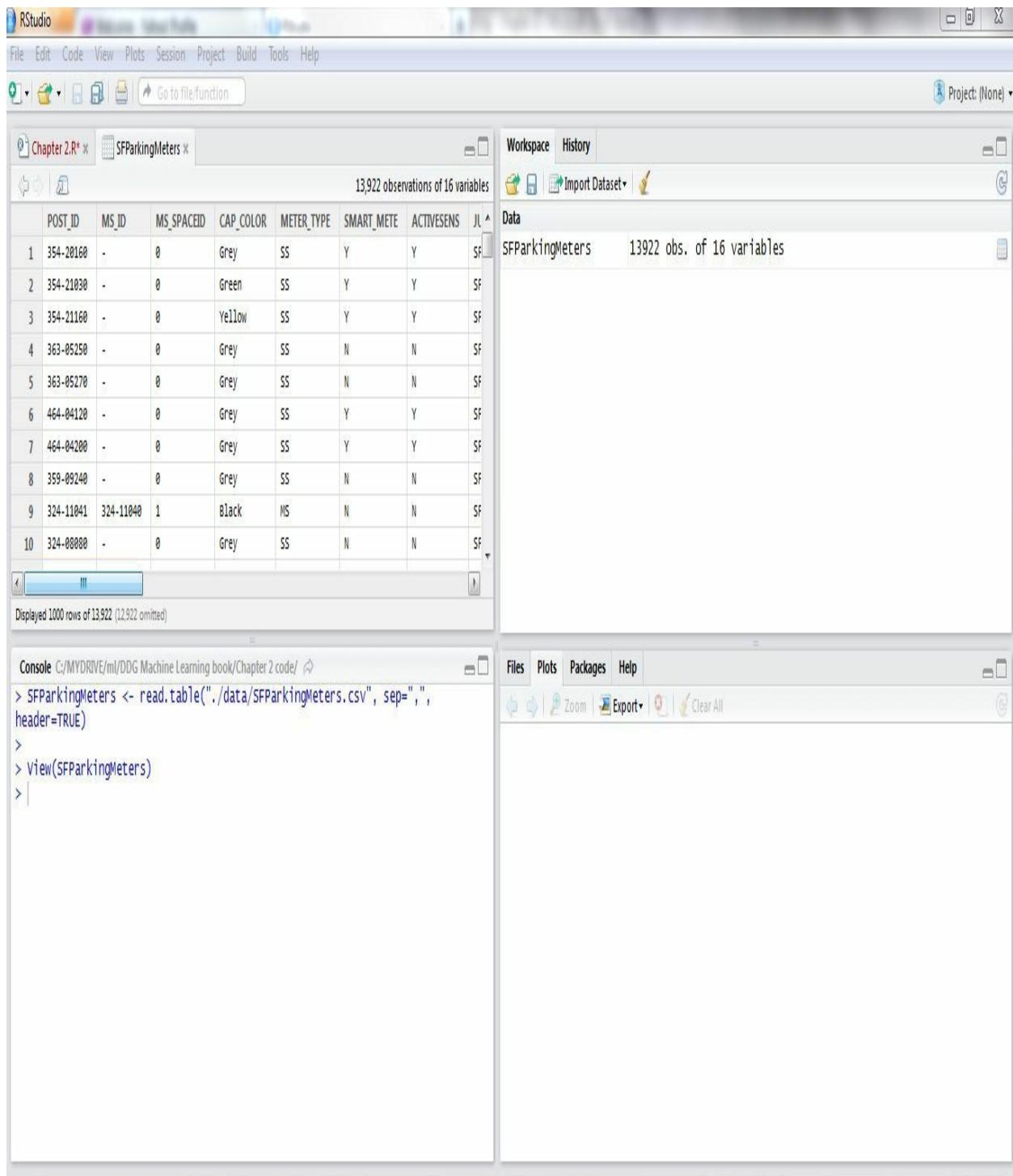


图2-2 使用RStudio查看数据集

另一种有用的读入文件的方式是使用file.choose()函数，它可以嵌入到read.table()或者read.csv()中。这种读文件的方式会弹出一个提示框，提醒选择指向计算机中的哪个文件。

```
> SFParkingMeters <- read.csv(file.choose())
```

2.6 读取Excel文件

机器学习中另一种重要的文件类型是Excel。Excel是应用很广泛的电子表格软件，各种规模的企业都依赖这个工具来存储商业信息。这样一来，你希望在机器学习中使用存放在Excel中的数据就很容易理解了。

R提供了直接从Excel 2007电子表格文件中读取数据的工具：read.xlsx()和read.xlsx2()函数。read.xlsx2()函数通常能更快地处理大型电子表格。为了测试读取Excel文件，我们需要回到San Francisco Data网站去下载同一个Parking Meters数据集，但是这一次是下载XLSX格式的数据。注意到Excel文件是一个二进制文件而不是纯文本文档，我们需要为download.file()函数指定mode="wb"（说明文件是二进制类型）。我们还需要使用library()函数来加载xlsx包，否则使用中会找不到包中的函数。最后，为了把Excel文件读入数据框中，我们将使用read.xlsx2()，并设定参数sheetIndex=1，这指示了读入Excel文件中的哪一个表单。

```
> fileUrl <- "https://data.sfgov.org/api/views/7egw-qt89/rows.xlsx?accessType=DOWNLOAD"
> download.file(fileUrl, destfile="./data/SFParkingMeters.xlsx", mode="wb")
> library(xlsx)
> SFParkingMeters <- read.xlsx2("./data/SFParkingMeters.xlsx", sheetIndex=1)
```

这时，最终结果应该和2.5节读入CSV文件的结果一致。也就是说，数据存入了SFParkingMeters数据框。值得注意的是，Excel的很多企业用户以一种自由的方式使用这个工具，电子表格的内容中有很多变化，而不仅仅是简单的行和列。你可能需要提前对数据进行处理，简化数据结构，才能将Excel文件读入R。

2.7 使用文件连接

另一种从数据源中读取信息的方式是通过文件连接。利用连接，你可以读入CSV文件，就像我们在前面看到的那样。不同的是，你也可以从文本文件中读取数据行。在数据不太规整的情况下，从文本文件中按行读取数据是有意义的。为此，R有一个有用的函数`readLines()`，可以和文件连接一同使用。在我们查看`readLines()`的例子之前，首先来看看文件连接是如何工作的。考虑下面的示例代码：

```
> con <- file("./data/SFParkingMeters.csv", "r")
> SFParkingMeters <- read.csv(con)
> close(con)
> head(SFParkingMeters)
```

前面的例子首先使用`file()`函数来建立CSV文件与命名为`con`的对象的连接。然后，我们调用`read.csv()`函数，使用连接对象作为参数，以读取文件的内容。最后，完成后关闭连接是很好的习惯。跟前面的一样，数据框`SFParkingMeters`包含了文件的内容。

现在让我们回到`readLines()`函数，然后做一些完全不一样的事情。这一次，数据源将会是一个网页，所以我们会使用`url()`函数来提供网页的地址。遵循和之前一样的步骤，但设定参数`n=20`，表示只读取网页的前20行（如果一次只想读取1行，取`n=1`）。

```
> con <- url("http://radicaldatascience.wordpress.com/", "r")
> RDS <- readLines(con, n=20)
> close(con)
> head(RDS)
```

在上文中，使用`head()`，展示了从我的博客上读取的HTML文本。

```
[1] "<!DOCTYPE html>"           "<!--[if IE 7]>"
[3] "<html id=\"ie7\" lang=\"en\">" "<![endif]-->"
[5] "<!--[if IE 8]>"           "<html id=\"ie8\" lang=\"en\">"
```

使用`readLines()`的一个很重要的方面是：数据行存储在特征向量而不是数据框中。你可以使用`class()`函数来查看：

```
> class(RDS)
[1] "character"
```

将文件中的文本行存储在向量中意味着你必须写R代码来处理数据，以解释数据的含义。举个例子，如果数据行中包含Twitter社交媒体内容，那么你可能希望开发一个算法来执行推文中的文本倾向性分析。

R中有很多其他函数涉及连接。要查看完整的列表，使用如下命令：

```
> ? connection
```

2.8 读取JSON文件

为机器学习项目读取数据时，另一种你可能遇到的数据文件类型是JSON，也就是JavaScript Object Notation。JSON是基于文本的开源标准，为创造人类可读的数据交换而设计。它经常和流行的Ajax网络编程技术一同使用。R有两个流行的包能够连接JSON数据文件：rjson和RJSONIO。rjson没有使用R的S3或S4系统，所以它不太容易扩展。同时，rjson也不使用向量化操作，这导致它处理重要数据时速度很慢。同样的，在将JSON数据读入R时，rjson也有点慢并且不能扩展到海量数据。因此，在本节中我们会使用RJSONIO。

我们提供了一个把JSON文件读进R的数据连接案例，第一步需要获得一个URL来下载SFParkingMeters数据集的JSON版本（JSON是San Francisco Data网站提供的另一种文件类型）。大多数的工作由RJSONIO包中的fromJSON()函数来完成。这个函数能将JSON数据内容转换成R对象，以便进行更深入的分析。

下面的R代码首先将JSON URL保存在变量fileURL中。下一步，我们在fromJSON()函数中提交URL，返回数据存储在一个嵌套列表的实体中，包括两个基本的部分：meta和data。我们只需要data部分，所以我们把它存储在列表实体parkdata中。这里的窍门是知道怎样将嵌套列表拆成变量的单独值。为了做到这一点，你需要用parkdata[[1]]来看第一行的观测值，尝试识别一些数据，然后标注这些值的索引，以便后续进行查阅来构造一个数据框。我们可以使用列表处理函数sapply()将数据从列表中抽取出来。最后，我们需要构造一个新的数据框park_df，里面包含初识JSON文件的3个变量：CAP_COLOR、METER_TYPE和STREETNAME。现在，JSON数据以一种合适的数据框的形式存在，我们可以对数据进行常用的分析：

```
> library(RJSONIO)
>     fileURL      <-      "https://data.sfgov.org/api/views/7egw-qt89/rows.json?
accessType=DOWNLOAD"
> parkdata <- fromJSON(fileURL)[[2]]
> park_df = data.frame(
  CAP_COLOR = sapply(parkdata, function(x) x[[12]]),
  METER_TYPE = sapply(parkdata, function(x) x[[13]]),
```

```
STREETNAME = sapply(parkdata, function(x) x[[20]])
)
> head(park_df)
  CAP_COLOR METER_TYPE STREETNAME
1  Grey        SS      CHESTNUT ST
2  Green       SS      CHESTNUT ST
3 Yellow      SS      CHESTNUT ST
4  Grey        SS    COLUMBUS AVE
5  Grey        SS    COLUMBUS AVE
6  Grey        SS     HAYES ST
```

2.9 从网站中抓取数据

将网站当成是一个数据源可能是个奇怪的想法，但是，一旦你想起网站是一组HTML文件，HTML可以更好地表示数据时，那么“读取”网站这一行为就会显得比较合理了。这一技术在需要的数据没有现成的CSV文件或者其他文件类型可供下载时特别有效。作为替代，我们可以直接从网页中“抓取”数据。

为了演示这一读取数据的工具，让我们来访问一个包含数据集的网站，并将其导入R进行处理。图2-3展示了制造业数据采购经理人指数（PMI，Purchasing Managers Index）的历史值。我们的目标是写一个R脚本，直接从网页中读取数据，然后将它载入数据框中进行更深入的分析。接着更进一步，将数据变形，转换成对我们的目标来说更容易使用的形式。

	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
2014	51.3											
2013	52.3	53.1	51.5	50.0	50.0	52.5	54.9	56.3	56.0	56.6	57.0	56.5
2012	52.8	52.4	53.0	53.7	53.2	51.0	50.6	51.1	52.2	51.2	49.5	50.4
2011	59.0	59.3	59.1	58.9	53.7	56.6	52.9	53.0	52.8	51.8	52.1	53.1
2010	57.2	55.8	58.8	58.1	58.3	56.4	56.4	58.0	56.3	57.7	57.6	57.5
2009	34.9	35.5	36.0	39.5	41.7	45.8	49.9	53.5	54.4	56.0	54.4	55.3
2008	50.3	47.6	48.3	48.8	48.8	49.8	50.0	49.2	44.8	38.9	36.5	33.1
2007	49.5	51.9	50.7	52.6	52.5	52.6	52.4	50.9	51.0	51.1	50.5	49.0
2006	55.0	55.8	54.3	55.2	53.7	52.0	53.0	53.7	52.2	51.4	50.3	51.4
2005	56.8	55.5	55.2	52.2	50.8	52.4	52.8	52.4	56.8	57.2	56.7	55.1
2004	60.8	59.9	60.6	60.6	61.4	60.5	59.9	58.5	57.4	56.3	56.2	57.2
2003	51.3	48.8	46.3	46.1	49.0	49.0	51.0	53.2	52.4	55.2	58.4	60.1
2002	47.5	50.7	52.4	52.4	53.1	53.6	50.2	50.3	50.5	49.0	48.5	51.6
2001	42.3	42.1	43.1	42.7	41.3	43.2	43.5	46.3	46.2	40.8	44.1	45.3
2000	56.7	55.8	54.9	54.7	53.2	51.4	52.5	49.9	49.7	48.7	48.5	43.9
1999	50.6	51.7	52.4	52.3	54.3	55.8	53.6	54.8	57.0	57.2	58.1	57.8
1998	53.8	52.9	52.9	52.2	50.9	48.9	49.2	49.3	48.7	48.7	48.2	46.8
1997	53.8	53.1	53.8	53.7	56.1	54.9	57.7	56.3	53.9	56.4	55.7	54.5
1996	45.5	45.9	46.9	49.3	49.1	53.6	49.7	51.6	51.1	50.5	53.0	55.2
1995	57.4	55.1	52.1	51.5	46.7	45.9	50.7	47.1	48.1	46.7	45.9	46.2
1994	56.0	56.5	56.9	57.4	58.2	58.8	58.5	58.0	59.0	59.4	59.2	56.1
1993	55.8	55.2	53.5	50.2	51.2	49.6	50.2	50.7	50.8	53.4	53.8	55.6
1992	47.3	52.7	54.6	52.6	55.7	53.6	53.9	53.4	49.7	50.3	53.6	54.2
1991	39.2	39.4	40.7	42.8	44.5	50.3	50.6	52.9	54.9	53.1	49.5	46.8
1990	47.2	49.1	49.9	50.0	49.5	49.2	46.6	46.1	44.5	43.2	41.3	40.8
1989	54.7	54.1	51.5	52.2	49.3	47.3	45.9	45.1	46.0	46.8	46.8	47.4
1988	57.5	56.2	54.6	55.8	55.5	59.3	58.2	56.0	54.6	55.5	55.6	56.0
1987	54.9	52.6	55.0	55.5	57.2	57.4	57.5	59.3	60.1	60.7	58.8	61.0

图2-3 我们希望从中抓取数据的网页

下面的代码首先下载两个包： XML和reshape2。 XML包中有一个对抓取网页来说非常有用的函数： readHTMLTable()，它能将数据从HTML表格中提取出来。 HTML表格通常是数据在网页中的展现形式。为了方便使用，我们将这一函数的返回数据列表转换成数据框的形式。 使用head(df)展示了现在在数据框中的数据，我们可以看到，与网页中的数据相同。

```
> library(XML)
> library(reshape2)
> webdata <- readHTMLTable('http://www.ism.ws/ISMReport/ content.cfm?
ItemNumber=10752')
> df <- data.frame(webdata[[1]])
> names(df)[1] <- 'Year' # Add a name for first column
> head(df)
  Year JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
1 2014 51.3
```

```
2 2013 52.3 53.1 51.5 50.0 50.0 52.5 54.9 56.3 56.0 56.6 57.0 56.5  
3 2012 52.8 52.4 53.0 53.7 53.2 51.0 50.6 51.1 52.2 51.2 49.5 50.4  
4 2011 59.0 59.3 59.1 58.9 53.7 56.6 52.9 53.0 52.8 51.8 52.1 53.1  
5 2010 57.2 55.8 58.8 58.1 58.3 56.4 56.4 58.0 56.3 57.7 57.6 57.5  
6 2009 34.9 35.5 36.0 39.5 41.7 45.8 49.9 53.5 54.4 56.0 54.4 55.3
```

让我们更进一步，改进从网页中抓取的数据。可以使用reshape2包中的melt()函数，它在快速转换数据框方面特别强大。在这种情况下，我们选择从网站中读入的列表格式，将其改造成有序三元组Year、Month和PMI。同时，也要将PMI转换成数值型，并移除PMI值为NA的观测行（有几个月的PMI没有值）。

```
> df <- melt(df,id.vars='Year')  
> names(df) <- c('Year','Month','PMI')  
> df$PMI <- as.numeric(df$PMI)  
> df <- na.omit(PMI)
```

2.10 SQL数据库

企业数据的常见来源是SQL数据库。SQL数据库是大大小小各种企业的生命线。很多情况下，数据存储在企业级的数据仓库或是部门级的数据集市中。虽然SQL数据库的应用相当广泛，但是最常用的思路是将数据存储在由行、列组成的“表格”中。事实上，大多数数据库应用软件将数据存储在多个表中。机器学习利用SQL数据的目的是写新的SQL查询语句（或者使用现有的）来得到一个平面文件，其中包含你在分析中想使用的数据。鉴于大多数SQL数据库处理工具将数据导成CSV格式，使用上面章节提到的工具来读取SQL数据库产生的CSV文件是一个有效的解决方案。但是，你可能会认为直接从SQL数据库中读取数据更吸引人。

在连接保存在SQL数据库中的数据内容时，有一个好消息是：R拥有几乎每一种数据库的驱动。这多亏了有各种各样的R包。即使你现在使用的是一个没有单独驱动的数据库，也可以使用一个通用的ODBC（开放数据库互联，Open Database Connectivity）进行连接。下面是一些比较流行的SQL数据库R包的列表：

- RMySQL;
- RMongo;
- Roracle;
- RPostgresSQL;
- RSQLite;
- RODBC。

为了展示从SQL数据库中读取数据的过程，我将使用容易获取的工具。早在2012年，我就用Kaggle.com网站中著名的Heritage Health Network机器学习比赛的数据集做过实验，当时用的是Microsoft SQL Server 2012 Express，这是Microsoft企业关系型数据库的一个免费版本。我选择将所有的原始数据文件读入SQL表格中，然后在SQL Server中做大部分的数据处理（这部分内容在第3章会进行讨论）。我使用了

大量存储过程来管理这个过程，这是我建议大家在执行复杂连接时使用的方法，尽管复杂的数据转换最好是在R环境中进行。

这里概括了如何建立一个到SQL数据库的连接，执行了一个简单的SELECT查询来把数据从表格中提取出来，然后将数据保存在R数据框中。这些步骤应用到我使用的SQL数据库和开发环境中，也就是在Windows 7专业版笔记本电脑上运行的SQL Server 2012 Express。你必须仔细研究所用的SQL数据库和开发环境的细节，不过上面这些步骤至少能让你熟悉处理过程。鉴于SQL Server没有自己的R包，我使用了RODBC包进行处理。

1. 在Administrative Tools下的Window Control Panel中，使用ODBC Data Source Administrator工具来创建一个用户DSN（数据源名字）。我将“Heritage”这个名字赋给用户DSN。
2. 加载RODBC库。
3. 使用`odbcConnect()`函数建立一个到DSN是“Heritage”的数据库的连接。
4. 向数据库传递一个SQL SELECT查询语句，用`sqlQuery()`函数进行连接，然后将结果集保存在数据框中。
5. 关闭连接。

下面是完成这些操作的所有R代码。最后，我们在结果数据框中展示所有的变量名，并计算选中变量PayDelayI的平均值，来展示从SQL数据库中传递过来的数据是完整的。

```
> library(RODBC)
> con <- odbcConnect("Heritage", uid="dan")
> df <- sqlQuery(con, "SELECT TOP 1000 [MemberID]
  ,[ProviderID]
  ,[Vendor]
  ,[PCP]
  ,[Year]
  ,[Specialty]
  ,[PlaceSvc]
  ,[PayDelay]
  ,[LengthOfStay]
  ,[DSFS]
  ,[PrimaryConditionGroup]
  ,[CharlsonIndex]
```

```

,[ProcedureGroup]
,[SupLOS]
,[dsfsI]
,[CharlsonIndexI]
,[LengthOfStayI]
,[PayDelayI]
FROM [Heritage].[dbo].[Claims"])

> odbcClose(con)
> names(df)
[1] "MemberID"      "ProviderID"        "Vendor"
[4] "PCP"            "Year"              "Specialty"
[7] "PlaceSvc"       "PayDelay"          "LengthOfStay"
[10] "DSFS"           "PrimaryConditionGroup" "CharlsonIndex"
[13] "ProcedureGroup" "SupLOS"            "dsfsI"
[16] "CharlsonIndexI" "LengthOfStayI"     "PayDelayI"
> mean(df$PayDelayI)
[1] 42.944

```

当我们谈及SQL数据库和R时，值得在这里绕一个“小远路”来演示如何在R数据框中执行SQL查询操作。如果你能熟练使用SQL，你可能会发现，在R中使用SQL来连接数据内容比使用R中一些更简洁的数据处理工具（我们会在第3章看到很多这部分内容）要更容易操作。在R中使用SQL的关键是sqldf包，它能帮助你像处理SQL表一样处理数据框，并能用标准SQL语法引用它们。即使你从来没有使用过SQL，你也可能对这个功能感兴趣，因为使用SQL能使一些高难度的R语言结构有更简单的用法。

让我们看一个在R中使用SQL的案例。这是一个典型的商业中关于订单和产品的应用。不引用现存的数据集，我们将通过两个数据框orders和product马上创建一些测试数据。orders数据框有3个变量：order_no、prod_id和qty，分别代表了订单编号、产品ID和订购数量。product数据框也有三个变量：product_id、desc和price，分别代表了产品ID、描述和产品单价。我们将使用SQL连接语句将两个数据框通过共同的关键字prod_id关联起来，然后创建一个新的数据框，其中是包含order_no、prod_id、qty和price的结果集。

```

> orders <- data.frame(order_no=c("10021", "10022", "10023", "10024", "10025"), prod
01", "AC-01", "AD-11", "AE-21", "AM-19"), qty=c(1, 1, 2, 3, 1))
> product <- data.frame(prod_id=c("AC-01", "AD-11", "AE-21", "AM-19", "AG-
40"), desc=c("Widget A", "Widget B", "Widget C", "Widget D",
"Widget E"), price=c(123.50, 25, 55, 17.95, 45.33))
> sqldf("SELECT o.* , p.price FROM orders o INNER JOIN product p ON o.prod_id = p.|
  order_no  prod_id  qty   price
1 10021      AC-01    1   123.50
2 10022      AC-01    1   123.50
3 10023      AD-11    2   25.00
4 10024      AE-21    3   55.00

```

5 10025 AM-19 1 17.95

2.11 R中的SQL等价表述

使用类似于SQL的功能，R有很多种方法能够连接保存在数据框中的数据集。如果你已经了解SQL，那么理解和SQL命令具有相同功能的、R用来操作数据的工具将会是很有用的。在本节中，我们将看到一些在连接数据时早晚会派上用场的工具。为了对这些工具进行举例说明，我们将使用在基础R系统里即可获得的CO2数据集。让我们看看这个数据集的结构和内容：

```
> data(CO2)
> head(CO2)
  Plant Type Treatment conc uptake
1 Qn1 Quebec nonchilled 95   16.0
2 Qn1 Quebec nonchilled 175  30.4
3 Qn1 Quebec nonchilled 250  34.8
4 Qn1 Quebec nonchilled 350  37.2
5 Qn1 Quebec nonchilled 500  35.3
6 Qn1 Quebec nonchilled 675  39.2
```

这个CO2数据集包含了84条观测值和5个变量：Plant、Type、Treatment、conc和uptake。通常，你要对数据集做的第一件事是基于一个过滤器筛选数据行。

```
SELECT * FROM CO2 WHERE conc>400 AND uptake>40
```

R中的等价表述使用了下面这种简单的语法：

```
CO2_subset <- CO2[CO2$conc>400 & CO2$uptake>40, ]
head(CO2_subset)
  Plant Type Treatment conc uptake
12 Qn2 Quebec nonchilled 500  40.6
13 Qn2 Quebec nonchilled 675  41.4
14 Qn2 Quebec nonchilled 1000 44.3
19 Qn3 Quebec nonchilled 500  42.9
20 Qn3 Quebec nonchilled 675  43.9
21 Qn3 Quebec nonchilled 1000 45.5
> dim(CO2_subset) # 8 observations selected
[1] 8 5
```

然后，我们将看一条SQL SELECT语句的ORDER BY子句。在这里，我们希望将结果集根据变量conc进行排列（升序序列），然后根据变量uptake进行排列（降序序列）。下面是SQL的表述：

```
SELECT * FROM CO2 ORDER BY conc, uptake DESC
```

R中的等价表述使用了下面这种简单的语法。我已经加了一些附加的R语句将结果限制在前20行，但是这只是为了方便起见，因为我们不想在这里展示CO2数据集全部的84行观测值。

```
> CO2[order(CO2$conc, -CO2$uptake),][1:20,]
   Plant  Type      Treatment  conc  uptake
15  Qn3  Quebec    nonchilled   95   16.2
  1  Qn1  Quebec    nonchilled   95   16.0
  36 Qc3  Quebec     chilled    95   15.1
  22 Qc1  Quebec     chilled    95   14.2
  8  Qn2  Quebec    nonchilled   95   13.6
  50 Mn2 Mississippi nonchilled   95   12.0
  57 Mn3 Mississippi nonchilled   95   11.3
  43 Mn1 Mississippi nonchilled   95   10.6
  78 Mc3 Mississippi chilled    95   10.6
  64 Mc1 Mississippi chilled    95   10.5
  29 Qc2  Quebec     chilled    95    9.3
  71 Mc2 Mississippi chilled    95    7.7
  16 Qn3  Quebec    nonchilled  175   32.4
  2  Qn1  Quebec    nonchilled  175   30.4
  9  Qn2  Quebec    nonchilled  175   27.3
  30 Qc2  Quebec     chilled   175   27.3
  23 Qc1  Quebec     chilled   175   24.1
  51 Mn2 Mississippi nonchilled 175   22.0
  37 Qc3  Quebec     chilled   175   21.0
  58 Mn3 Mississippi nonchilled 175   19.4
```

另一个强大的SQL结构是GROUP BY子句，用来计算聚合值，例如平均值。在这种情况下，我们想要计算每个不同的Plant值对应的uptake的平均值。下面是完成这个的SQL表述：

```
SELECT Plant, AVG(uptake) FROM CO2 GROUP BY Plant
```

R中的等价表述基于aggregate函数，使用了下面的语法来主导处理过程。第一个自变量x，对于aggregate()来说是CO2[,c("uptake")]]，将uptake这一列从CO2的数据框中分离出来。第二个变量by，是data.frame(CO2\$Plant)，是一个组变量。最后，R函数中的FUN变量用于计算摘要统计量，在这个例子中，代表的是mean。

```

> aggregate(x=CO2[,c("uptake")], by=data.frame(CO2$Plant), FUN="mean")
      CO2$Plant    x
1     Qn1    33.22857
2     Qn2    35.15714
3     Qn3    37.61429
4     Qc1    29.97143
5     Qc3    32.58571
6     Qc2    32.70000
7     Mn3    24.11429
8     Mn2    27.34286
9     Mn1    26.40000
10    Mc2    12.14286
11    Mc3    17.30000
12    Mc1    18.00000

```

我们将举一个如何用R做SQL多表查询的例子来结束讨论。考虑这样一种情况：你想从一张标注了州和省的次表中查找国家。下面是完成这个任务的SQL语句：

```

SELECT c.Type,
c.Plant,
c.Treatment,
c.conc,
c.uptake,
g.country
FROM geo_map g
LEFT JOIN CO2 c ON(c.Type = g.Type)

```

R中的等价表述使用了下面这种基于merge()函数的语法。看看CO2数据集中的前几行观测数据，我们了解到变量Type中包含了植物最初生活的州或省。我们使用这个变量作为通用的键值。下一步，我们将创建一个新的数据框geo_map，它会扮演将国家和州/省配对的查找表的角色。然后，在为geo_map设定了合适的列名之后，使用merge()来生成和SQL多表查询得到相同的结果集joinCO2。注意新的数据框含有一个变量country，它是基于Type的查找值。这和SQL多表查询有相同的效果。

```

> head(CO2)
  Plant  Type Treatment  conc  uptake
1  Qn1 Quebec nonchilled   95    16.0
2  Qn1 Quebec nonchilled  175    30.4
3  Qn1 Quebec nonchilled  250    34.8
4  Qn1 Quebec nonchilled  350    37.2
5  Qn1 Quebec nonchilled  500    35.3
6  Qn1 Quebec nonchilled  675    39.2

> stateprov <- c("Mississippi", "California", "Victoria",
  "New South Wales", "Quebec", "Ontario")
> country <- c("United States", "United States", "Australia", "Australia", "Canada")

```

```
> geo_map <- data.frame(country=country, stateprov=stateprov)
> geo_map
  country      Type
1 United States Mississippi
2 United States California
3 Australia     Victoria
4 Australia     New South Wales
5 Canada        Quebec
6 Canada        Ontario

> colnames(geo_map) <- c("country", "Type")
> joinCO2 <- merge(CO2, geo_map, by=c("Type"))
> head(joinCO2)
  Type      Plant Treatment conc   uptake country
1 Mississippi Mn1 nonchilled  95    10.6 United States
2 Mississippi Mn1 nonchilled 175    19.2 United States
3 Mississippi Mn1 nonchilled 250    26.2 United States
4 Mississippi Mn1 nonchilled 350    30.0 United States
5 Mississippi Mn1 nonchilled 500    30.9 United States
6 Mississippi Mn1 nonchilled 675    32.4 United States
```

2.12 读取Twitter数据

鉴于数据科学正在快速演变，我们可以看到社交媒体中的非结构化数据正在为传统的数据源增添异彩。毋庸置疑的是，最流行的社交媒体数据源是Twitter。“非结构化”这一术语用于描述社交媒体数据，因为它不像典型的企业事务数据那样通常呈现表格形式。非结构化数据是自由格式，可能是纯文本，在推文中是140个字符、缩略语、控件、标签和特殊符号。就像其他所有数据源一样，使用这类数据的第一步是将它读进R环境。在本节中，我们将分步演示使用R读取推文的过程。

遗憾的是，使用Twitter作为数据源有点费劲，因为从2013年3月开始，安全要求成为了必备条件（在此之前，读取数据容易得多）。你必须拥有一个Twitter账户，并在你的R代码中提供对这个账户的身份验证。

首先，我们要安装twitteR包，来帮助读取推文。这是基于R的Twitter客户端，这个包提供了一个到Twitter网页API的接口，也包括了ROAuth库（开放式身份验证），用Twitter对我们进行身份验证必须用到该库。

```
> install.packages("twitteR") # Contains ROAuth  
> library(twitteR)  
> library(ROAuth)
```

下一步，我们将定义几个变量来存放几个特殊的URL，用于和Twitter进行通信：requestURL、accessURL和authURL。而且，如果你是一个Windows用户，需要从下面提供的URL中下载特殊的CA（社区驱动的证书颁发机构）证书文件。

```
> requestURL <- "https://api.twitter.com/oauth/request_token"  
> accessURL <- "https://api.twitter.com/oauth/access_token"  
> authURL <- "https://api.twitter.com/oauth/authorize"  
> download.file(url="http://curl.haxx.se/ca/cacert.pem", destfile= "cacert.pem")
```

接下来，你要访问<http://dev.twitter.com/apps/new>，用你的Twitter账户登录，创建一个Twitter应用。然后在页面上输入详细信息，才能通过Test Auth按钮得到用户关键字和用户秘钥。你需要将秘钥复制粘贴到你的R代码中。

```
> consumerKey <- "tu4dkb9fHLgLrNptaI2CA"  
> consumerSecret <- "UzlhoMFyF9IZ6bxG89DLdPB74VUzur3mBWcr6LcVE"  
> Cred <- OAuthFactory$new(consumerKey=consumerKey,  
+ consumerSecret=consumerSecret,  
+ requestURL=requestURL,  
+ accessURL=accessURL,  
+ authURL=authURL)
```

现在输入下面这行代码，要求Twitter提供一个数值型的PIN号码给你。你将看到R控制台的消息，指导你将控制台中展示的特殊URL复制粘贴到你的浏览器中（注意：如果你无法高亮显示控制台中的URL并将它复制到剪贴板中，需要升级RStudio了）。一旦你做完这个，Twitter API会展现一个网页，包含7位的PIN。你需要把这些数字打印到一直等待输入的控制台中。

```
> Cred$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))
```

这时，可以使用save()函数将你的Twitter身份验证保存下来，便于以后使用。最后，你需要用twitteR包中的registerTwitterOAuth()函数注册你的Twitter身份验证。

```
> save(Cred, file="twitter authentication.Rdata")  
> registerTwitterOAuth(Cred)  
[1] TRUE
```

如果想在以后读取推文，只需要使用下面这段代码：

```
> load("twitter authentication.Rdata")  
> registerTwitterOAuth(Cred)
```

最后，我们已经做好了读取推文的准备了。在下面这段代码中，我们使用了searchTwitter()函数，并给它传递了一个搜索关键字。在这个例子中，标签#MLB代表了美国职业棒球大联盟。Twitter API限制了你可以读取的推文最多是499条。推文以列表的形式返回，所以我们使用

twListToDF()函数将它们转化成数据框的形式，便于后续分析。最后，我们可以将推文以CSV的格式写出，便于保存。

```
> MLB.list <- searchTwitter('#MLB', n=499, cainfo="cacert.pem")
> MLB.df = twListToDF(MLB.list)
> write.csv(MLB.df, file='MLBTweets.csv', row.names=F)
```

2.13 从谷歌分析中读取数据

另一个非常有用的数据源是谷歌分析。很多公司使用谷歌的网站统计服务来观察网站的使用情况。谷歌分析的成熟程度是令人惊讶的，其中最大的优点是：这项服务是免费使用的。一旦植入到网站中（使用谷歌提供的一个可以添加到网站中的代码片段），就能收集大量数据，用于后续分析。虽然谷歌提供了一套令人印象深刻的工具来分析网站统计数据，你也可以直接连接数据，这样就能用这些数据执行自己定制的机器学习算法。在本节中，我们会介绍如何连接谷歌分析数据。

有两种方法连接谷歌分析数据。第一，你可以使用你用来跟踪网站的账户来登录谷歌分析。你运行的每一份报表都有一个Export按钮，可以用来下载报告中的原始数据。我们的想法是，用一段长时间范围的数据运行得到一份详细的报表，这样你就能得到一份优良的数据样本，可用于后续的机器学习分析。我们将在下面的案例中看到这样的示例。连接谷歌分析的另一种方法是使用RGoogleAnalytics包，其中包含了大量R函数，能帮助你使用谷歌分析Data Export API。我们也能在下面的例子中看见这种用法。

让我们首先用导出工具连接谷歌分析数据。在这个例子中，我用一个监控客户端网站的账号登录谷歌分析。我通过Traffic Sources -> Sources -> All Traffic这一路径选择一种标准报表。在查看了报表之后，我从顶端的水平菜单选择Export -> Excel (XLSX)，将文件保存在工作目录下面的data文件夹中。谷歌分析生成的Excel文件有多个工作表，通常有一个Summary表和一个或多个Datasheets表。取决于你想用数据做什么事情，你必须决定把哪个表读入R中。谷歌分析使用的一些列名和xlsx包不兼容，所以你需要在Excel中打开文件，手动编辑一个或者多个可能造成冲突的列名。例如，你需要把列名Pages/Visit改成更适合R的表述，例如Pages_Visit。图2-4展示了一个准备读入到R中使用的表格：

```
> library(xlsx)
> GA <- read.xlsx2("./data/Analytics_All_Traffic.xlsx", sheetIndex=2)
> head(GA)
  Source_Medium Visits  Pages_Visit Avg_Visit_Duration
1   google / organic 1349.0 5.638991845811712 151.54558932542625
```

```
2   (direct) / (none) 562.0 4.119217081850533 114.98220640569394
3 fashiondistrict.org / referral 242.0 3.9214876033057853
140. 30578512396696
4   yahoo / organic 73.0 3.6301369863013697 66.53424657534246
5   bing / organic 71.0 5.549295774647887 104.14084507042253
6   oohlaluxe.net / referral 36.0 4.361111111111111
116. 8055555555555556
Percent_New_Visits      Bounce_Rate
1  0.614529280948851    0.5181616011860638
2  0.597864768683274    0.5747330960854092
3  0.8099173553719008    0.45041322314049587
4  0.684931506849315    0.410958904109589
5  0.647887323943662    0.29577464788732394
6  0.8888888888888888    0.361111111111111
```

现在让我们看看使用RGoogleAnalytics包中的RGoogleAnalytics API。在R中，你需要用OAuth2例行程序来认证你的身份，创建一个查询，并把结果保存在数据框中。在开始之前，RGoogleAnalytics库依赖于另外两个库：RCurl提供了在R中的https支持；RJSON提供了解析API返回的RJSON的功能。

The screenshot shows a Microsoft Excel spreadsheet titled "Dataset1". The data is organized into a table with the following columns:

	A	B	C	D	E	F	G	H	I	J
1	Source_Medium	Visits	Pages_Visit	Avg_Visit_Duration	Percent_New_Visits	Bounce_Rate				
2	google / organic	1349	5.64	151.55	61.45%	51.82%				
3	(direct) / (none)	562	4.12	114.98	59.79%	57.47%				
4	fashiondistrict.org / referral	242	3.92	140.31	80.99%	45.04%				
5	yahoo / organic	73	3.63	66.53	68.49%	41.10%				
6	bing / organic	71	5.55	104.14	64.79%	29.58%				
7	ohlaluxe.net / referral	36	4.36	116.81	88.89%	36.11%				
8	latourist.com / referral	30	1.93	38.70	100.00%	63.33%				
9	upload.facebook.com / referral	21	5.90	303.38	0.00%	9.52%				
10	facebook.com / referral	16	4.94	58.81	62.50%	43.75%				
11	t.co / referral	14	2.36	46.57	35.71%	42.86%				
12		2535	4.87	132.37	64.22%	50.89%				
13										
14										
15										
16										
17										
18										
19										
20										
21										
22										
23										

图2-4 谷歌分析All Traffic Sources报表

```
> install.packages("RCurl")
> install.packages("RJSON")
> library("RGoogleAnalytics")
```

使用下面这段语句能打开一个浏览器窗口，这样就可以用谷歌的

OAuth2认证身份。参照页面上的指示，将Access Token复制到剪贴板中，然后将它粘贴到R控制台中。

```
> query <- QueryBuilder()
> access_token <- query$authorize()
```

接下来，这段语句会使用你上面得到的access_token来创建一个新的谷歌分析API对象。

```
> ga <- RGoogleAnalytics()
> ga.profiles <- ga$GetProfileData(access_token)
> ga.profiles # List the GA profiles
```

下面这段代码是将谷歌分析作为数据源读取过程的主要部分。你建立了一个查询字符串，并使用配置文件设置其索引值。

```
> query$Init(start.date = "2013-07-01",
  end.date = "2013-07-01",
  dimensions = "ga:date,ga:pagePath",
  metrics = "ga:visits,ga:pageviews,ga:timeOnPage",
  sort = "ga:visits",
  #filters="",
  #segment="",
  max.results = 99,
  table.id =
  paste("ga:",ga.profiles$id[1],sep="",collapse=","),
  access_token=access_token)

> ga.data <- ga$GetReportData(query)
[1] "Your query matched 88 results that are stored to dataframe ga.data"
> head(ga.data)
#   date      pagePath
1 20130701 /SearchResult.asp?Type=ALL&String=&CompanyID= 127&CategoryID=0
2 20130701 /SearchResult.asp?Type=ALL&String=&CompanyID= 130&CategoryID=0
3 20130701 /SearchResult.asp?Type=ALL&String=&CompanyID= 175&CategoryID=0
4 20130701 /SearchResult.asp?Type=ALL&String=&CompanyID= 181&CategoryID=0
5 20130701 /SearchResult.asp?Type=ALL&String=&CompanyID= 184&CategoryID=0
6 20130701 /SearchResult.asp?Type=ALL&String=&CompanyID= 186&CategoryID=0

  visits  pageviews  timeOnPage
1    0        3          11
2    0        4          20
3    0        2          13
4    0        1           1
5    0        2          11
6    0        3            6
```

2.14 写数据

在一个机器学习项目中工作时，虽然数据科学家做的通常都是将外部文件导入R中，但有时在R环境下把数据写到外部文件中也是很有必要的。好在，我们在本章中见过的很多用于数据连接的R包都提供了写文件的功能。例如，`write.table()`函数能写一个CSV文件。在下面的例子中，我们将用R移除数据框的第一列（变量`POST_ID`），并写出一个不包含这个变量的新版CSV文件。然后，仅仅为了证明它真的有效，我们将新的CSV文件读回R中，然后使用`head()`来展示它的前几行。

```
> tempDF <- SFParkingMeters[,-1] # Remove POST_ID variable
> write.table(tempDF, file="./data/newSFParkingMeters.csv", sep=",")
> newSFParkingMeters <- read.table("./data/newSFParkingMeters.csv", sep=",")
> head(newSFParkingMeters)
  MS_ID MS_SPACEID CAP_COLOR METER_TYPE SMART_METER ACTIVESENS JURISDICTI ON_OFF_S
1 -     0      Grey      SS       Y       Y      SFMTA      ON
2 -     0      Green     SS       Y       Y      SFMTA      ON
3 -     0    Yellow     SS       Y       Y      SFMTA      ON
4 -     0      Grey     SS       N       N      SFMTA      ON
5 -     0      Grey     SS       N       N      SFMTA      ON
6 -     0      Grey     SS       Y       Y      SFMTA      ON

  OSP_ID STREET_NUM STREETNAME   STREET_SEG RATEAREA  SFPARKAREA
1  0      2016    CHESTNUT ST 3977000  Area 5  Marina
2  0      2103    CHESTNUT ST 3979000  Area 5  Marina
3  0      2116    CHESTNUT ST 3979000  Area 5  Marina
4  0      525     COLUMBUS AVE 4295000  Area 3
5  0      527     COLUMBUS AVE 4295000  Area 3
6  0      412     HAYES ST   6816000  Area 5 Civic Center

  LOCATION
1 (37.800798, -122.43687)
2 (37.800522, -122.438067)
3 (37.800589, -122.438525)
4 (37.800053, -122.409985)
5 (37.800088, -122.410035)
6 (37.776878, -122.423512)
```

同样地，`xlsx`包的`write.xlsx()`函数、`rjson`包的`toJSON()`等，都能实现类似的功能。

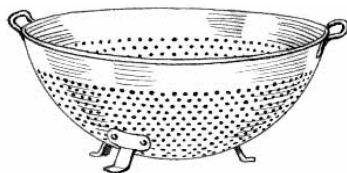
2.15 小结

在本章中，我们了解了为何数据源是机器学习方程的命脉。我们介绍了许多连接原始数据并将其导入R环境中以用于机器学习算法的方法。下一步将会是“加工”这些原始数据，便于你选择的算法能够使用数据。这叫做“数据处理”，这是第3章的主题。

下面是本章主要内容的小结：

- 机器学习用的数据以各种形式存在，仅举几例，包括CSV、Excel和JSON。
- 使用各种各样的R包，你可以直接连接以SQL数据库表形式存在的数据。
- 来自像Twitter这样的社交媒体的数据已经成为机器学习应用流行的数据源。在R的帮助下，你可以很容易地连接Twitter作为数据源。
- 谷歌分析代表了另一种激动人心的机器学习数据源。
- 你可以非常容易地将数据从R中写到外部文件中。

第3章 数据处理



“数据处理”（data munging）是一个不同寻常的术语，用来描述数据科学项目中的一个步骤：将数据集转换成更适用于机器学习算法的形式。数据处理是“数据管道”形成的主要要素之一。这一系列的处理步骤需要获得原始数据，并将它们转换成便于生产系统使用的形式。这项工作包含清洗、转换、操作、解析、过滤和映射，将原始状态的数据处理成更精炼的形式。数据处理是机器学习过程中很重要的一个步骤，经常花费整个项目中高达80%的时间和成本。可以把数据处理看成是重要的数据预处理阶段，这个阶段能使数据更适合机器学习算法。数据处理过程可能是冗长乏味的，但是其目标和收益是久经考验的：经过数据处理后，数据分析和机器学习阶段会（或者至少可能会）容易得多。此外，处理过程使得数据更易于建模和可视化。

数据处理的一个主要目标是整理数据。也就是说，每个变量形成一列，每个观测值形成一行，每个表格存储同一类观测结果。从领域专家那里获得的数据并不一定能符合机器学习的模型（事实上，很少能符合），所以你的工作是在工作流程中规划一系列的步骤，从原始数据开始，最后得到机器学习算法的训练数据。十分重要的是，要记录下你在数据处理过程中所做的所有步骤，这样才能有可复制的分析过程。你应该保留所有用于数据处理的R脚本，并标注运行顺序。

鉴于你已经检查了原始数据集，这里有一些最佳实践指南，在决定数据处理步骤对于特定项目来说是否必要时，它们可能会给你帮助。

- 列名应该易于使用，并能提供有效信息。
- 行名应该易于使用，并能提供有效信息。
- 明显的错误数据应该删除，或者如果可行的话，进行修正。
- 变量值的格式应该具有一致性（例如，日期格式YYYYMMDD）。
- 变量值应该具有一致性（例如，一个国家变量应该只有一种表述：US、USA或者United States）。
- 将缺失值降到最低。
- 已经对变量进行了合适的转换。

为了实现上述目标，必须进行一系列的数据处理操作。具体需要执行什么操作取决于数据和机器学习问题的具体需求。你的起始点很重要。如果从图3-1所示的数据入手，前方有很多工作在等着你。

The screenshot shows a Microsoft Excel spreadsheet titled "FUNDINGPLAN 10Demo [Compatibility Mode] - Microsoft Excel". The interface includes a ribbon bar with Home, Insert, Page Layout, Formulas, Data, Review, View, and Developer tabs. The Home tab is selected.

The main content of the spreadsheet is a "SALES PLAN YEARS 1-3" table. This table has columns for Sales Volume (Units) and Year 1 (January-07 to December-07), followed by a row of Totals. The data rows include Flowers, Organic vegetables, Grains, Dairy, and Piggery. Below this table is a section for "Prices" with columns for Prices, Per unit, and Variance. A note says "Enter unit prices here" and "You can vary each forecast by altering the variance percentage". The data rows for prices correspond to the same categories as the sales table.

Below the prices is a "Revenues" table, which includes a "Yield" column. The data rows for revenues also correspond to the categories in the previous tables. The final section shown is "Sales Volume" for Years 2 & 3, with columns for Units, March-08, June-08, September-08, December-08, Totals, March-09, June-09, September-09, December-09, and Totals. The data rows for this section are Flowers, Organic vegetables, Grains, Dairy, and Piggery, with values matching the previous sections.

At the bottom of the screen, there is a navigation bar with icons for Back, Forward, Home, ToolPak, Guide, Set Up, Equity Plan, Fixed Assets Plan, Sales Plan (which is highlighted in blue), Cost of Sales, and Fixed Expenses. The status bar at the bottom right shows "Ready" and "100%".

图3-1 一个典型的、不利于导入R环境的Excel管理报表

虽然这些数据看起来很整齐，但是电子表格有好几类数据，它们有不同的结构/布局（例如，每一部分的列数都是不同的）。你无法轻易地把这类数据导入R，所以在导入R之前，你需要在Excel中对数据进行重新整理。在另一方面，你也可能刚开始就幸运地得到图3-2所示的数据。

据。所有的观测值（行）排列得很好，特征变量（列）都有相同的格式，并且没有缺失值。如果你的数据科学项目得到的原始数据是存放在Excel中的，在将数据导入R之前对行、列进行快速的操作是很有必要的。只举一个例子，如果在Excel中的数据是按行进行排列的，但你更希望按列进行排列，Excel中有一个好用的Transpose工具来帮助你执行这一操作。

	Name	Address	City	State Zip	Phone	Territory Manager	Store Manager	Sales	Year over Year	Shop Audit
1	IN-N-OUT Burger - Alhambra	1210 N Atlantic Blvd	Alhambra	CA 91001	(477) 241-2409	Middleton, Neal I.	Chung, Kristina H.	\$ 1,645,723	-7%	Green
2	IN-N-OUT Burger - American Fork	601 West Main St	American Fork	UT 84003	(877) 379-3735	Goldstein, Gretchen O.	Chen, Paige H.	\$ 1,165,626	-3%	Green
3	IN-N-OUT Burger - Anaheim	609 S Brookhurst St	Anaheim	CA 92804	(237) 481-1024	Watts, Tim A.	Melton, Shem E.	\$ 869,871	27%	Green
4	IN-N-OUT Burger - Anaheim Hills	5646 E La Palma Ave	Anaheim Hills	CA 92807	(756) 359-9151	Johnston, Jerome O.	Hill, Gretchen L.	\$ 103,693	96%	Green
5	IN-N-OUT Burger - Arcadia	429 N Santa Anita Ave	Arcadia	CA 91006	(947) 459-2459	Weeks, Shelley E.	Puckett, Karen U.	\$ 201,568	-14%	Red
6	IN-N-OUT Burger - Atascadero	6000 San Anselmo Rd	Atascadero	CA 93422	(633) 155-2228	Wilkinson, Priscilla L.	Song, Patrick O.	\$ 356,327	-53%	Yellow
7	IN-N-OUT Burger - Auburn	130 Grass Valley Hwy	Auburn	CA 96003	(575) 661-8596	Barton, Elsie A.	Hamilton, Elsie A.	\$ 1,762,993	-30%	Red
8	IN-N-OUT Burger - Avondale	1525 Dysart Road	Avondale	AZ 85323	(330) 469-3220	Walton, Beth A.	Bender, Hazel E.	\$ 840,455	-34%	Yellow
9	IN-N-OUT Burger - Azusa	324 S Azusa Ave	Azusa	CA 91702	(640) 562-3124	Hall, Erica A.	Wagner, Malcolm A.	\$ 1,037,849	42%	Green
10	IN-N-OUT Burger - Bakersfield	2310 Panama Lane	Bakersfield	CA 93307	(636) 359-2229	Ross, Douglas O.	McLaughlin, Dolores C.	\$ 1,654,834	-84%	Red
11	IN-N-OUT Burger - Bakersfield	5100 Stockdale Hwy	Bakersfield	CA 93309	(576) 524-6763	Chung, Donald H.	McNamee, Francis C.	\$ 197,340	-50%	Green
12	IN-N-OUT Burger - Baldwin Park	13850 Francisquito Ave	Baldwin Park	CA 91706	(637) 168-1540	Bender, Katherine E.	Raynor, Sandy A.	\$ 229,709	38%	Yellow
13	IN-N-OUT Burger - Barstow	2821 Lenwood Rd	Barstow	CA 92311	(371) 368-0108	Weeds, Paul O.	Moest, Marion O.	\$ 1,992,535	-98%	Green
14	IN-N-OUT Burger - Brentwood	5581 Lone Tree Way	Brentwood	CA 94513	(586) 479-3279	Mangum, Patricia A.	Woodard, Beth O.	\$ 225,743	33%	Green
15	IN-N-OUT Burger - Buena Park	7926 Valley View St	Buena Park	CA 90620	(101) 524-2030	Joseph, Lois O.	Desai, Julia E.	\$ 1,670,422	-29%	Green
16	IN-N-OUT Burger - Burbank	761 N First St	Burbank	CA 91502	(123) 875-6932	Rosenthal, Louis D.	Wallace, Jerome A.	\$ 105,215	12%	Red
17	IN-N-OUT Burger - Camarillo	1316 Ventura Blvd	Camarillo	CA 93010	(919) 986-1153	Bowden, Christina O.	Lawrence, Neal A.	\$ 952,375	-67%	Green
18	IN-N-OUT Burger - Canoga Park	6841 N Topanga Canyon Blvd	Canoga Park	CA 91303	(796) 107-3581	Barton, Danielle A.	Griffin, Jean R.	\$ 472,342	-85%	Yellow
19	IN-N-OUT Burger - Carlsbad	5950 Avenida Encinas	Carlsbad	CA 92008	(973) 878-1624	Underwood, Harvey H.	Dougherty, Kristine O.	\$ 1,428,696	-15%	Yellow
20	IN-N-OUT Burger - Carmel Mountain	11880 Carmel Mountain Rd	Carmel Mountain	CA 92128	(332) 428-0004	Jones, William O.	Powers, Crystal O.	\$ 1,938,067	98%	Green
21	IN-N-OUT Burger - Carson Valley	957 Topsy Lane	Carson Valley	NV 89705	(431) 235-2796	Baker, Frederick A.	May, Alex A.	\$ 308,278	-55%	Yellow
22	IN-N-OUT Burger - Casa Grande	673 N Promenade Pkwy	Casa Grande	AZ 85294	(867) 570-1460	Merritt, Shirley E.	Steele, Eric T.	\$ 1,504,034	-4%	Yellow
23	IN-N-OUT Burger - Chandler	2790 W Chandler	Chandler	AZ 85224	(433) 309-9194	Cross, Jason R.	Teague, Wesley E.	\$ 1,186,986	-90%	Red
24	IN-N-OUT Burger - Chico	2050 Business Ln	Chico	CA 95928	(450) 921-2671	Sharpe, Don H.	Vick, Franklin I.	\$ 1,663,071	-3%	Red
25	IN-N-OUT Burger - Colton	3927 Grand Avenue	Chino	CA 91710	(524) 542-2127	Morgan, Glenda O.	Gallagher, Claire A.	\$ 1,494,583	13%	Green
26	IN-N-OUT Burger - Covina	1725 Eastlake Pkwy	Chula Vista	CA 91915	(626) 301-9313	Hoyle, Scott O.	Solomon, Marian O.	\$ 1,258,185	-42%	Yellow
27	IN-N-OUT Burger - Culver City	17849 E Colima Rd	City of Industry	CA 91748	(271) 112-7496	Allen, Paul L.	Walsh, Marcia A.	\$ 1,687,014	-83%	Red
28	IN-N-OUT Burger - City of Industry	21620 Valley Blvd	City of Industry	CA 91789	(380) 731-5565	Rich, Michelle I.	Monroe, Dwight O.	\$ 871,625	-96%	Red
29	IN-N-OUT Burger - Clovis	382 N Clovis Ave	Clovis	CA 93612	(343) 709-5338	Rich, Jessica I.	Connolly, Wayne O.	\$ 445,022	-89%	Red
30	IN-N-OUT Burger - Corona	2305 Compton Ave	Corona	CA 92881	(757) 125-1154	Grant, Evan R.	Hawkins, Stephanie A.	\$ 395,925	40%	Red
31	IN-N-OUT Burger - Costa Mesa	450 Auto Center Dr	Corona	CA 92882	(393) 114-2522	Proctor, Melinda R.	Middleton, Neal I.	\$ 807,320	77%	Yellow
32	IN-N-OUT Burger - Costa Mesa	594 W 19th St	Costa Mesa	CA 92626	(495) 901-1540	Diaz, Calen I.	Goldstein, Gretchen O.	\$ 1,746,361	71%	Green
33	IN-N-OUT Burger - Covina	1371 N Grand Ave	Covina	CA 91724	(872) 266-5253	Graham, Eugene R.	Watts, Tim A.	\$ 1,838,876	54%	Red
34	IN-N-OUT Burger - Downey	13425 Washington Blvd	Culver City	CA 90292	(610) 401-3230	Watkins, Vickie A.	Johnston, Jerome O.	\$ 889,050	-77%	Green
35	IN-N-OUT Burger - El Cajon	260 Washington Street	Daly City	CA 94015	(706) 765-7094	Hinton, Luis I.	Weeks, Shelley E.	\$ 1,708,265	55%	Green
36	IN-N-OUT Burger - El Cajon	1020 Olive Dr	Davis	CA 95616	(312) 235-0176	Marsh, Alan A.	Wilkinson, Priscilla L.	\$ 1,293,519	31%	Green
37	IN-N-OUT Burger - Diamond Bar	21133 Golden Springs Dr	Diamond Bar	CA 91789	(130) 498-6205	Hewitt, Melanie E.	Barton, Elsie A.	\$ 1,329,383	51%	Red
38	IN-N-OUT Burger - Draper	12191 South Factory Outlet Dr	Downey	CA 90241	(322) 980-0719	Branch, Marianne R.	Walton, Beth A.	\$ 1,437,735	42%	Green
39	IN-N-OUT Burger - El Cajon	1541 N Mainland Ave	El Cajon	CA 92020	(361) 689-2756	Walton, Natalie A.	Hall, Enca A.	\$ 1,143,421	80%	Red
40	IN-N-OUT Burger - El Cajon	21133 Golden Springs Dr	El Cajon	CA 92020	(361) 811-8906	O'Brien, Caroline L.	Ross, Douglas O.	\$ 587,822	-82%	Green
41	IN-N-OUT Burger - El Cajon	8767 Firestone Blvd	El Cajon	CA 92020	(361) 811-8906	Walton, Natalie A.	Chung, Donald H.	\$ 226,468	58%	Yellow
42	IN-N-OUT Burger - El Cajon	12191 South Factory Outlet Dr	El Cajon	CA 92020	(361) 811-8906	O'Brien, Caroline L.	Bender, Katherine E.	\$ 1,011,008	-42%	Yellow

图3-2 一个便于进行数据处理的例子

在本章中，我们尝试呈现一个“数据处理工具箱”，当你从事一个机器学习项目时，可以从中获得满足你需求的工具。在另一方面，你可能需要额外的变换工具，在这种情况下，可以以这里给出的工具作为基础，设计你需要的工具。随着时间的流逝，你可能开发出一个大型的数据处理工具库，有需要的时候可以从中挑选。下面是我们在本章中会介绍到的常用操作：

- 数据采样;
- 变量名转换;
- 创建新变量;
- 数值离散化;
- 日期处理;
- 将类变量二值化;
- 合并数据集;
- 排列数据集;
- 重塑数据集;
- 使用dplyr进行数据操作;
- 处理缺失数据;
- 特征缩放;
- 降维。

好在，R有一长串的嵌入式工具和R包来满足数据处理的需求。

plyr、dplyr、reshape2、stringr和lubridate这些包迟早都会用到，我们将
在后面的小节中看到这些包的使用案例。

3.1 特征工程

特征工程（feature engineering）用于识别在模型或者机器学习算法中使用的数据子集或者转换后的数据。不同的学术分支使用不同的表述来描述同一个东西。当描述输入到模型的数据子集时，统计学家使用“解释变量”“因变量”、或者“预测因子”。在另一方面，数据科学家使用“特征”。在原始数据集中，可能有很多冗余或者相关变量，你不会想将所有这些变量都输入到模型中的。类似地，你可能希望通过转换现有的变量来创造一些新变量，例如，在将数据集导入模型之前，可以把一个连续变量转换成二进制变量。

特征工程是机器学习中最重要但也是最被低估的步骤。一般而言，它是建立统计模型和算法的重要部分。有了数据，并不意味着要把全部数据使用在模型中。人们常说，优秀、严密的特征比算法更有价值。因为数据处理工作在机器学习过程的早期进行，涉及处理原始数据，所以现在是关注特征工程的黄金时间。

优秀的特征工程包含两个过程。首先，你必须了解要解决的问题的性质，以及它们将如何影响在用算法的优点和局限性。第二，进行试验，测试你的期望，并发现哪些部分的工作是真正有效的，哪些部分是不必执行的。这些工作可以反复进行：你对问题的理解可以自上而下地促进试验，然后在这些试验中得到的信息可以自下而上地帮助你更好地理解问题。对问题更深入的理解可以驱动更多的试验。通常情况下，特征工程是一个与探索性数据分析妥协的过程，我们将在第4章讨论探索性数据分析。特征工程是利用关于数据的知识来选择创建特征，使得机器学习算法有更好的效果。

特征工程的处理过程既是一门科学，也是一门艺术。在这一过程中，如果有领域专家的帮助固然是很好的，但是使用你的想象力也同样能产生不错的效果。问题在于，一般情况下数据太多了。随着当代大数据技术的发展，我们处于一个产生大量特征的状态之下。但过去并非总是如此。仅仅在几年前，一项典型的调查只需要被访者提供针对20个问题的回答，而到今天，可能高达成百上千个数据项。你需要回答的问题是，这些特征中有多少是噪音？在大数据的环境下，你获得的一些信息

可能是无用的。

虽然特征工程是研究过程中不间断的一个主题，让我们来看一个简便的方法，用这个方法可以将特征工程和数据处理工作结合起来，也可以和后面章节会提到的监督学习主题结合起来。在完成数据处理过程之后，我们可以说你有了一组特征。你可以从一个没有特征的回归模型开始，然后根据哪个特征最能提升模型的效果，每次加入一个特征。你基本上只需要使用每一个预测因子建立所有可能的回归模型，然后从中挑选效果最好的模型。现在，将最好的预测因子和第二好的预测因子组合在一起，尝试建立所有可能的模型，并从中选择出最好的模型。然后每次加入一个特征，直到你的模型的效果无法再提升、反而有下降为止。

另一个方法涉及特征的反向淘汰。在这里，你从一个包含所有特征的回归模型开始，然后根据移除哪个特征对模型有最大的提高，每次移除一个特征，直到移除特征会使预测模型效果变差为止。

在我们继续往前走之前，给你提个醒，如果你有可以利用的领域专家的资源，在你从事漫长的特征工程过程前，挖掘他们的关于数据的业务和认知的专业知识。

3.2 数据管道

在机器学习项目中，保持数据处理代码整洁性，具备良好的备注习惯是很重要的。原因在于，你一定会回头查看将数据集转换成适用于机器学习算法过程的所有步骤。我建议把所有的代码片段都保存在一个主要的R脚本文件中或者工作目录下。这些脚本将成为数据管道的基础。

数据管道（data pipeline）是数据转换任务中决定性的一部分，需要获取原始数据集，然后把它们转换成适合机器学习的变换后数据集。通常情况下，你需要一次又一次地运行数据管道，用最新的数据来更新你的机器学习数据集。在后面章节我们会看到，机器学习其中的一个特征是需要用最新数据重新训练算法，从而提升算法的预测能力。

在接下去的小节中，我将展示一个数据处理技巧工具包，可以用于处理你的数据。这些讨论的顺序大致是它们应该实施的顺序，但这个顺序并不是固定不变的。具体的顺序很大程度上取决于预处理的数据集和使用的机器学习算法的需求。你选择的数据处理步骤将会成为机器学习数据管道的初级阶段。

3.3 数据采样

在某些情况下，你获得用于解决机器学习问题的数据量很大，在R环境下尝试处理它们可能出现问题，因为R是基于内存的，并且你能使用的处理能力也是有限的。在这些情况下，对数据集进行取样，减少处理数据的大小更合适。

在很多采样模型中，随机取样是至今为止应用最普遍的模型，它给每个记录分配一个“被选中”的概率。采样可以是不可重复的（每条记录最多被抽取一次），或者是可重复的（同一条记录可以被多次抽取）。让我们使用iris数据集作为数据抽样的例子。用sample()函数来随机选择10行可重复的记录。生成的sample_index是一个整型向量，包含了指向iris数据集中被选中的记录的索引。我们使用这个抽样索引来创建一个新的抽样数据集sample_set，如下面所示。

```
> sample_index <- sample(1:nrow(iris), 10, replace=T)
> sample_index
[1] 25 134 116 140 91 98 18 49 17 73
> sample_set <- iris[sample_index, ]
> sample_set
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
25      4.8       3.4       1.9       0.2    setosa
134     6.3       2.8       5.1       1.5  virginica
116     6.4       3.2       5.3       2.3  virginica
140     6.9       3.1       5.4       2.1  virginica
91      5.5       2.6       4.4       1.2 versicolor
98      6.2       2.9       4.3       1.3 versicolor
18      5.1       3.5       1.4       0.3    setosa
49      5.3       3.7       1.5       0.2    setosa
17      5.4       3.9       1.3       0.4    setosa
73      6.3       2.5       4.9       1.5 versicolor
```

3.4 修正变量名

从组织中的不同数据源获得数据集之后，你经常会发现文件中的特征变量名前后不一致，使用不便，有些还约定了不恰当的字段名。很多时候，不合适的变量名来自于和现代软件技术不相称的旧系统。数据科学家的工作就是提高可用性，这意味着需要对变量进行重新命名和改造。这里有一些在进行变量重命名时会遇到的潜在问题，不过，你也可能遇到很多其他问题。你需要做的就是编写代码来执行适合你的命名约定。

- employeeSalary: 大小写混合的名称，一些从业者认为最好都小写。
- office.1: 特殊字符嵌入到名称中，例如R社区中经常使用的句点“.”。可以选择剔除所有的特殊字符。
- Country of Origin: R不支持名称里带有空格，所以所有嵌入的空格都必须删除。
- zip_code: 在其他编程语言中，下划线很常见。同样，也有一些R从业者会使用它们。

现在让我们来看一个使用R编程纠正变量名的例子。首先，我们要创建一个空数据框，也就是说，一个有列名但没有实际数据的数据框。对一些R用户来说，例子中的一些列名需要进行修正，例如，使用tolower()函数将CrossStreet转换成crossstreet。

```
> df <- data.frame("Address 1"=character(0), direction=
  character(0), street=character(0), CrossStreet=character(0),
  intersection=character(0), Location.1=character(0))
> names(df)
[1]"Address.1" "direction" "street" "CrossStreet" "intersection"
[6]"Location.1"
> names(df) <- tolower(names(df)) # convert to all lower case
> names(df)
[1]"address.1" "direction" "street" "crossstreet" "intersection"
[6]"location.1"
```

在下一个代码段中，我们使用strsplit()函数来遍历所有的变量名，

然后把所有含有嵌入式句点的名字都分离出来。strsplit()输出的是一个R列表对象。这个基础R函数的工作方式是：当它发现一个字符串中有句点时，就把字符串拆成一个子列表，其中第1个元素是句点前面的字符串，第2个元素是句点后面的字符串。

```
> splitnames <- strsplit(names(df), "\\.")
> class(splitnames)
[1] "list"
> length(splitnames)
[1] 6
> splitnames[2]      # Single list element
[[1]]
[1] "direction"
> splitnames[6]      # Sub-list element
[[1]]
[1] "location" "1"
> splitnames[[6]][1]  # "location"
[1] "location"
> splitnames[[6]][2]  # "1"
[1] "1"
```

在这个例子的最后一部分，我们将写一个特殊的R函数firstelement()，它能遍历整个splitnames列表对象，并选择每个子列表中的第一部分（别忘了，我们的测试数据框中有两个变量名包含了句点字符“.”），然后将变量名设置成合适的形式。请注意，firstelement()将被不断重复调用，直到splitnames从每一个元素中都提取出子列表的第一个元素。

```
> firstelement <- function(x){x[1]}
names(df) <- sapply(splitnames, firstelement)
```

3.5 创建新变量

在数据处理过程中，你常常会查看数据集中现有的特征，并决定你是否需要一个新的变量，新变量通常是衍生于现有变量的计算变量。在R环境中，往现有的数据框中添加新变量是很容易的。为了说明这个原理，我们将使用airquality数据集，添加一个新字段ozoneRanges。这个字段将采集目前的定量变量Ozone，然后基于它的数值计算出一个对应的范围。为此，我们使用cut()函数，它能协助我们根据Ozone值的范围进行分类。比如说，Ozone值为41的时候处于(25, 50]范围。

```
> airquality$Ozone[1:10] # First 10 rows of Ozone values
[1] 41 36 12 18 NA 28 23 19 8 NA
> ozoneRanges <- cut(airquality$Ozone, seq(0, 200, by=25))
> ozoneRanges[1:10] # Show calculated ranges of Ozone values
[1] (25,50] (25,50] (0,25] (0,25] <NA> (25,50] (0,25] (0,25] (0,25]
[10] <NA>
8 Levels: (0,25] (25,50] (50,75] (75,100] (100,125] (125,150] ... (175,200]
> class(ozoneRanges) # Note, this is a factor!
[1] "factor"
> table(ozoneRanges, useNA="ifany")
ozoneRanges
(0,25] (25,50] (50,75] (75,100] (100,125] (125,150] (150,175]
50      32      12      15       5       1       1
(175,200] <NA>
0        37
> airquality$ozoneRanges <- ozoneRanges
> head(airquality)
   Ozone Solar.R Wind Temp Month Day ozoneRanges
1    41     190  7.4   67     5    1   (25,50]
2    36     118  8.0   72     5    2   (25,50]
3    12     149 12.6   74     5    3   (0,25]
4    18     313 11.5   62     5    4   (0,25]
5    NA      NA 14.3   56     5    5   <NA>
6    28      NA 14.9   66     5    6   (25,50]
```

添加新变量的另一个例子是，衍生特征将计算两个变量的比值或者归纳一些特征形成粗粒度的特征，例如，将地理位置信息转换成邮政编码，或是将年龄变量转换成年龄段。

3.6 数值离散化

有时，你获得的数据集中有一个数值变量，对于机器学习来说，离散的“范围”值比连续值处理起来更方便。例如，你有一个salary字段，比起实际的薪水数额来说，六个数值范围将会更有用——比方说，\$0～\$10000、\$10001～\$25000、\$25001～\$50000、\$50001～\$75000、\$75001～\$100000和\$100001～\$150000。所以说，如果在一个具体的记录中，薪水值是\$56000，那么它的离散形式是在\$50001～\$75000范围内。将一个连续值变量划分到区间中，创建一个新因素变量的范围值，并将变量值分配到相应的范围中，这整个过程叫做“离散化”。在R中使用cut()函数，可以很方便地进行这些操作。

在下面的例子中，我们将iris数据集中的Sepal.Length变量进行离散化。我们随意地将数值切分到10个桶内。为此，我们使用seq()函数，基于这个变量的最大值、最小值和桶的数目来创建合适的分割点。然后我们使用cut()函数来创建一个因素对象，其中包含对应每个Sepal.Length值的范围桶。最后，为了说明这一点，将这个因子与初始的Sepal.Length值结合到一起，进入新的数据框newiris。

```
> data(iris)
> buckets <- 10
> maxSepLen <- max(iris$Sepal.Length)
> minSepLen <- min(iris$Sepal.Length)
> cutPoints <- seq(minSepLen, maxSepLen, by=(maxSepLen-minSepLen) /buckets)
[1] 4.30 4.66 5.02 5.38 5.74 6.10 6.46 6.82 7.18 7.54 7.90
> cutSepLen <- cut(iris$Sepal.Length,breaks=cutPoints,
+ include.lowest=TRUE)
> newiris <- data.frame(contSepLen=iris$Sepal.Length,
+ discSepLen=cutSepLen)
> head(newiris) # Display first 6 records of data frame
  contSepLen discSepLen
1      5.1     (5.02,5.38]
2      4.9     (4.66,5.02]
3      4.7     (4.66,5.02]
4      4.6     (4.3,4.66]
5      5.0     (4.66,5.02]
6      5.4     (5.38,5.74]
```

3.7 日期处理

日期值展现了在R中进行数据处理时特别棘手的问题。一部分原因在于，在数据集中，日期和时间有那么多的表现形式。同时也因为处理日期时用的R语法使用不便、容易混淆。这个主题可以讲一整章，所以在本节中，我们只能介绍几种常用的情况和对应的处理方法。这是数据处理中一个熟能生巧的领域，所以在读完本书之后，你应该在熟悉的领域中发掘几个数据集，并开发程序来处理你遇到的日期和时间数值。

日期有很多不同的形式，这使得识别和解析日期成为一项挑战。当进行数据处理时，你必须清楚地知道数据集中的日期值的格式。一旦把数据导入R中，你就应该有一个明确的想法：让日期以什么形式展示。举个例子，你需要完整的日期（MM/DD/YYYY），还是只需要年月（MMYYYY）？

基础R中用于数据处理的机制，用起来比较困难，但好在有一个非常方便的lubridate包可以更简便地处理日期。为了数据处理方便，我强烈建议你熟悉这个包。在本节中，我们用到的都是lubridate包。这个包里面有一个叫作lakers的数据集，在这里我们用它来做示例。这个数据集包含了2008-2009赛季洛杉矶湖人的每场篮球比赛的详细数据。其中有两个我们感兴趣的字段：lakers\$date（比赛日期，这是一个整型数值）和lakers\$time（比赛中的赛钟时间，这是一个字符值字段）。

在下面的代码中，我们将加载lubridate包和lakers数据集。因为我们要对date和time变量做一些数据处理工作，将数据框复制一份，并命名为df。使用str(df\$date)之后，我们可以知道这个变量是整型，数据的格式是年、月、日。这类变量十分典型，日期和时间数值没有以R日期或时间对象存储，而是以整型或是字符型存在。我们的任务就是转换这些数值。

```
> library(lubridate)
> data(lakers)
> df <- lakers # Make a copy for testing
> str(df$date)
int [1:34624] 20081028 20081028 20081028 20081028 20081028 20081028 20081028 2008:
```

我们想做的一部分数据处理工作是把date和time变量结合起来，成为一个R日期对象，里面也包含了时间。你可能希望这么做，举个例子，你可以用这个新变量计算出一场比赛到另一场比赛的时间间隔。在进行其他数据处理任务时，我们可以先用单个观测值进行实验，并把变量值存储在playdate和playtime中。然后我们用paste()把它们合在一起，创建新变量playdatetime。注意到，在paste()函数中，playdate的整型数值转换成（另一种类型）字符型，然后和playtime进行合并。最后，我们使用lubridate包中的parse_date_time()来生成一个R日期对象。同时我们也展现了新的合并后的日期/时间值和它的类型。POSIXct是R中用来表示日历日期和时间的类型。

```
> playdate <- df$date[1] # Integer  
> playtime <- df$time[1] # Character  
> playdatetime <- paste(playdate, playtime)  
> playdatetime <- parse_date_time(playdatetime, "%y-%m-%d %H.%M")  
> playdatetime  
[1] "2008-10-28 12:00:00 UTC"  
> class(playdatetime)  
[1] "POSIXct" "POSIXt"
```

我们能进行的另一个操作是将整型的日期变量替换成R中的日期/时间等价对象。为此，我们使用lubridate包中的ymd()函数。请注意，lubridate提供了一些其他函数，例如mdy()和dmy()，你可以根据数据集中日期值的具体格式来选用。

```
> df$date <- ymd(df$date)  
> str(df$date)  
POSIXct[1:34624], format: "2008-10-28" "2008-10-28" "2008-10-28" "2008-10-  
28" "2008-10-28" ...  
> class(df$date)  
[1] "POSIXct" "POSIXt"
```

我们可以做的另一件事情是在数据框中创建新的PlayDateTime列。为了达到这个目标，我们也可以使用parse_date_time()函数，但是这次要对数据集中所有观测值进行操作。

```
> df$PlayDateTime <- parse_date_time(paste(df$date, df$time), "%y-%m-%d %H.%M")  
> str(df$PlayDateTime)  
POSIXct[1:34624], format: "2008-10-28 12:00:00" "2008-10-28 11:39:00" "2008-10-  
28 11:37:00" ...
```

上面的例子仅仅是日期/时间数据处理的一些皮毛。其他可能需要

完成的任务包括：提取日期/时间，包括年、月、秒等；切换时区；将使用夏令时和不使用夏令时地区的时间进行比较；进行日期和时间的计算；处理闰年等等。一般来说，我们的目的是分析（识别）不标准或不便于使用的日期和时间值，然后把它们转换成R中的日期/时间对象。

3.8 将类变量二值化

当使用具体的机器学习算法时，把一个类变量（R中叫做因子）替换成多个二进制变量会方便得多。你可以完成这个操作以供二值分类算法使用。iris数据集的Species变量是个很好的例子。这个因子变量有三个“级别”：setosa、versicolor和virginica。我们可以创建3个新的二进制变量，每个都代表着每条Species变量值的TRUE或是FALSE的状态。

```
> species_cat <- levels(iris$Species)
> species_cat
[1] "setosa"    "versicolor" "virginica"
```

为了得到预期的结果，我们需要写一个新函数binarySpecies()，它能被基本R的sapply()函数重复调用，创建包含代表Species的二进制值的矩阵。下一步，我们根据种类名称命名新的列。最后，我们把新列加入iris数据框的副本bin_iris中。现在我们就可以使用这个数据框运行合适的机器学习算法了。

```
> binarySpecies <- function(c) {return(iris$Species == c)}
> newVars <- sapply(species_cat, binarySpecies)
> newVars[50:55,]
      setosa   versicolor   virginica
[1,] TRUE     FALSE       FALSE
[2,] FALSE    TRUE        FALSE
[3,] FALSE    TRUE        FALSE
[4,] FALSE    TRUE        FALSE
[5,] FALSE    TRUE        FALSE
[6,] FALSE    TRUE        FALSE
> colnames(newVars) <- species_cat
> bin_matrix <- cbind(iris[,c('Species')], newVars)
> bin_matrix[50:55,]
      setosa   versicolor   virginica
[1,] 1       1           0
[2,] 2       0           1
[3,] 2       0           1
[4,] 2       0           1
[5,] 2       0           1
[6,] 2       0           1
> bin_iris <- iris
> bin_iris$setosa <- bin_matrix[,2]
> bin_iris$versicolor <- bin_matrix[,3]
> bin_iris$virginica <- bin_matrix[,4]
> names(bin_iris)
[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"
[6] "setosa"       "versicolor" "virginica"
```

3.9 合并数据集

当获得两个或多个结构相似的数据集时，你可能需要把它们合并到一起，生成用于机器学习的数据集。数据处理阶段是合并数据集，生成包含所有有效记录的新数据集的好时机。R中有非常有用的merge()函数，可以基于公共变量将数据框合并到一起。如果你熟悉SQL，你可能已经猜到merge()和连接操作非常相似。实际上确实如此，不同的地方在于merge()不但可以执行内部和外部的连接，而且可以进行左连接和右连接。

merge()函数允许4种合并数据的方式：

- 内连接 (inner join)：只保留两个数据框中一致的行，指定参数 all=FALSE。
- 外连接 (outer join)：保留数据框中所有的行，指定all=TRUE。
- 左外连接 (left outer join)：包含数据框x的所有行加上数据框y中能匹配到数据框x的所有数据，指定all.x=TRUE。
- 右外连接 (right outer join)：包含数据框x的能匹配到数据框y的所有数据，加上数据框y中所有的行，指定all.y=TRUE。

在下面的例子中，我们在一个很简单的数据框中用merge()函数对4种连接进行演示。

```
> df1 = data.frame(CustId=c(1:6),Product=c(rep("Mouse",3), rep("Keyboard",3)))
> df2 = data.frame(CustId=c(2,4,6),State=c(rep("California", 2),rep("Oregon",1)))
> # Outer join
> merge(x = df1, y = df2, by = "CustId", all = TRUE)
  CustId Product      State
1     1    Mouse      <NA>
2     2    Mouse   California
3     3    Mouse      <NA>
4     4  Keyboard   California
5     5  Keyboard      <NA>
6     6  Keyboard    Oregon
> # Left outer join
> merge(x = df1, y = df2, by = "CustId", all.x=TRUE)
  CustId Product      State
1     1    Mouse      <NA>
2     2    Mouse   California
```

```
3   3       Mouse      <NA>
4   4       Keyboard   California
5   5       Keyboard   <NA>
6   6       Keyboard   Oregon
> # Right outer join
> merge(x = df1, y = df2, by = "CustId", all.y=TRUE)
  CustId  Product    State
1     2       Mouse   California
2     4       Keyboard California
3     6       Keyboard Oregon
> # Inner join
> merge(x = df1, y = df2, by = "CustId", all=FALSE)
  CustId  Product    State
1     2       Mouse   California
2     4       Keyboard California
3     6       Keyboard Oregon
```

3.10 排列数据集

在为一个新的数据科学项目评估数据集时，你经常会发现排完序的数据对解决机器学习问题很有帮助。假设你在查看全国各地的房屋价格这一不动产数据。在这种情况下，通过州、地区、城市对数据进行地理位置上的排序，比随机排序的数据浏览起来更直观。通常情况下，记录的顺序都是最初加入到数据集的顺序。对数据框内容进行排序更便于浏览。R有order()函数，可以根据一个或多个变量对数据框进行排序。

让我们看看使用ToothGrowth数据集进行排序的例子。数据框的前几行是这样的：

```
> data(ToothGrowth)
> head(ToothGrowth)
  len supp dose
1 4.2  VC  0.5
2 11.5 VC  0.5
3  7.3 VC  0.5
4  5.8 VC  0.5
5  6.4 VC  0.5
6 10.0 VC  0.5
```

这个数据集由60条观测值和3个数值变量len、supp和dose组成。你可以用ToothGrowth来更深入了解这个数据集。使用下面的R语句，我们可以根据len对数据集进行排序，然后展示排序后新数据框的前10条记录。我们可以注意到，现在len列是有顺序的。

```
> sortedData <- ToothGrowth[order(ToothGrowth$len),]
> sortedData[1:10,]
  len supp dose
1 4.2  VC  0.5
9 5.2  VC  0.5
4 5.8  VC  0.5
5 6.4  VC  0.5
10 7.0 VC  0.5
3 7.3  VC  0.5
37 8.2 OJ  0.5
38 9.4 OJ  0.5
34 9.7 OJ  0.5
40 9.7 OJ  0.5
```

在下一个例子中，我们根据两个变量supp和len对ToothGrowth数据集进行排序。你可以把supp当做主要排序关键字，len是次要排序关键字。下面是排完序的数据框中前10行记录。

```
> sortedData <- ToothGrowth[order(ToothGrowth$supp,
ToothGrowth$len),]
> sortedData[1:10,]
   len   supp   dose
37  8.2   OJ    0.5
38  9.4   OJ    0.5
34  9.7   OJ    0.5
40  9.7   OJ    0.5
36 10.0   OJ    0.5
35 14.5   OJ    0.5
49 14.5   OJ    1.0
31 15.2   OJ    0.5
39 16.5   OJ    0.5
33 17.6   OJ    0.5
```

3.11 重塑数据集

很多时候，你获得的用于机器学习算法的数据集是“畸形”的，也就是说，数据都有，但是格式或者结构不便于使用。当这种情况发生时，你的工作就是重塑数据集，使得更便于执行后面的建模。这时，你可以用reshape2包中的melt()函数，这个函数是重塑数据集的通用工具。很多数据集都可以当做例子，在这里，我们仅使用一个案例来说明这一过程。

我们先创建一个测试数据框，假设它是从外部数据源获得的。这个数据集中包含了一个班级中3个学生的两次小测成绩。这个数据集的问题是，在同一行中，每个学生有两个成绩。对于机器学习来说，每行只有一个成绩处理起来会方便得多（行的数目会变为两倍）。

```
> library(reshape2)
> misShaped <- as.data.frame(matrix(c(NA,5,1,4,2,3), byrow=TRUE, nrow=3))
> names(misShaped) <- c("Quiz 1", "Quiz 2")
> misShaped$student <- c("Ellen", "Catherine", "Stephen")
> misShaped

  Quiz 1 Quiz 2 student
1     NA      5    Ellen
2      1      4 Catherine
3      2      3   Stephen
```

为了解决这个构造问题，我们使用melt()函数对数据集进行重塑。在这里，我们将数据框misShaped和用于存储数据的变量名（在这个例子中是score）传递给函数。就像下面展示的那样，melt()输出的正是我们需要的结构。

```
> melt(misShaped, id.vars="student", variable.name="Quiz", value.name="score")
  student     Quiz score
1  Ellen    Quiz 1   NA
2 Catherine  Quiz 1    1
3  Stephen   Quiz 1    2
4  Ellen    Quiz 2    5
5 Catherine Quiz 2    4
6  Stephen   Quiz 2    3
```

3.12 使用dplyr进行数据操作

dplyr包是数据处理过程中的一个有价值的工具，它提供了在R中对列表数据进行过滤、查询、重建和集合的方法。dplyr是广泛使用的plyr包的升级版，这两个包都是著名的R开发者Hadley Wickham开发的。关键改进中很重要的一点是，dplyr只应用于数据框。它介绍了一种“操作数据的语法”，允许你把%.%操作符和操作连贯起来。同时，这些操作的速度比plyr或标准R有了很大的提升。在本节中，我们可以看到一些dplyr协助处理数据处理过程的例子。可以注意到，dplyr提供了一些新方法来进行数据操纵，用以替换本章和第2章介绍的技术：创建新变量、将数据集进行排序、SQL以及R中的SQL等效表达。鉴于dplyr是单个R包，具有一致的语法，你可能需要钻研dplyr以便在以后的数据处理工作中进行选择。

在本节中，我们将演示基本的dplyr数据操作，它们都在单个表格中使用：filter()、arrange()、select()和mutate()。dplyr还提供了很多其他功能，我们鼓励你自行探索。我们从安装dplyr包开始，然后加载库和数据集。同时也会使用到dplyr中的tbl_df()函数，它能优化数据框以便于打印（使输出最小化）。

```
> install.packages("dplyr")
> library(dplyr)
> data(ToothGrowth)
> ToothGrowth_df <- tbl_df(ToothGrowth) # dplyr function
```

下一步，我们将给出使用filter()函数的一个实例，返回值是由数据集中的一些行组成的子表。在这个例子中，我们创建了一个子表，包含了len值为11.2且supp为“VC”的行。你可以用这个工具筛选出特定的数据组成。从某些方面来说，filter()和SQL SELECT在功能上差不多，它的参数和SQL WHERE从句也很类似。

```
> filter(ToothGrowth_df, len==11.2 & supp=="VC")
  len  supp  dose
1 11.2  VC    0.5
2 11.2  VC    0.5
```

现在，我们来看看arrange()函数的例子，它用于对数据集中的行进行重新排序。在这个案例中，我们对ToothGrowth_df数据框进行重新排序，先根据supp进行排序，然后按len进行降序排列。这个顺序如下面所示。当你对数据更熟悉时，在探索性数据分析阶段可能会想对数据集中的行进行重新排序。对行进行排序能帮助你确定某个变量特殊值的数目、某个变量的数值范围、某个变量的相对数量值等等。你会发现，arrange()和本章之前提到的基础R中的order()函数在功能上类似。

```
> arrange(ToothGrowth_df, supp, desc(len))
Source: local data frame [60 x 3]
  len   supp   dose
1 30.9   OJ    2.0
2 29.4   OJ    2.0
3 27.3   OJ    1.0
4 27.3   OJ    2.0
5 26.4   OJ    1.0
6 26.4   OJ    2.0
7 26.4   OJ    2.0
8 25.8   OJ    1.0
9 25.5   OJ    2.0
10 25.2  OJ    1.0
```

现在让我们看看select()函数的例子，这个函数的返回值是数据集中列的子集。在这个例子中，我们删除了len变量，返回只包含dose和supp列的子集。通常情况下，你会将select()返回的值分配到另一个数据框中。在特征工程练习过程中，如果想从数据集中剔除一些变量，就可以使用select()函数。你会发现，select()在功能上和SQL SELECT语句类似，但它有一个缩写的字段列表，而不是用星号表示的所有字段列表。

```
> select(ToothGrowth_df, dose, supp)
Source: local data frame [60 x 2]
  dose supp
1  0.5  VC
2  0.5  VC
3  0.5  VC
4  0.5  VC
5  0.5  VC
6  0.5  VC
7  0.5  VC
8  0.5  VC
9  0.5  VC
10 0.5  VC
```

最后，我们可以使用mutate()函数在数据集中创建新的列。在下面的例子中，我们创建了新的一列supp_num，它是supp的数值等效表达。

获得mutate()的输出后，将这个包含新列的数据集重新分配回ToothGrowth_df中。最后一步是在plot()函数中使用新的列，如图3-3所示，对数据集进行可视化。supp_num的不同数值（1代表OJ，2代表VC）在图中以不同的符号表示（值为1的标为圆圈，值为2的标为三角）。在实际应用中，你可以自由选择mutate()或是本章前面讨论的创建新变量的方法。

```
> ToothGrowth_df <- mutate(ToothGrowth_df,  
+ supp_num=as.numeric(supp))  
> attach(ToothGrowth_df)  
> plot(len~dose,pch=supp_num)
```

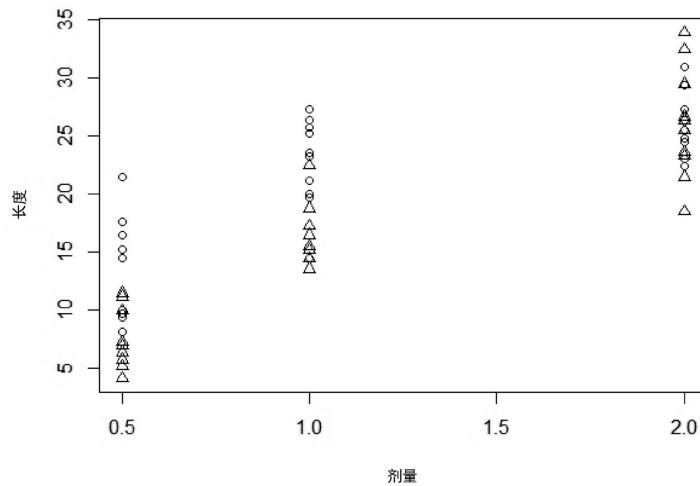


图3-3 使用新变量supp_num后Tooth Growth数据集的散点图

3.13 处理缺失数据

作为一个用企业数据集进行工作的数据科学家，你会对收到数据的状态感到惊奇（有时候是挫败感）。多年来贫乏的数据管理导致企业数据通常是很脏的。一个反复出现的问题是数据缺失。如果某个数据字段（特征）丢失或者错误，那么不完整的记录就会很常见。我们需要验证每一条记录，确保它包含了同样数目的字段，并且每个字段的类型都如我们预期的那样。同时，我们需要一个处理缺失数据的计划。如果一条记录是不完整的，我们可以丢弃整条记录，或者基于其他记录的数据推断出缺失的字段值。一种常用的方法是用其他数据值的平均值或者中位数填补缺失的数据，这称为输入数据值（imputing data value）。

为了演示一种处理缺失数据的方法，我们将使用iris数据集，并有选择地将一些数据值置为NA（R中表示缺失数据的方式）。然后我们使用e1071包中的impute()函数将缺失值置为这列的平均值。注意一下，impute()返回一个矩阵，而不是一个数据框。所以我们必须多加一步，将结果中的R对象iris_repaired转换回数据框的形式。另外，e1071中还有很多能为数据科学家带来便利的有用功能。

```
> library(e1071)
> iris_missing_data <- iris
> iris_missing_data[5,1] <- NA
> iris_missing_data[7,3] <- NA
> iris_missing_data[10,4] <- NA
> iris_missing_data[1:10, -5]
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1      5.1         3.5       1.4        0.2
2      4.9         3.0       1.4        0.2
3      4.7         3.2       1.3        0.2
4      4.6         3.1       1.5        0.2
5      NA          3.6       1.4        0.2
6      5.4         3.9       1.7        0.4
7      4.6         3.4        NA        0.3
8      5.0         3.4       1.5        0.2
9      4.4         2.9       1.4        0.2
10     4.9         3.1       1.5        NA
> iris_repaired <- impute(iris_missing_data[,1:4], what='mean')
> iris_repaired <- data.frame(iris_repaired)
> iris_repaired[1:10, -5]
  Sepal.Length Sepal.Width Petal.Length Petal.Width
[1,] 5.100000   3.5       1.400000  0.200000
[2,] 4.900000   3.0       1.400000  0.200000
[3,] 4.700000   3.2       1.300000  0.200000
[4,] 4.600000   3.1       1.500000  0.200000
[5,] 5.848993   3.6       1.400000  0.200000
```

```
[6,] 5.400000 3.9 1.700000 0.400000
[7,] 4.600000 3.4 3.773826 0.300000
[8,] 5.000000 3.4 1.500000 0.200000
[9,] 4.400000 2.9 1.400000 0.200000
[10,] 4.900000 3.1 1.500000 1.206711
```

现在，让我们考虑这样的情况：你希望丢弃那些在一个或多个字段上有缺失值的记录。如果要移除的记录数目比记录总数少得多的话，这种方法是十分有效的。下面的例子展示了用`complete.cases()`函数执行移除操作的一种方法。

```
> df <- iris_missing_data # Make a copy of the data frame
> nrow(df) # 150 rows including 3 rows with missing data
[1] 150

> iris_trimmed <- df[complete.cases(df[,1:4]),]
> iris_trimmed <- na.omit(df) # This works too
> nrow(iris_trimmed)
[1] 147
```

还有另一种有效的方法，首先确定多少条观测记录有缺失值，如果合适的话，再删除它们。

```
> df.has.na <- apply(df, 1, function(x){any(is.na(x))})
> sum(df.has.na) # Show a count of observations with NA
[1] 3
> iris_trimmed <- df[!df.has.na,] # Remove the observations
```

3.14 特征缩放

在你的特征列表中，数值型数据在数据处理过程初期特别明显的一个特性是：不同的特征变量，数值的范围很可能差别很大。例如，一个家庭中卧室的数目范围可能是1到5，然而，家庭的建筑面积可能是1 000到3 500平方英尺。不同的量级是由于不同的计量单位造成的。特征缩放能让我们在同一尺度下比较数值特征；将数据减去平均值，然后除以标准差，使数据标准化。我们的目标是生成一个介于-1和1之间的共同数值范围。好在，在基础R的stats包中有scale()函数，可以用于缩放数值。下面的例子使用了iris数据集，然后用scale()函数对所有的数值变量进行缩放。

因为机器学习算法的一些运行方式，特征缩放变得很重要。例如，很多分级器要计算两个数据点之间的距离。如果其中一个特征的数值范围很大，那么距离就会取决于这个特征。所以，所有特征的范围应该进行标准化，这样每个特征才能对最后的距离有比例合适的贡献。

```
> head(iris) # Note the range of each feature
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1   5.1         3.5       1.4        0.2    setosa
2   4.9         3.0       1.4        0.2    setosa
3   4.7         3.2       1.3        0.2    setosa
4   4.6         3.1       1.5        0.2    setosa
5   5.0         3.6       1.4        0.2    setosa
6   5.4         3.9       1.7        0.4    setosa
> scaleiris <- scale(iris[, 1:4])
> head(scaleiris) # Now view the scaled features
  Sepal.Length Sepal.Width Petal.Length Petal.Width
[1,] -0.8976739  1.01560199 -1.335752 -1.311052
[2,] -1.1392005 -0.13153881 -1.335752 -1.311052
[3,] -1.3807271  0.32731751 -1.392399 -1.311052
[4,] -1.5014904  0.09788935 -1.279104 -1.311052
[5,] -1.0184372  1.24503015 -1.335752 -1.311052
[6,] -0.5353840  1.93331463 -1.165809 -1.048667
```

3.15 降维

试图将高维模型（换句话说，有大量特征的模型）用于机器学习算法，这常常称为“维数灾难”，原因有很多：

- 当处理很多特征时，性能可能会降低。
- 有很多特征的模型可能不适合在你使用的计算机中运行，因为开源的R受可用内存数量的约束。
- 在处理超过三个特征（三维图表）时，常用的可视化工具很难使用。

有两种常用的策略可以用于减少模型中的特征数目。第一，你可以移除所有不相关的特征。从检查模型中的所有特征开始，思考每一个特征是否真的能对机器学习工作带来帮助。举个例子，如果你的数据集中包含顾客数量字段，那么保留这个特征就是没有意义的，因为本质上它是没有价值的。通常的做法是尝试特征子集的不同组合，观察哪个组合能产生最好的效果（使用测试数据集时能预测得多准确）。就像3.1节提到的那样，从少量特征开始，测试它们的预测能力，然后加入更多的特征并重复这一过程。你也可以尝试反向方法——从一整套特征开始，然后在测试预测能力的过程中移除一些特征。

一般来说，由于上面提到的3个原因，在模型中加入不相关的特征会影响机器学习解决方案的整体性能，但它不会在很大程度上影响预测准确性。与特征冗余相比，这是次要问题。特征冗余会使多余信息获得不平等的权重。所以移除冗余特征处于更高的优先级。

另一个更精细的用于降维的方法涉及一个过程，叫做主成分分析（PCA），它基于线性代数中的一个理论：奇异值分解（SVD）。同时，PCA也是非监督机器学习的一个重要工具，我们会在第8章中详细讲这方面内容。现在，我们要快速学习一下PCA，看看在数据处理阶段如何应用它。PCA能在保留最大限度原始变量的前提下，将特征集变换到更低维的空间。PCA不是很直观，除非你一步步钻研数学原理。但好

在R中有一个很有用的包，里面的prcomp()功能可以帮助你做这一切。

让我们使用iris数据集来探索PCA的功能。步骤和R代码如下所示：

- 基于包含iris数据集的数据框，使用基础R stats包中的函数cor()计算相关矩阵。同时把第五列变量Species移除，因为它不是定量的。
- 检查生成的相关矩阵，发现变量Petal.Length和Petal.Width高度相关（96%），Petal.Length和Sepal.Length关联度也很高（87%）。
- 使用基础R包stats中的prcomp()函数对iris数据框进行主成分分析。按照一般推荐的那样，使用scale参数对变量进行缩放，获得单位权方差。我们创建了一个叫做iris_pca的prcomp对象。
- 在iris_pca对象上使用summary()函数，我们可以看到4种主成分。其中，PC1和PC2主成分维系了最多的变量，分别是73%和23%。
- 接着，我们在prcomp对象中的iris_pca上使用plot()函数，图3-4展现了每个主成分的相对方差。

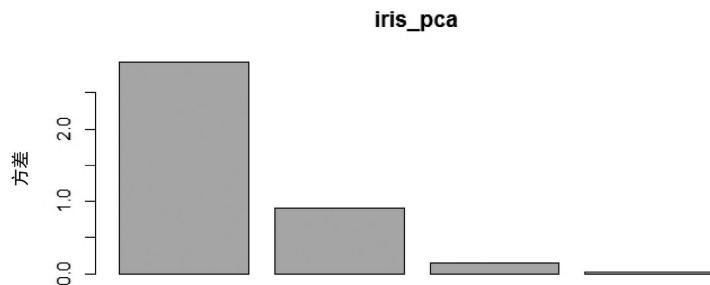


图3-4 PCA图展现了每个主成分的相对方差

- iris_pca\$rotation命令展现了一个“加载”变量的矩阵，举例来说，一个列中包含特征向量的矩阵。
- 现在我们在prcomp对象上使用predict()函数，用前两种主成分计算出降维后的新数据集。
- 最后，我们使用biplot()函数绘制出图3-5，在一张图中同时展现两个主成分的得分和主成分载荷。

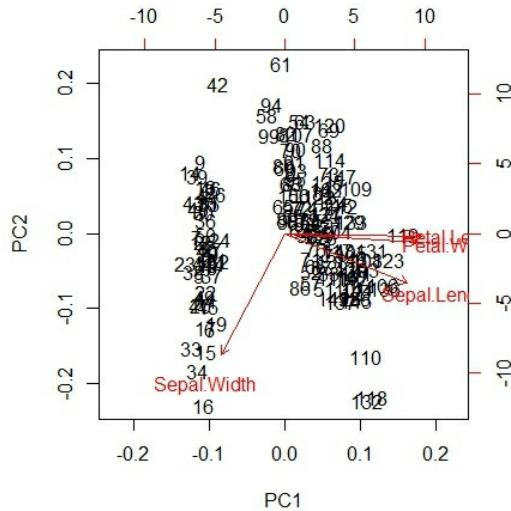


图3-5 PCA前两种主成分的双标图

这个练习简要展现了PCA如何对iris数据集进行降维，从4个特征降到两个特征。这中间涉及的过程和术语我们将在第8章进行更深入的讨论。

```

> cor(iris[,-5]) # calculate a correlation matrix
      Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length  1.00000000   -0.1175698  0.8717538  0.8179411
Sepal.Width   -0.1175698  1.0000000   -0.4284401 -0.3661259
Petal.Length  0.8717538   -0.4284401  1.0000000  0.9628654
Petal.Width   0.8179411   -0.3661259  0.9628654  1.0000000
> iris_pca <- prcomp(iris[,-5], scale=T) # compute a principal
component analysis
> summary(iris_pca)
Importance of components:
              PC1        PC2        PC3        PC4
Standard deviation     1.7084    0.9560    0.38309   0.14393
Proportion of Variance 0.7296    0.2285    0.03669   0.00518
Cumulative Proportion  0.7296    0.9581    0.99482   1.00000
> plot(iris_pca)
> iris_pca$rotation
            PC1        PC2        PC3        PC4
Sepal.Length  0.5210659 -0.37741762  0.7195664  0.2612863
Sepal.Width   -0.2693474 -0.92329566 -0.2443818 -0.1235096
Petal.Length   0.5804131 -0.02449161 -0.1421264 -0.8014492
Petal.Width    0.5648565 -0.06694199 -0.6342727  0.5235971
> predict(iris_pca)[1:2,] # calculate the new reduced dimension
data set
            PC1        PC2        PC3        PC4
[1,] -2.257141 -0.4784238  0.1272796  0.02408751
[2,] -2.074013  0.6718827  0.2338255  0.10266284
> biplot(iris_pca)

```

3.16 小结

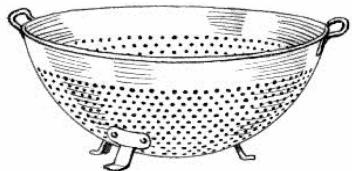
本章的重点是在机器学习项目中经常花费最长时间的部分：数据处理。解决常见数据处理需求的技术都在本章中重点展示了。事实上，这只是冰山一角。还有许多可能出现的要求，例如处理异常值（outlier）、处理字段中前后矛盾的值（举例来说，一个城市字段可能包含LA、L.A.和Los Angeles）、幂和对数变换等等。作为一名数据科学家，你的工作就是在完成更多项目、遭遇不同状况的过程中，不断扩充你的数据处理工具箱。

在机器学习处理过程中，下一步涉及在清洗过的有组织的数据集中执行探索性数据分析。这是第4章的主题。在干净的数据集的帮助下，分析工具能产出更有用的结果。

下面是本章的小结：

- 特征工程采取步骤来识别恰当的特征变量，以便在机器学习算法中使用。
- 数据处理是机器学习“数据管道”中不可分割的一部分。
- 随机取样用于在大型数据集中减少观测行的数目。
- 我们开始用大量常用技术建造一个数据处理工具箱。基于你的个人需求，你可以不断地往里面增加工具。留意新的R包，查看其中是否有符合数据处理需求的工具也是一个很好的方法。
- 在保持相同变量总数的前提下，降维可用于减少数据集中特征变量的数目。

第4章 探索性数据分析



艰苦的数据处理阶段结束之后，机器学习处理的下一步就是通过执行探索性数据分析（Exploratory Data Analysis, EDA）来熟悉数据集。获取这种熟悉程度的方式，是利用R统计环境中能支持这项工作的特性：数据统计、聚合、分布、密度、检查所有的因子变量、一般统计方法的应用、探索性图表、解释性图表等。通常来说，用多种工具探索一个数据集是个好方法，尤其是当它们可以互相比较时。在本章中，我们会给出用于探索性数据分析的指导手册。对数据集有充分认识之后，你很可能需要回头重做一个或多个数据处理任务，更进一步地对数据进行提纯或转换。探索性数据分析的目标是获取对数据的信任，直到你准备好用它执行机器学习算法。

EDA的另一个附加好处是对特征集合进行精炼，以用于后面的机器学习中。在你非常熟悉数据集之后，如果发现选择的特征不能达到预期的目的，就可能需要回到特征工程步骤。此外，你也可能发现加入的其他特征已经满足了数据所要传达的信息。一旦完成了探索性数据分析步骤，你应该有一个确定的特征集，供监督和非监督统计学习使用。

由于急于进行机器学习阶段，一些数据科学家选择跳过整个探索性过程或是敷衍了事。这样的行为是错误的，会导致很多不利的影响，包括生成错误的模型、生成了正确的模型但使用了错误的数据、在数据准备阶段没有创建正确的变量类型，由于只有在生成模型之后才发现数据可能是有偏差的、有异常值、有太多缺失数据或是前后不一致而做很多无用功。

图表是EDA很重要的组成部分。让我们看看为什么在EDA中使用图表来传达数据的信息：

- 理解数据特性。查看整个用于机器学习项目的数据矩阵太困难了，所以观察图表是一种对数据有总体认识的更好的方式。
- 找到数据中的模式。你可以发现新的模式，并在变量之间进行关联。因为之前没有以图表形式看过数据，你可能得到一些意想不到的收获。
- 启发建模策略。图表能告诉你使用哪种建模策略。
- 对分析过程进行纠错。你可以使用图表来确定哪一部分的统计建模可能出错了。记住，大多数机器学习都是迭代过程，所以图表可以为新增步骤提供新视角。
- 说明结果。具备跟其他人说明结果的能力对于任何机器学习项目来说都是很重要的。通常，在执行完EDA之后，你就可以马上讲一个故事。因为目前很少有人能如此深入审视数据，在图表的帮助下，能够浮现一些新视角。“用数据讲故事”是机器学习中很重要的一部分，需要数据科学家具备一些无形的不同寻常的才能。很多探索性数据分析阶段生成的图表为用数据讲故事打下了基础。

在EDA中，有一种不太常用的图表是饼图，因为它涉及角度比较（判断扇形图的相对大小）。比起其他图表类型（例如，柱状图和箱型图）的位置比较，饼图通常更难以解释。3D柱状图也不常用，因为比较体积也是很困难的。

4.1 数据统计

对一名数据科学家来说，希望在数据集中执行的第一种探索形式是数据统计或聚合。熟悉一个特殊值在变量中出现的次数通常来说是有帮助的。例如，使用airquality数据集，让我们看看Month变量中所有的特殊值。为此，我们可以使用R中的unique()函数。这个函数将变量作为参数传递，能返回所有的特殊值。

```
> unique(airquality$Month)
[1] 5 6 7 8 9
```

你可能也想对数据集中某个变量的某个特殊值进行数量统计。虽然在R中有很多方法能实现这一目标，这次我们将使用第2章详细介绍过的sqldf包。用下面的SQL语句能获得某个值的数量（在这个例子中，是Ozone值为11的观测条数），然后发现有3条。

```
> library(sqldf)
> sqldf("select count(Ozone) from airquality where Ozone=11")
      count(Ozone)
1             3
```

另一个有用的函数是summary()，它能检查整个数据集，并提供每个数值变量的众多统计数据：最小值、最大值、平均值、中位数、第一四分位数和第三四分位数。对于因子变量来说，summary()能给出最常出现的值的数目（低频数值的数目在“Other”分类中）。

```
> summary(airquality)
   Ozone          Solar.R          Wind
  Min. : 1.00    Min. : 7.00    Min. : 1.700
  1st Qu.:18.00  1st Qu.:115.8   1st Qu.: 7.400
  Median :31.50  Median :205.0   Median : 9.700
  Mean   :42.13  Mean   :185.9   Mean   : 9.958
  3rd Qu.:63.25 3rd Qu.:258.8   3rd Qu.:11.500
  Max.  :168.00  Max.  :334.0   Max.  :20.700
NA's   :37     NA's   :7
   Temp          Month          Day
  Min. :56.00    Min. :5.000    Min. : 1.0
  1st Qu.:72.00  1st Qu.:6.000   1st Qu.: 8.0
  Median :79.00  Median :7.000   Median :16.0
  Mean   :77.88  Mean   :6.993   Mean   :15.8
  3rd Qu.:85.00 3rd Qu.:8.000   3rd Qu.:23.0
  Max.  :97.00  Max.  :9.000   Max.  :31.0
```

在具体的定量变量上，其他的一些R函数可以执行summary()的大部分功能：mean()可以计算算术平均值，min()能找到最小值，max()能找到最大值，range()返回一个包含最小值和最大值的数组，quantile()可以用于计算最小值、第一四分位数、中位数、第三四分位数和最大值。注意，我们在这些函数中加入了参数na.rm=TRUE，用于忽略所有NA的值。

```
> mean(airquality$Ozone, na.rm=TRUE) #remove all NAs
[1] 42.12931
> min(airquality$Wind) # No NAs in this variable
[1] 1.7
> max(airquality$Solar.R, na.rm=TRUE)
[1] 334
> range(airquality$Month)
[1] 5 9
> quantile(airquality$Ozone, na.rm=TRUE)
0%   25%   50%   75%   100%
1.00 18.00 31.50 63.25 168.00
```

方差是一个统计量，用于展示定量数据值偏离平均值的程度。如果方差为0，说明所有的值都相同。方差总是非负的。若方差很小，说明数据点与平均值都很接近，也就是说，数据值彼此都很接近；若方差较大，说明数据点与平均值相差很远，换句话说，数据点彼此之间相差很远。方差的算数平方根称为标准差。

让我们用airquality数据集来计算几个方差。第一个例子展示了变量Temp的方差。第二个例子展示了去除所有的NA值之后，变量Ozone的方差。

```
> var(airquality$Temp)
[1] 89.59133
> var(airquality$Ozone, na.rm=TRUE)
[1] 1088.201
```

R中也有head()和tail()函数，分别能够快速展示数据集中前6行和后6行记录。这既能快速获得对数据的感觉，又不会花费太多时间。面对一个全新的数据集时，这很可能是你要做的第一个检查。

```
> head(airquality)
  Ozone Solar.R  Wind Temp Month Day
1  41     190    7.4   67     5     1
2  36     118    8.0   72     5     2
3  12     149   12.6   74     5     3
4  18     313   11.5   62     5     4
```

```
5 NA      NA      14.3   56      5      5
6 28      NA      14.9   66      5      6
```

很多数据集中含有类变量，它们的值可能是“男”或“女”，而不是数值。在R中，这些变量称作“因子”变量。对于大多数因子变量，你要做的一项很重要的探索性工作就是，查看这些变量中包含哪些值。可以用R中的levels()函数来达到这一目标。为了演示这项特性，我们用ToothGrowth数据集中的supp变量作为例子。

```
> levels(ToothGrowth$supp)
[1] "OJ"  "VC"
```

在探索一个数据集时，了解某个变量有多少无缺失数据通常是很常用的。在这里展示3种方法，其中，最后一种最简便：

```
> length(airquality$Ozone[is.na(airquality$Ozone) == FALSE])
> length(airquality$Ozone[!is.na(airquality$Ozone)])
> sum(!is.na(airquality$Ozone))
[1] 116
```

4.2 探索性可视化

探索性可视化（或者图表）对于熟悉数据集来说是很有帮助的。使用R中的很多图表功能，你可以快速了解数据传达的信息。你可能不会把探索性图表（exploratory plot）保存或者加入到报告中，但是，它们能指引你做一些初期的决定：如何最好地在机器学习中使用你的数据、选择哪种模型。探索性图表很快完成，通常不必考虑它们是如何创造出来的，只要能将重要的信息传达给数据科学家就可以了。把这些图表当作是供后面使用的“草稿”，最后进行分析。我们画了大量的探索性图表，其中大部分在执行最终的分析之前都会被丢弃。你所要做的，就是让这些图表帮你探索数据。目标是为个人理解带来帮助，不必用这些图表跟别人交流。所以在这个时候别纠结于使用哪些颜色、标题、图例、坐标轴标签等，在探索性阶段，美学不重要。

解释性的图表有些许不同。你可以修饰一下，让它们更适合展示给你们公司的决策者或者客户。这些图表可以嵌入最后的成果报告中。在本章的后面部分，我们会学习这部分内容。

4.3 直方图

在执行EDA时，理解数据集的分布形状十分重要。数据的分布能告诉你数据中是否有异常值，某个机器学习算法是否适合这个数据集，或者仅仅只是让你知道某个数值范围内有多少观测值。一个单变量的频度图，也就是直方图，很简单但是对于快速了解某个变量的值是如何分布的有很大帮助。画直方图的目的是对数据的分散程度进行量化，所以从这个意义上来说，它跟我们后面要看到的boxplot()函数很类似。在iris数据集的帮助下，我们将使用一个直方图对Sepal.Length变量进行探究：

```
> hist(iris$Sepal.Length)
```

图4-1展示了变量Sepal.Length中每个值出现的频率。例如，它展现了在150条观测值中，有5条Sepal.Length观测值介于4和4.5之间。基于这个，我们可以知道它跟条形图不一样，条形图只能展示某个变量特定值的数目。将直方图与条形图进行对比，你可以理解分布形态。

现在让我们看看另一种直方图的形式，它可以展示频数分布。我们将使用hist()函数，参数probability=TRUE。这种形式的直方图显示了值的密度而不是频率。密度可以看做是所有观测中某个值出现的百分比。在直方图中，你可以用参数breaks对间隔的数目进行规定，从而改变显示时的间隔尺寸。最后，用lines()和density()函数添加一条平滑的线，展示分布的密度。密度线是直方图中一个很好的补充，如图4-2所示。

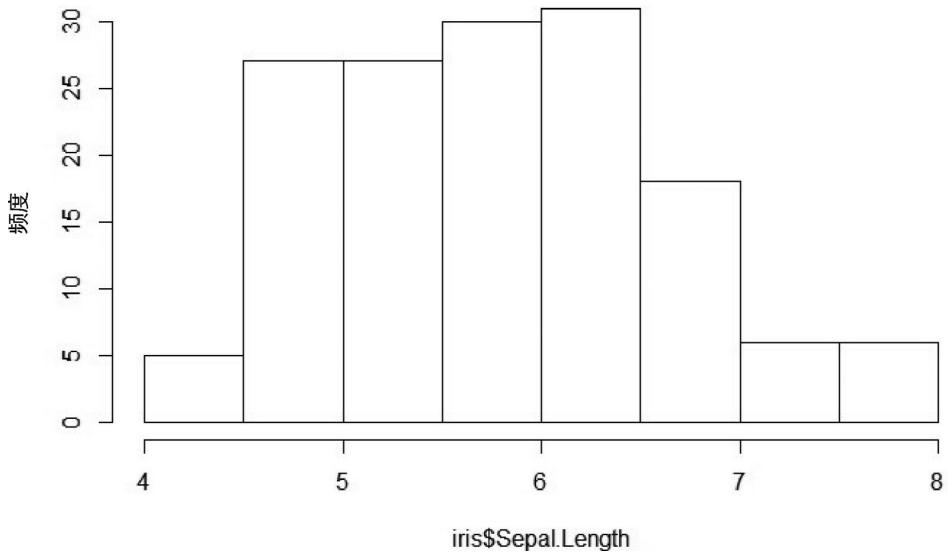


图4-1 直方图展示了变量Sepal.Length的值的频度

```
> hist(iris$Sepal.Length, probability=TRUE, breaks=10)
> lines(density(iris$Sepal.Length))
```

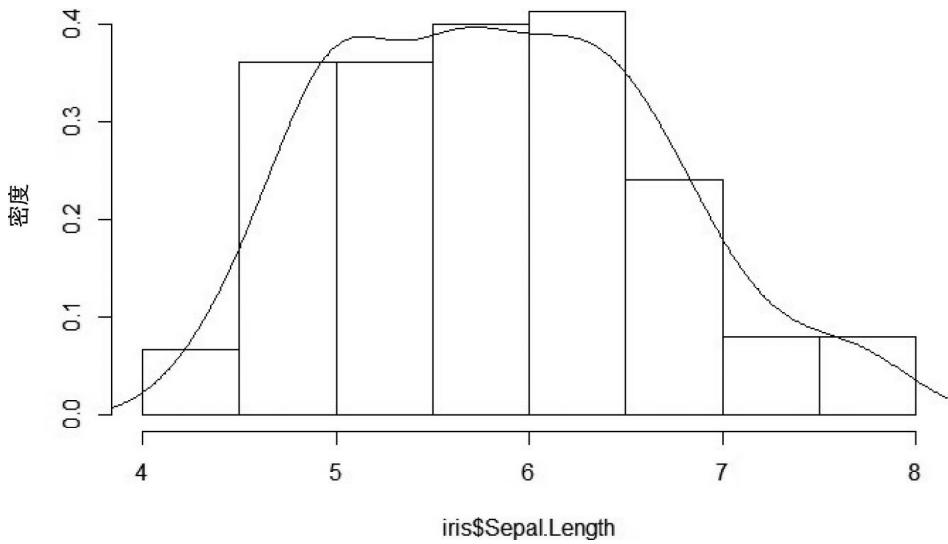


图4-2 带有拟合线的直方图，展示了Sepal.Length变量的密度分布

4.4 箱形图

我们要讨论的下一种图表是箱形图。它对定量变量尤其有用。箱形图的目标是让你从另一个方面看数据的分布规律。下面的例子使用了airquality数据集中的Ozone变量，用boxplot()函数进行绘图。将定量变量作为参数传递，同时设置col="blue"，这样能使箱形图的箱体部分不再是黑色，更便于与图表的其他部分区分开来。

```
> boxplot(airquality$Ozone, col="blue")
```

让我们描述一下图4-3中箱形图的所有组成部分。箱形图可以表示多个不同数值。

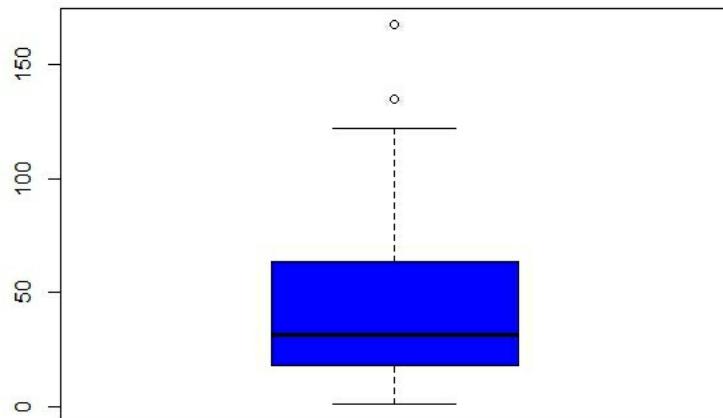


图4-3 变量airquality.Ozone的箱形图

在箱体中央的粗黑线代表了中位数，或者是分布的中心（在图中展示的是31.5）。中位数又被称为第二四分位数（Q2）。箱体的上下界分别代表了数据的第三（Q3）和第一（Q1）四分位数。矩形从上到下的长度指的是四分位差（IQR）。从箱体中纵向延伸出来的线称为“触须”，上方的触须表示的是最大值或者第三四分位数加上1.5倍的IQR，

取两者中的较小值；下方的触须表示最小值或者第一四分位数加上1.5倍的IQR，取两者中的较大值。你会发现有一些数据点游离在上方触须之外，这些表示的是极端值（同样，极端值也会出现在下触须的下方）。

通过观察箱形图，你可以得到数据的定量分布。也就是说，知道数据值分布是发散的还是紧密的。同时，你也可以通过观察中位线落在箱体的哪个部分来判断分布是否均匀。如果中位线靠近箱体的中心，那么分布是比较均匀的；反之，如果中位线靠近箱体的顶部或者底部，那么分布不太均匀。在这个例子中，我们可以看到Ozone的分布不太均匀，因为中位线靠近箱体的底部。

箱形图也有利于在一个通用尺度下比较多个变量的分布规律。在同一标准下生成图表，我们可以直接比较中位数、四分位数和四分位数间距。使用箱形图对中位数进行比较可以看做是双样本t检验和单向方差分析的图形等效。例如，下面调用的boxplot()函数会展示用supp进行分类后ToothGrowth数据集中len的数值。在这个例子中，因为supp只有两种值：VC和OJ，所以我们得到了两个箱形图。

```
> boxplot(ToothGrowth$len~as.factor(ToothGrowth$supp), col="blue")
```

生成的结果如图4-4所示。我们可以看到OJ中len的中位数比VC中的要高。所以，通过把箱形图放在同一尺度下，你可以比较分布的中心和变化。

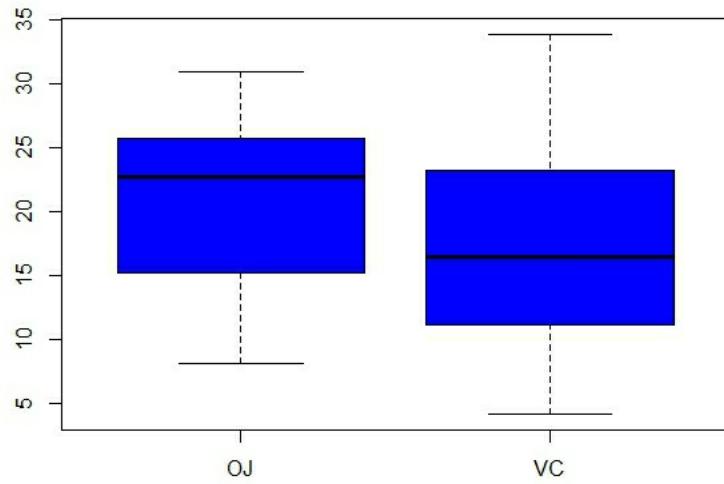


图4-4 通过supp分类后变量ToothGrowth.len的箱形图

为了确定分类OJ和VC中观测的相对数目，你可以给boxplot()函数加入参数varwidth=TRUE。这样，箱体的宽度就会与观测数目成比例。在我们演示的例子中，因为每个分类都有30条观测数据，所以箱体是一样大小的。因此，我们为箱体填充了不同的颜色。

```
> boxplot(ToothGrowth$len~as.factor(ToothGrowth$supp),
  col=c("blue", "orange"), varwidth=TRUE)
```

4.5 条形图

EDA中另一种有用的图表是条形图。在条形图中你可以基于位置比较数值。在这个例子中，我们使用了barplot()和table()函数，用于统计airquality数据集中变量Temp的每一个数值的数量。在这里我们可以看到，Temp值为56的观测有1条，75的有4条，90的有3条。我们可以使用这些数据绘制条形图，如图4-5所示。

```
> table(airquality$Temp)
56 57 58 59 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 :
 1  3  2  2  3  2  1  2  2  3  4  4  3  1  3  3  5  4  4  9  7  6  6  5 11  9  4
88 89 90 91 92 93 94 96 97
 3  2  3  2  5  3  2  1  1
> barplot(table(airquality$Temp), col="blue")
```

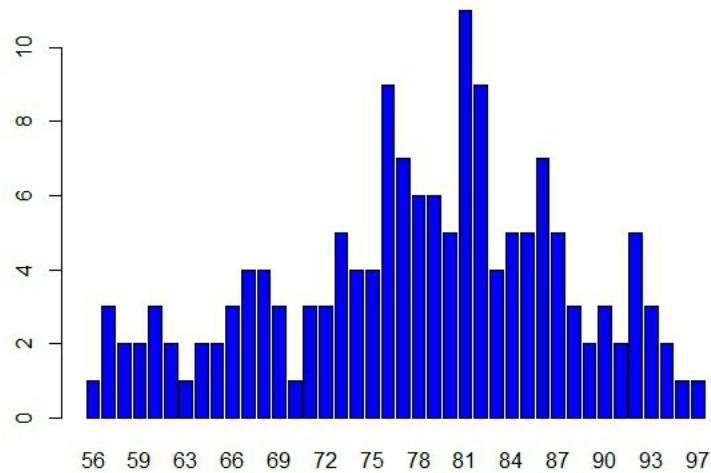


图4-5 展示变量airquality\$Temp数值分布的条形图

在条形图中，条的高度等于数值的数量。这意味着只要看一眼你就能知道在任何值下的观测值的数目。在本例中，我们可以快速发现数据集中Temp等于81出现的频率最高。在机器学习项目的EDA阶段，了解

数据的分布是十分重要的。

4.6 密度图

EDA中另一种很重要的图表类型是密度图（density plot）。密度图类似于平滑后的直方图。所以除了直方条之外，我们也可以用曲线表示分布的密度。图4-6是用airquality数据集中的变量Temp生成的密度图示例。我们使用density()函数，可以在R中计算变量的核密度估计。核函数是概率密度函数（PDF）中特殊的一种，它添加了属性：必须是偶数（除此之外，必须是非负实数）。一些常用的PDF是核函数，例如均匀分布和正态分布。生成的图表如图4-6所示。

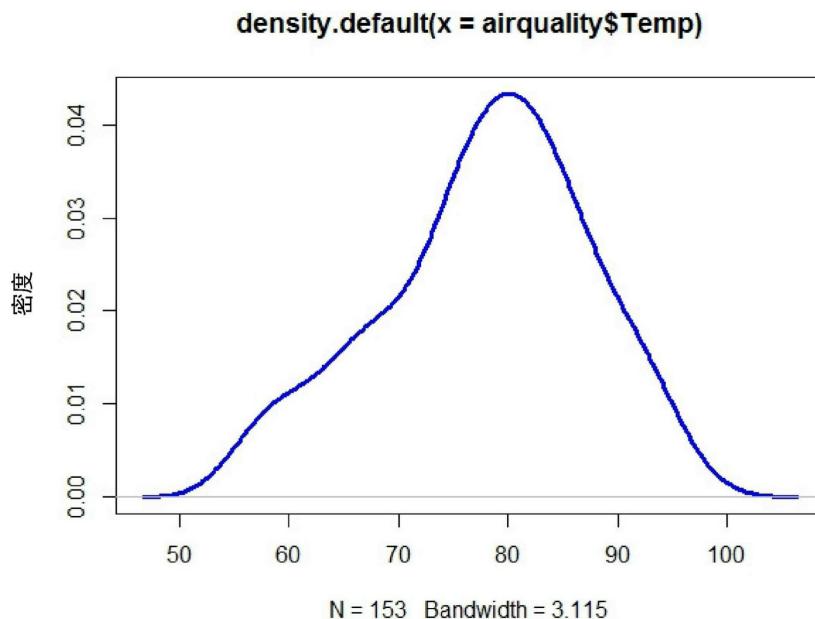


图4-6 变量airquality\$Temp的密度图

```
> temp_dens <- density(airquality$Temp)
> plot(temp_dens, lwd=3, col="blue")
```

密度和我们在直方图中看到的频率是不同的统计概念。所以我们计算观测值介于0和50之间的百分比，而不是确切的数目。即使曲线已经平滑过，平滑过程会在边界位置引入少量误差，但你还是能看到像直方图一样的形状。虽然图似乎显示了没有气温在50以下或者100以上的情况。这意味着你在解释密度图的边界时，应该格外仔细。

选择使用密度图而不是直方图的一个原因是，密度图可以在一张图中叠加多个分布。为了演示这个概念，我们使用ToothGrowth数据集，并用与上面类似的方法为变量len创建一个密度图。更进一步，为变量len加入第二个密度，但是这次只针对于supp值为VC的观测。在下面的R代码中，使用which()函数获取VC观测的子集。在生成的图4-7中，我们可以很容易地比较密度，所有观测的密度与子集的密度略有不同。

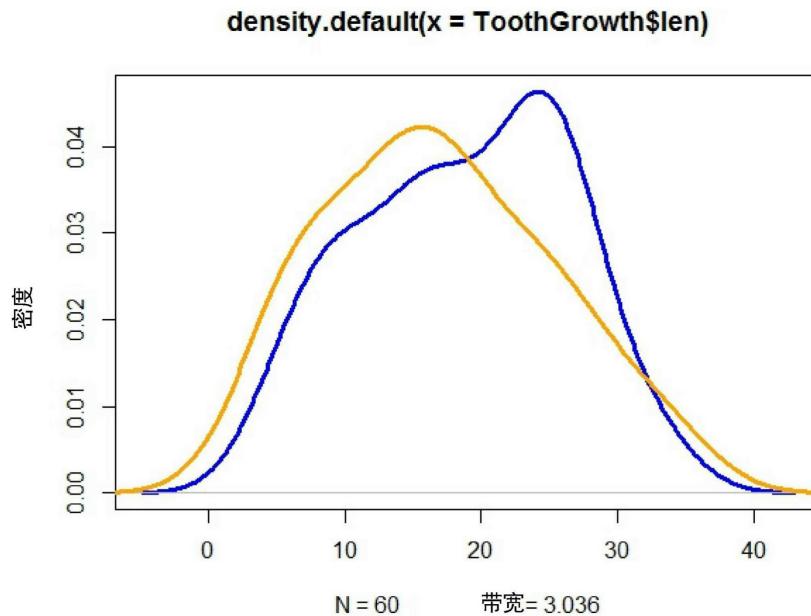


图4-7 变量ToothGrowth\$len的多重密度图

```
> len_dens <- density(ToothGrowth$len)
> plot(len_dens, lwd=3, col="blue")
> VC_dens <- density(ToothGrowth$len[which(ToothGrowth$supp=="VC")])
> lines(VC_dens, lwd=3, col="orange")
```

4.7 散点图

在EDA中，我们要考虑的下一种图表是散点图（scatterplot）。散点图在识别变量的视觉关系时十分有用，因此它也是在探索性数据分析阶段最常用的图表。在下面的R代码中，我们使用plot()函数来实现散点图。使用ToothGrowth数据集中两个定量变量len和dose。参数pch指定了图中代表数据点的符号。取值为19代表实心圆，也有其他很多符号可供选择。在R帮助中，?pch可以说明其他选项。注意，在R的散点图中，如果数据集中有NA值（缺失值），R不会画出这个点。如果你在探究阶段没有意识到这个问题，可能会得出一些错误的结论。

```
> plot(ToothGrowth$len, ToothGrowth$dose, pch=19, col="blue")
```

生成的散点图如图4-8所示。

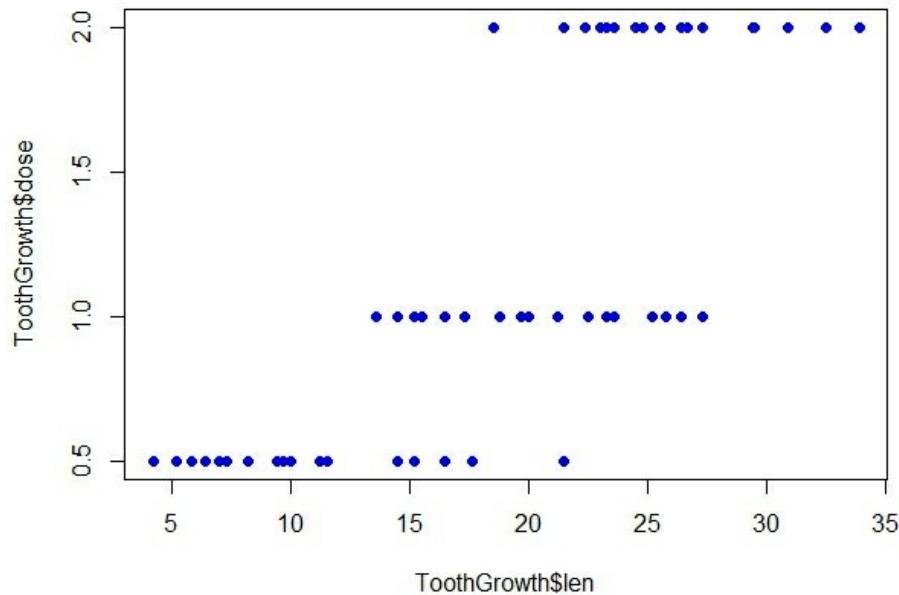


图4-8 比较变量len和dose的散点图

图中x轴是变量len, y轴是变量dose。图中的每个点表示一条观测记录。从图中我们可以看到, dose的值是离散的, 即0.5、1.0和2.0。在这种形式展现的数据中, 你总是能马上发现特征。这种特征有助于了解数据。我们能很快发现的一个特点是: 随着剂量增加, 牙齿的长度也随之增加。这样的特征对于解释数据以及在后面的机器学习过程中如何最好地对数据集进行建模都非常重要。

在散点图的帮助下，你也可以使用col和pch参数在图中展示第3个变量。例如，col可以用不同的数据点颜色展示不同的数值，举例来说，当supp的值为“VC”时，我们可以将数据点标成红色；当supp的值为“OJ”时，可以将数据点标为蓝色。这意味着我们可以把3个变量展现在同一个二维图表中。或者，用pch为每个数据点分配不同的形状。考虑下面给出的R代码。若supp值为“VC”，将pch置为0（即空心方形）；若supp值为“OJ”，将pch置为1（即空心圆形）。结果如图4-9所示。

```
> plot(ToothGrowth$len, ToothGrowth$dose, pch=ifelse  
(ToothGrowth$supp=="VC", 0, 1), col="blue")
```

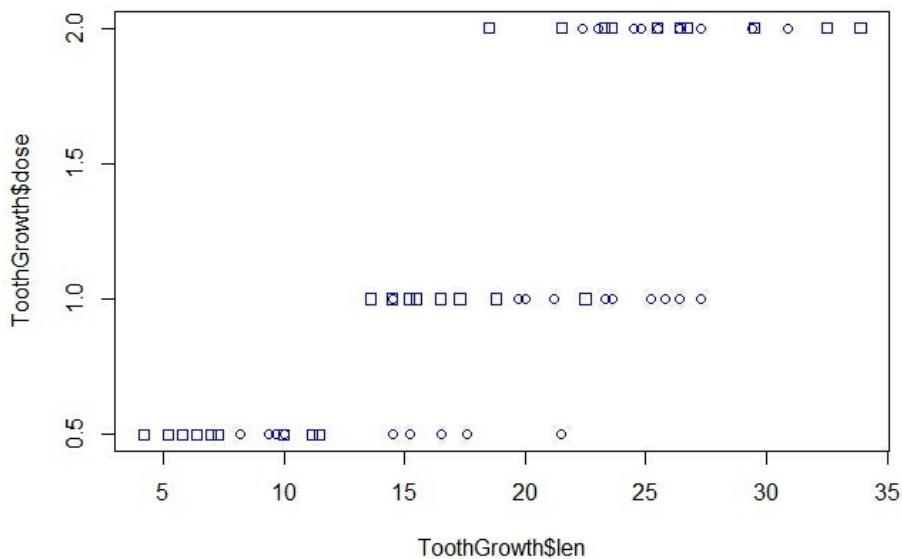


图4-9 不同supp值下len与dose变量关系散点图

另一个散点图的示例涉及scatterplot3d包的使用，它能在一个三维图

中展示变量间的三维关系。下面是示例的R代码，使用airquality数据集，变量Solar.R为x轴，Wind为y轴，Temp为z轴。结果如图4-10所示。

```
> library(scatterplot3d)
> scatterplot3d(airquality$Solar.R, airquality$Wind,
airquality$Temp, highlight.3d=TRUE, col.axis="blue", col.grid=
"lightblue", main="Air Quality Data Set", pch=20, xlab=
"Solar Radiation", ylab="Wind", zlab="Temp")
```

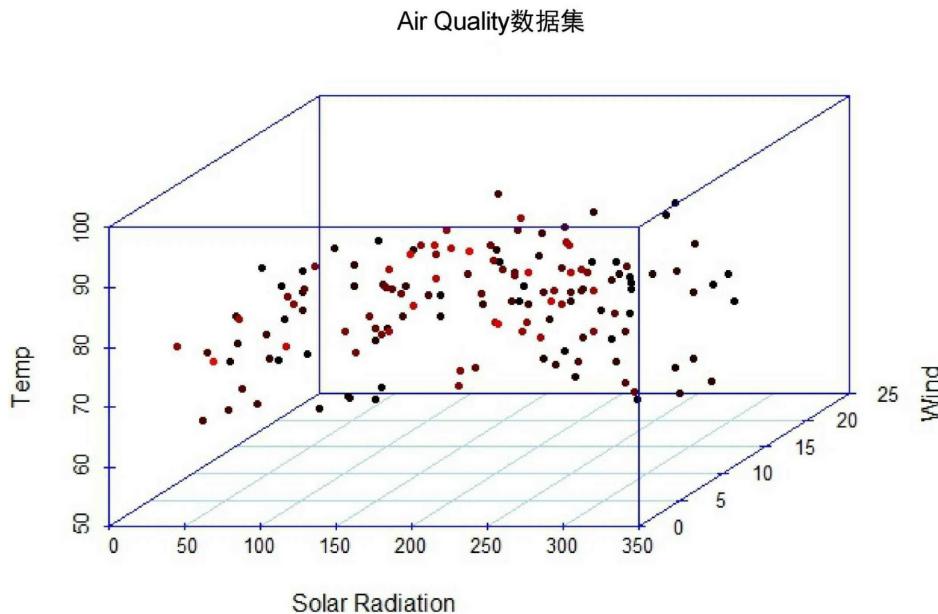


图4-10 airquality数据集中Solar Radiation、Temp和Wind的3D散点图

计算一个数据集中定量变量的相关矩阵，然后绘制一张散点图矩阵来目测相关性，这是一项很有价值的工作。让我们使用iris数据集来演示这一技巧。R中的cor()函数可以用来计算相关矩阵。

```
> cor(iris[,c(1,2,3,4)], method="pearson")
   Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length  1.0000000 -0.1175698  0.8717538  0.8179411
Sepal.Width   -0.1175698  1.0000000 -0.4284401 -0.3661259
Petal.Length   0.8717538 -0.4284401  1.0000000  0.9628654
Petal.Width    0.8179411 -0.3661259   0.9628654  1.0000000
```

相关矩阵将变量组合在一起，计算两个变量之间的相关性。例如，我们看到Sepal.Length和Petal.Width的相关性是81.8%，这说明了它们之间的相关性较强。更进一步，Petal.Length和Petal.Width之间达到了更高的相关性，即96.3%。相关性指的是任何一类广泛的统计关系都涉及到

的依赖。在R的例子中，默认使用的是Pearson相关性计算法。不过，通过改变method参数，也可以使用Kendall法或Spearman法计算。

现在让我们使用pairs()函数对相关性进行散点图可视化，如图4-11所示。在这里，我们可以用相关矩阵对相关性进行定量展示。

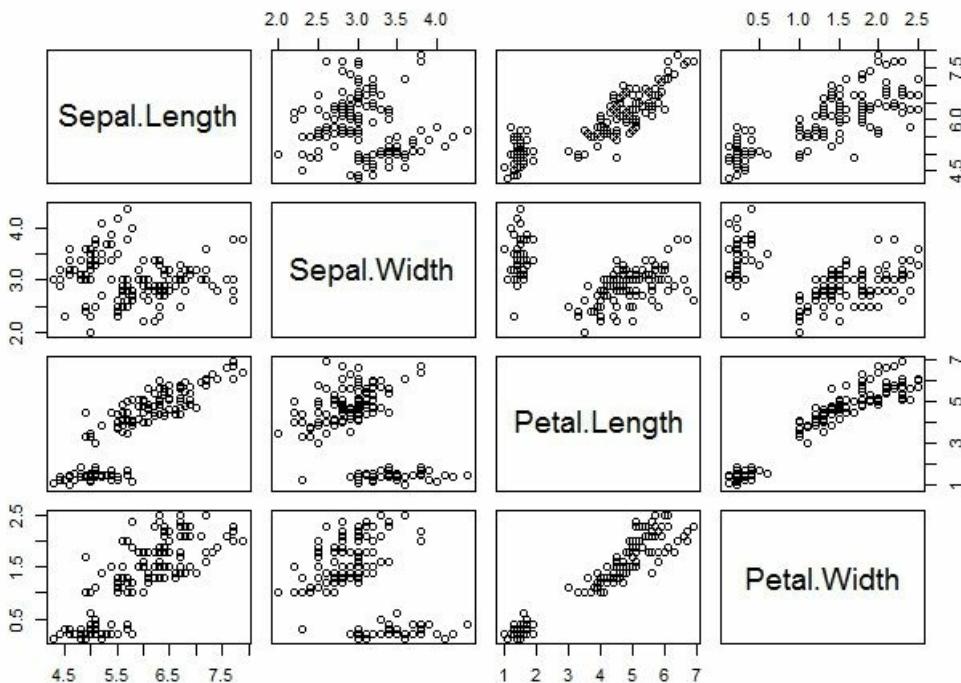


图4-11 iris数据集中定量变量的散点图矩阵

看看变量Petal.Length和Petal.Width的子图。这两个变量的图表展示了一个非常明显的趋势：随着Petal.Width的增加，Petal.Length也会增加。在另一方面，变量Sepal.Length和Sepal.Width展现出-11.7%的负相关。这些知识可以帮助你进行后面的监督机器学习。

```
> pairs(iris[,c(1,2,3,4)])
```

我们将在基于海量数据的大数据机器学习项目中看看散点图在探索性数据分析中的效果。为此，我们将使用一个大型仿真数据集。在下面的R代码中，我们使用了rnorm()函数生成了两组正态分布的随机数（每组100 000条数据）。生成的两个变量的散点图如图4-12所示。它基本上

是一大团点，这对于大型数据集来说很常见。因为大多数的点叠加在一起，不可能知道图表密集的中间部分发生了什么。具体地说，你无法确定点的密度，即点可能处于哪一块确定的区域。所以，我们需要使用其他工具来探究数据的特点。

```
> x <- rnorm(1e5)
> y <- rnorm(1e5)
> plot(x, y, pch=16)
```

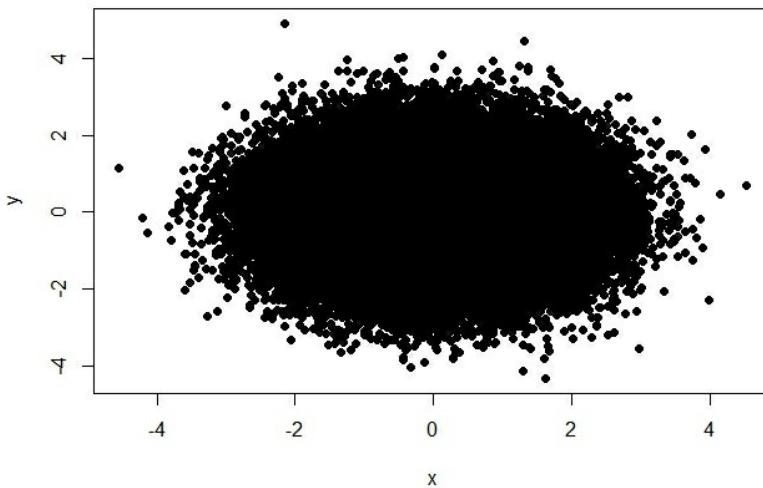


图4-12 极度稠密的散点图是海量数据数据集的典型特征

一种技巧是对数据集中的数值进行取样。在下面的R代码中，使用`sample()`函数抽取1 000条不重复的随机数据，生成一个整型向量当做x和y指标。由于我们的目标是用肉眼观察一千个数据点，使用`plot()`。注意我们如何用`sampledSubset`作为索引获取x和y对应的值。结果如图4-13所示。现在数据点易分辨得多了。即便我们处理的是原始数据集中的一小部分随机子样本，你也可以看到两个变量之间的关系。对于探索性数据分析来说，你不必看到所有的数据点。

```
> sampledSubset <- sample(1:1e5, size=1000, replace=FALSE)
> plot(x[sampledSubset], y[sampledSubset], pch=16)
```

另一种处理海量数据点的方法是使用`smoothScatter()`函数，它能通过核密度估计获取一个代表散点图的彩色平滑密度。在这个例子中，我

们使用整个仿真数据集。数据点越多的地方，颜色越深，如图4-14所示。注意，离中心越近，数据点越多。异常值以实心点的形式展示，它们分布在图的边缘。所以你不用看到所有的数据点，一张平滑散点图就能帮助你了解数据之间的关系。

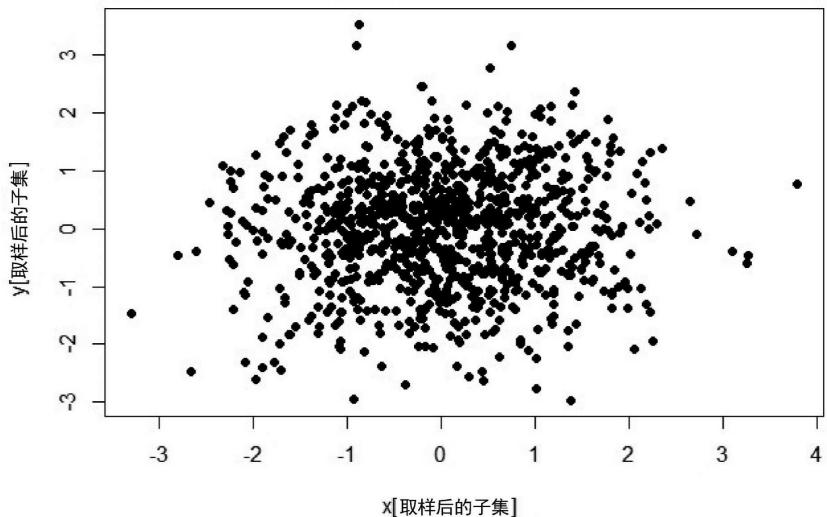


图4-13 100 000条数据中1 000条随机子样本的散点图

```
> smoothScatter(x,y)
```

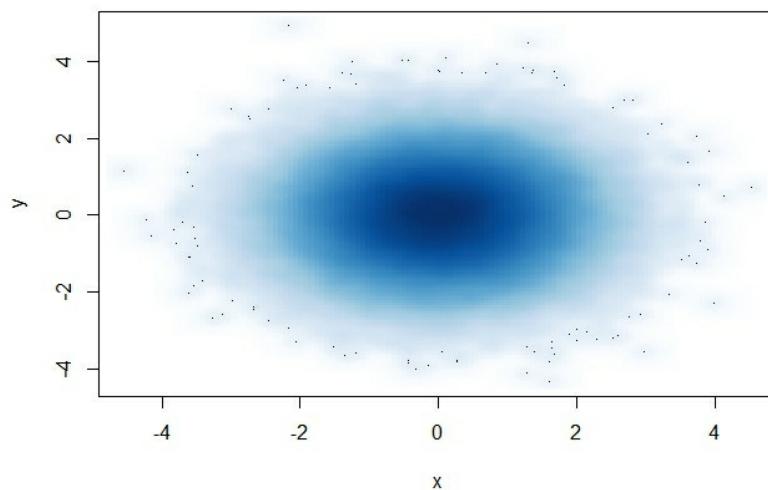


图4-14 整个数据集的平滑散点图

最后，我们使用一个叫“六边形箱”的工具来展示一个数据集中的大量数据点。在下面的R代码中，使用hexbin包根据x和y创建hexbin对象。图4-15将x和y的值分解为六边形箱，然后计算每个箱中数据点的数目。

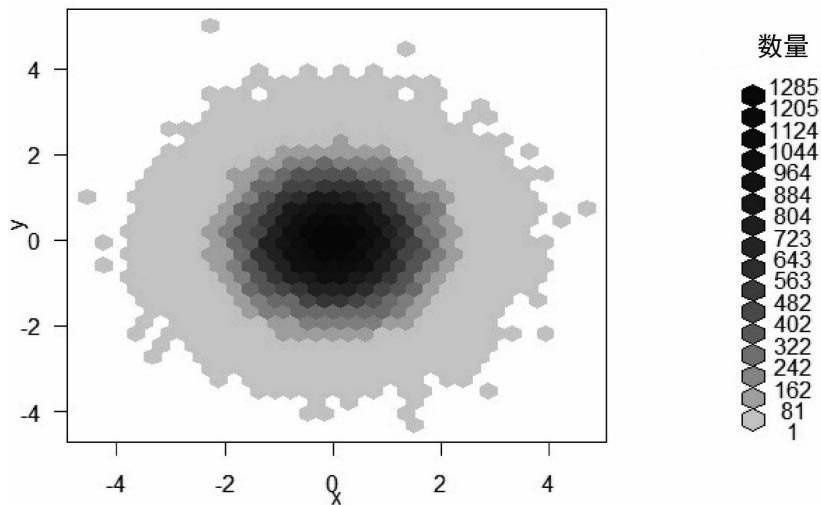


图4-15 用于对大量数据点进行可视化的六边形箱

同理，图中深色的箱子意味着更多的数据点，而外部浅色的箱子意味着较少的数据点。在有大量数据点的情况下，使用箱方法能帮助你观察x和y之间的关系。

```
> install.packages("hexbin")
> library(hexbin)
> hbins <- hexbin(x,y)
> plot(hbins)
```

4.8 QQ图

另一种探索性图表是QQ图，或者称作分位数-分位数图（quantile-quantile plot）。因为都用于定量变量，这种图在某些方面和散点图比较相似。区别在于这种图用一个变量的分位数-另一个变量的分位数来作图。它用于目测两个变量之间潜在的分布相似性。图中的坐标轴代表变量的第一分位数一直到第一百分位数。在下面的R代码中，我们使用了一个只有50个数据点的小型仿真数据集和两个变量x和y。如果这两个变量分布相同，你可以预测分位数准确地落在图中abline()函数画出的那条线上（截距为0，斜率为1）。然而，在图4-16中，我们看到一些差异，比如图的顶部y的分位数比x的分位数要大。

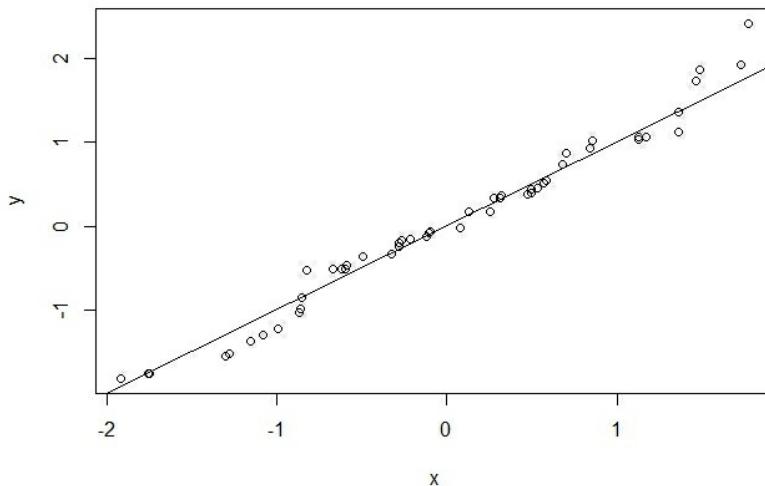


图4-16 一个正态分布仿真数据集的QQ图

QQ图对于观察数据集是否正态分布十分有用。

```
> x <- rnorm(50)
> y <- rnorm(50)
> qqplot(x,y)
> abline(c(0,1))
```

4.9 热图

另一种有用的探索性图表是热图（heatmap），它有点类似于二维的直方图。热图使用函数image()。热图背后的想法是使用色彩亮度来展示一个数据值的大小，所以，颜色越亮，值越大。白色对应最大值，红色对应最小值，中间的渐变色代表了中间的其他值。在热图的帮助下，你可以在一张图中看到整个矩阵。

让我们用iris数据集画一张热图，R代码如下所示。第一个参数代表用于图表中的矩阵行（观测行），第二个参数指的是特征变量。第三个参数需要是一个矩阵对象，所以我们将iris数据框用as.matrix()函数转换为矩阵的形式。在这个例子中，数据集为中等大小，所以我们将使用全部150行观测数据和所有的4个定量变量。

```
> image(1:150, 1:4, as.matrix(iris[1:150, 1:4]))
```

图4-17展示了生成的热图，其中行4的值最小，行1的值最大。在看热图的过程中，要注意图的顺序是颠倒的。也就是说，图中的行相当于特征变量，列相当于观测值。这可能有点违反直觉，因为我们在image()函数调用中先指定行再指定列。当然，如果你愿意的话，你可以用下面的R代码对矩阵进行转置。

```
> transMatrix <- as.matrix(iris[1:150, 1:4])
> transMatrix <- t(transMatrix)[,nrow(transMatrix):1]
> image(1:4, 1:150, transMatrix)
```

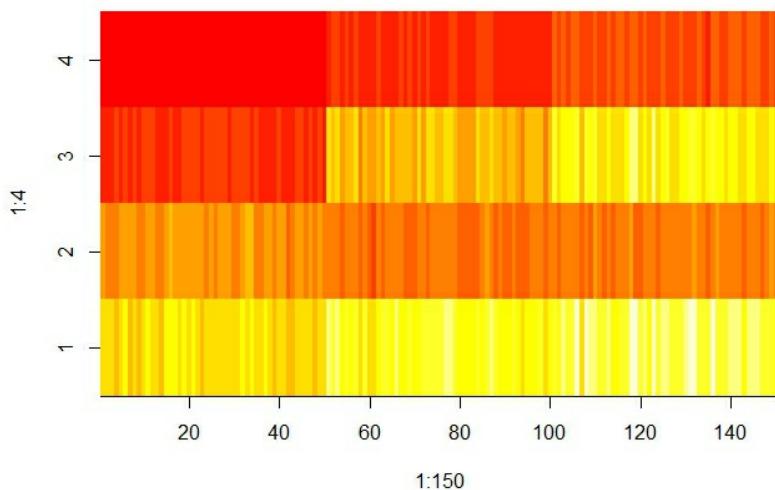


图4-17 iris数据集的热图

4.10 缺失值的图表

我们要讨论的最后一种探索性图表是箱形图，可以用来更好地理解数据集中的缺失值。例如，我们使用airquality数据集中含有NA值的一个变量Solar.R，来探究Solar.R中的缺失值是否跟Temp的值有联系。为此，我们使用下面的boxplot()，它能比较Solar.R的值为空和不为空的情况。如图4-18所示，你可以发现当Solar.R值为NA时，Temp的值普遍更小；当Solar.R值不为NA时，Temp的值普遍更大。

```
> boxplot(airquality$Temp ~ is.na(airquality$Solar.R))
```

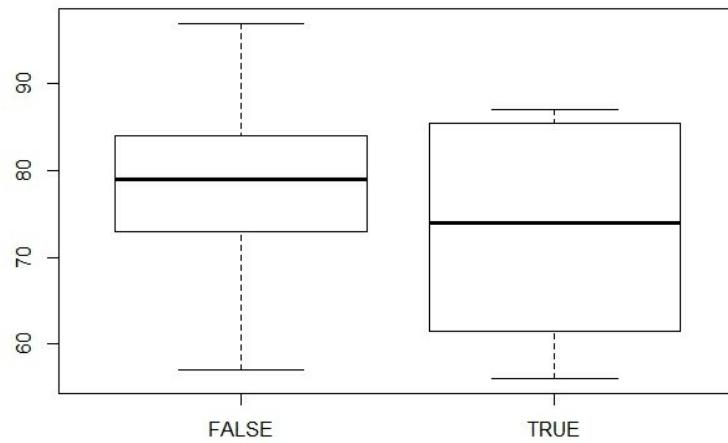


图4-18 使用箱形图探究含有缺失值的变量间的关系

4.11 解释性图表

相比于上面提到的探索性图表，解释性图表要更正式、更固定。此外，这种图用于跟其他人交流结果。要对探索性图表进行数次迭代，才能生成一张解释性图表。探索性图表用于发现数据中的隐藏信息，而解释性图表用于传达数据的故事，并且它们常常能进入机器学习项目的最终报告和结果中。因为这些图表是项目最终记录的一部分，你需要美化标题、标签、图例等。下面是解释性图表常见的一些目标：

- 总体目标是传递信息。
- 信息密度需要达到一定的水平，即用语言不可能解释结果。
- 尺寸和颜色不但要传递信息，也要有美学方面的考虑。
- 这些图表应该包含大号且易于理解的标题、坐标轴和图例。

让我们看一个使用airquality数据集的详细例子，R代码如下所示。我们的目标是生成有两个面板的一个解释性图表。R可以以多种方式堆积面板。使用par(mfrow=c(1,2))函数，我们可以得到一行两列的面板，所以图表将会并排出现。然后我们使用变量Ozone的直方图画出第一个面板。用xlab参数在x轴加上标签（不用R描述，它是默认使用的），用main参数为图表加上标题。

第二个图表是变量Ozone和Temp的散点图。我们用一些特殊的特征，例如用cex参数将数据点进行放大（为了更易辨别），同时我们也用描述性的文本对坐标轴和单位进行标记。在解释性图表中，对所有坐标轴的单位进行标注是很重要的。最后，我们用legend()函数在图中添加一个简单的图例。注意legend()出现在左上角坐标轴上，那里是legend框根据前两个参数确定的位置。得到的解释性图表如图4-19所示。R提供了很多额外的特征来美化图表，从而提供一个有效而专业的展示。

```
> par(mfrow=c(1, 2))
> hist(airquality$Ozone, xlab="Ozone (ppb)", col="blue", main="Ozone Frequencies"
> plot(airquality$Ozone, airquality$Temp, pch=16, col="blue", cex=1.25, xlab="Ozon
main="Air Quality - Ozone vs. Temp", cex.axis=1.5)
> legend(125, 60, legend="May-Sep 1973", col="blue", pch=16, cex=1.0)
```

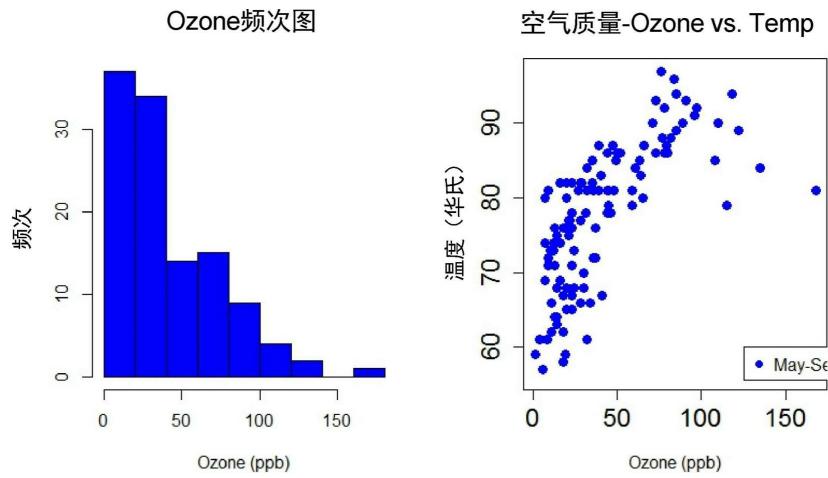


图4-19 可以用于最终数据科学报告的解释性图表案例

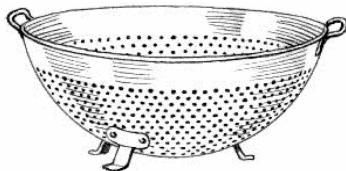
4.12 小结

在本章中，我们学习了如何通过执行简单的探索性数据分析来熟悉数据集。我们用了多种工具对数据集建立了深入的了解。下一步是使用大量的算法和监督学习工具来进行机器学习实验，这将会在第5章、第6章进行讨论。

下面是本章知识的小结：

- 数据统计在深入探索用于机器学习的数据集时特别有用。我们学习了如何使用数据统计工具，例如，`unique()`、`summary()`、`mean()`、`min()`、`max()`、`range()`、`quantile()`、`var()`、`head()`、`tail()`、`levels()`和`sum()`。
- 探索性可视化是将数据集的数值以图形化的形式展现。不必看具体的数据值，图表就能提供很多重要信息。我们学习了一些探索性工具，例如，直方图、箱形图、条形图、密度图、散点图、QQ图、热图以及缺失数据图。
- 解释性可视化是更正式的图表，它们常常用于机器学习项目的最终展示。这些图表经过美化，可以更好地向别人传达结果。

第5章 回归



在本章中，我们要迈出将机器学习方法应用于解决数据科学问题的第一步。监督学习是机器学习中最常见的一种，它经常和预测性分析联系在一起。为了演示监督学习技术，我们一起来看几种最常用的算法，例如，一元线性回归、多元线性回归和多项式回归（polynomial regression）。回归能有效预测定量数值，例如产品销售收入、房屋价格、通货膨胀率等。这种方法已经存在了很长时间了。和其他受数据科学家欢迎的算法相比，虽然学习这种算法看起来有些单调乏味，但是回归依然是监督机器学习的主力，它非常有用并且也广泛使用。在学习更复杂的工具之前，对回归算法有充分了解，将有利于你掌握机器学习工具。在本章中，我们将围绕着回归模型学习一些关键概念。

在学习回归算法时，我们需要思考跟手头项目有关的几个重要问题，例如：

- 预测的变量和用于预测的变量之间有关系吗？更进一步，它们之间的关系有多强？预测结果具有很高的精度吗？
- 哪些可用的特征变量对预测有帮助？每个变量贡献多少？使用这些特征变量能使预测到达怎样的精度？
- 它们的关系是线性的吗？

这些问题可以通过使用机器学习中的回归方法来回答。在接下去的小节中，我们会看几个不同方向的回归，你可以用这些操作解决重要的数据科学难题。

5.1 一元线性回归

线性回归（linear regression）在统计学领域有很长的历史，并且它也是最容易理解和最受欢迎的监督机器学习算法。它基于特征变量（预测因子）和响应变量（response variable）（预测的变量）之间有线性关系这一假设。对于线性回归，这里假定响应变量是定量的，预测因子是定量或者定性的。这个算法使用已知的线性数学函数来基于观测到的数据估计“未知”的函数。已知的函数常称为一个“假说”函数，因为这是你选择的近似实际的（虽然实际怎么样我们还不知道）函数。在线性回归中，我们假设未知的函数是线性的。使用已知的函数，我们可以用数据集来“拟合”或者“训练”模型。在这个例子中，我们需要学习假设函数的一套回归系数以及最常见的拟合模型的方式即最小二乘法（least squares）。

最小二乘法是由所有误差项的平方和决定的。在这里，误差是指实际响应变量值和预测值的差。我们将每个误差项做平方，是为了剔除负误差值可能造成的影响。目的是得到最小的误差平方值，从而使得误差尽可能小。我们下面会讨论到，R中的lm()算法能帮助处理最小二乘计算。

然而，我们不想完全完美拟合训练集，因为这可能造成过拟合，这意味着对于新观测的响应变量，模型可能产生不精确的估计。因为新观测不是用于训练线性模型的原始数据中的一部分。在本章的后面部分，我们将讨论将数据集拆分成训练集和测试集。目前，我们将使用lm()函数的输出作预测。

现在我们举一个实际的一元线性回归案例，使用MASS库中的Boston数据集。这个数据集中包含波士顿郊区的房屋价格，它有506行观测值和14个变量。我们使用的两个变量大致上呈线性关系：mdev（自住房屋价格的中位数，以\$1000为单位）和rm（每个住宅的平均房间数）。一般来说，有更多房间的大房子会对应更高的价格。我们尝试使用线性回归和房间数来预测住宅价格。两个变量都是定量的。

在下面的代码中，我们加载了MASS库和Boston数据集，然后展示数据集中的变量名。

```

> library(MASS)
> data(Boston) # 506 obs and 14 variables
> names(Boston) # Show variables
[1] "crim"   "zn"     "indus"  "chas"   "nox"    "rm"
[7] "age"    "dis"    "rad"    "tax"    "ptratio" "black"
[13] "lstat"  "medv"

```

下一步，我们使用`lm()`“线性模型”函数，它是R中执行线性回归算法的工具。`lm()`有很多参数，但是这时候我们只关心响应变量和单一预测变量（predictor variable）。传递给`lm()`的参数是“模型公式”，在这个例子中是`medv~rm`，在波浪字符～的左边是响应变量（读作“建模为”、“由……预测”、“取决于”等等），模型设定公式的右边是预测因子（在下一节中，我们将使用多个预测因子）。在我们的例子中，`medv~rm`可以读作“`medv`由`rm`预测”。另一个参数`data=Boston`，指定了数据存在的数据框。这样你就不用重复输入数据框的名字了，就像是`Boston$medv~Boston$rm`，虽然这样也有相同的效果。就像R中其他机器学习算法一样，计算的结果存放在一个对象变量中——在这个例子中是`lm1`。这里，`lm1`是线性模型拟合。如果展示这个对象，你可以看到原始函数调用和计算出的回归系数（截距和斜率）。

```

> lm1 <- lm(medv~rm, data=Boston)
> lm1
Call:
lm(formula = medv~rm, data = Boston)
Coefficients:
(Intercept)      rm
-34.671       9.102

```

一旦使用`lm()`来拟合数据模型，一个重要的问题是，拟合是否有价值。它能成功地做出预测吗？为了确定这些，你可以使用`summary()`函数来获得一些判断拟合质量的依据。注意，在线性模型对象中获取信息需要一些数学和统计学方面的知识，这超出了本书的范围。作为替代，我们将学习线性模型对象的一些基本元素，并学会如何解释它们。

```

> summary(lm1)
Call:
lm(formula = medv~rm, data = Boston)

Residuals:
    Min      1Q  Median      3Q      Max 
-23.346  -2.547   0.090   2.986  39.433 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -34.671     2.650  -13.08   <2e-16 ***

```

```

rm           9.102      0.419    21.72   <2e-16 ***
---
Signif. codes: 0'***'0.001'**'0.01'*'0.05'.0.1' '1
Residual standard error: 6.616 on 504 degrees of freedom
Multiple R-squared: 0.4835, Adjusted R-squared: 0.4825
F-statistic: 471.8 on 1 and 504 DF, p-value: < 2.2e-16

```

使用summary()能得到假设几率、回归系数的标准误差（standard error），还有模型的R²统计量和F检验统计量。如果你愿意的话，可以在线性模型总结中参考每个统计量。在下面的讨论中，我们将学习每一个组件。作为开始，你可以使用? summary.lm命令来查看具体的可用信息。

第一，我们将读取回归系数矩阵（coefficients matrix）（一个参数表），其中包含4列：估计系数、截距和斜率的标准误差、截距和斜率的t值（t检验）、截距和斜率得到p值。在执行多元线性回归时，需要在这个矩阵中添加行，每多一个预测因子就添加新的一行。下面我们将针对每一列进行讨论。

```

> summary(lm1)$coefficients
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -34.670621  2.6498030 -13.08423 6.950229e-34
rm            9.102109  0.4190266  21.72203 2.487229e-74

```

- **Estimate**栏包含估计系数（estimated coefficients），也就是截距（rm = 0时medv的值）和斜率（rm变化一个单位时，medv改变的值）。
- **Std. Error**栏包含了关于回归线（常用SE来标记）系数估计的标准偏差，用于衡量估计系数的差异性，也就是说，回归方程过大预示或过小预示的平均数量。这个值越小越好，但是这个值和回归系数有关。根据经验，这个值至少要比估计系数小一个数量级。在我们的例子中，rm变量的标准偏差是0.42，它差不多是估计系数9.1的二十二分之一。
- **t value**栏是回归系数估计的t统计量（t statistic，又名Student t检验）。它提供了一个分数，用于衡量这个变量的回归系数是否对模型有帮助。它显示了t检验与第一列中参数（截距和斜率）意义之间的关联。例如，t值为-13.08指的是（截距）t检验-34.671除以估值得标准差2.650。你可能不会使用这个值本身，但是你要知道它

用于计算假设几率和显著水平。

- $\Pr(>|t|)$ 称为变量的假设几率 (variable p-value) , 提供t检验 (在那个df (自由度) 下t分布的比例, 比t统计量的绝对值要大) 的假设几率。它是变量之间不相关的概率。你希望这个值越小越好。如果这个值很小, R会用科学计数法显示假设几率的数值, 就像这个例子中, 是 $6.95e-34$ (6.95×10^{34}) 。

这里用一个略微不同的符号来获取线性模型对象中回归系数组成的标准误差列。

```
> summary(lm1)$coefficients[, 2]
(Intercept)      rm
2.6498030  0.4190266
```

下一步, 我们将获得残差 (residuals) 。残差是预测变量的实际值 (medv) 和通过回归预测值之间的差值。对于大多数回归来说, 你希望残差在画图时呈现正态分布。如果残差是正态分布的, 意味着预测值和实际值之间的差异接近0, 这正是我们需要的。在线性模型中, residuals成分是一个数值向量, 它的长度与数据集中观测数目相同。在这个例子中是506。因为这个向量太长了, 我们在这里只展示前20个残差。

```
> summary(lm1)$residuals[1:20]
 1          2          3          4          5          6          7
 -1.175745 -2.174021  3.9719677  4.3740621  5.8178479  4.844060  2.848741
 8          9         10         11         12         13         14
 5.5924041 -0.083354 -1.078441 -8.373528 -1.123952  2.768301  0.922174
15         16         17         18         19         20
-2.615835  1.4689170  3.7496040 -2.351012  5.2095142  0.742842
```

下一步, 我们用r.squared推算决定系数 (R-squared) 。决定系数是用来评估模型拟合得好不好的标准。这个数值介于0和1之间。随着决定系数的增加, 拟合质量越来越好, 决定系数为1时, 拟合程度最好。用模型进行解释, 决定系数就是响应变量medv的总变差百分比。之所以称之为决定系数, 是因为在一元回归模型中, 它是预测因子和响应变量相关系数的平方。在我们的例子中, medv有48%取决于rm。但是请注意, 决定系数高标志着相关性好, 但相关性并不一定意味着因果关系。

```
> summary(lm1)$r.squared
[1] 0.4835255
```

接下来，我们将用sigma推导标准化残差（RSE）。RSE是残差的标准差。你希望这个值在数量上和残差成比例。对于正态分布来说，第一和第三分位数应该是 $1.5 \pm$ 标准差。当RSE刚好是0时，模型完美拟合数据（很可能是因为过拟合）。在使用summary()函数时，能得到自由度和RSE。自由度是训练集观测的数量和模型中使用的变量数量（截距算作一个变量）的差值，所以在这个例子中是 $506-2=504$ 。

```
> summary(lm1)$sigma    # RSE  
[1] 6.61616
```

最后，我们来反演F统计量。统计F检验法（statistical F-test）在模型上执行。它是长度为3的数值向量，包含F值的数值、模型中自由度的数目（也就是分子上的numdf）和剩余自由度（分母dendf）。这个检验获取了模型中的参数（在我们的例子中，只有一个参数），然后把它跟含有更少参数的模型进行比较。理论上来说，模型的参数越多，拟合程度应该越好。F统计量的值接近1，意味着响应变量和预测因子之间没有关系。另一方面，如果值超过1，意味着它们之间有关联。在例子中，这个值是472，所以我们推断medv和rm是关联的。在使用summary()时，F统计量包括DF，也就是自由度。这与模型中有多少变量有关。在我们的例子中，只有一个变量，所以自由度是1。同样也跟假定几率有关，当一个F检验具有高的假定几率时，模型的参数越多（你的模型），效果不会比参数少的模型好。

```
> summary(lm1)$fstatistic  
  value   numdf   dendf  
471.8467   1.0000 504.0000
```

在summary()的展示中，你会在回归系数表中看到一些显著星号(significance stars)，下一个表中你会看到显著编码说明(significance codes legend)。下面看我们例子中的显示：

```
Coefficients:  
            Estimate Std. Error t value Pr(>|t|)  
(Intercept) -34.671     2.650  -13.08  <2e-16 ***  
rm             9.102      0.419   21.72  <2e-16 ***  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '1
```

星号用于速记显著水平，根据计算得到的假设几率对应星号的数

目。*表示高显著性，*表示低显著性。在我们的例子中，表示medv和rm之间不太可能是没有关系的。参考显著性图例，变量旁边符号越多越好。空格是不好，点表示还不错，而更多的星号表示相当好。

我们可以用names(lm1)来发掘模型中的其他信息。在这个对线性回归的介绍中，我们主要关心fitted.values、coefficients和residuals。

```
> names(lm1)    # Show all calculations by lm()
[1] "coefficients" "residuals"    "effects"      "rank"
[5] "fitted.values" "assign"       "qr"          "df.residual"
[9] "xlevels"       "call"        "terms"       "model"
```

现在让我们看看几种使用拟合模型计算组件的方法。

有几种方法来使用各组件：

- 拟合值是由模型预测的，包括预测变量（特征变量）的值。你可以使用fitted(lm1)、lm1\$fitted.values或lm1\$fitted来获取模型的拟合值。线性模型的这一组件是一个数值向量，长度等于数据集中观测值的数目——在这个例子中是506。
- 回归系数是模型的估计参数。可以使用coef(lm1)、lm1\$coefficients或lm1\$coef获取模型的回归系数。线性模型的这一成分是数值型向量，它的长度等于预测因子的数目加1——在这个例子中等于2。
- 残差是响应变量的实际值（测量得到的）和预测值之间的差值。可以使用resid(lm1)、lm1\$residuals或lm1\$resid来获取模型的残差。线性模型的这一组件是数值向量，长度是数据集的观测值数目——在这个例子中是506。

就像上面一些方便的标记方法一样，你可以用coef(lm1)、lm1\$coefficients或者lm1\$coef展示线性模型的回归系数。注意到这里给出了两个值，截距和斜率。

```
> coef(lm1)
> lm1$coefficients
> lm1$coef
(Intercept)           rm
-34.670621     9.102109
```

使用R的数据描点功能，我们可以用下面的命令将线性模型可视化。使用attach()函数将Boston数据集放入R的搜索路径中，这样我们就不用在plot()和lines()函数中一直重复输入数据集的名字了。

```
> attach(Boston) # Attach data set so only variable name  
> plot(rm, medv, pch=20, xlab="Avg. # Rooms", ylab="Median Value")  
> lines(rm, lm1$fitted, lwd=3)
```

图5-1展示了rm和medv的数值对，还根据拟合值画了一条展示最佳拟合的回归线。

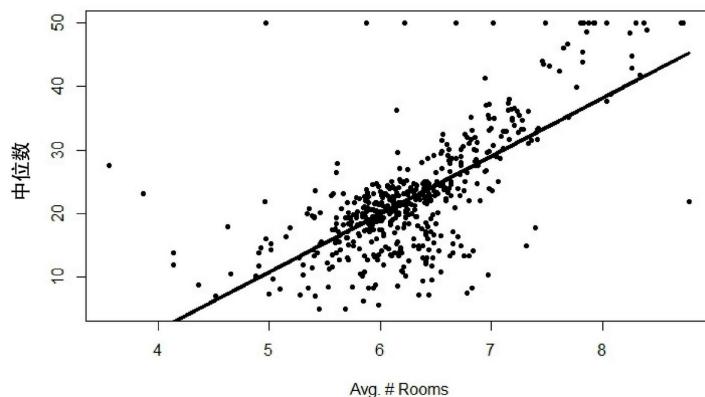


图5-1 变量medv和rm带有回归线的回归模型散点图

现在你可以使用回归系数训练模型，对新数据（不是来自Boston数据集的数据）进行预测。比方说，你想基于基于平均房间数——在这个例子中是6——预测房屋价值的中位数。在下面的代码中，使用coef(lm1)[1]获取第一个系数，然后把它与第二个系数coef(lm1)[2]乘以6相加。这里，我们看到预测的房屋价格中位数是\$19 942。

```
> coef(lm1)[1] + coef(lm1)[2]*6  
(Intercept)  
19.94203
```

这里有另一种方法来做预测，这次给线性模型lm1使用predict()函数。我们需要把数据框作为一个参数传递，带上rm的值为6。我们得到了跟之前一样的预测房屋价格中位数。

```
> newdata <- data.frame(rm=6)  
> predict(lm1, newdata)  
1
```

最后，让我们看一些诊断图，它们是线性模型和plot()函数结合提供的，如图5-2所示。在使用线性模型对象时，plot()提供了一套四个图表：残差-拟合值、标准化残差的Q-Q图、比例尺定位图（标准化残差的平方根-拟合值）和残差-杠杆比例（对应Cook距离0.5和1增加附加值）的散点图。

```
> par(mfrow=c(2,2)) # Split display into 2x2 panel grid
> plot(lm1)
```

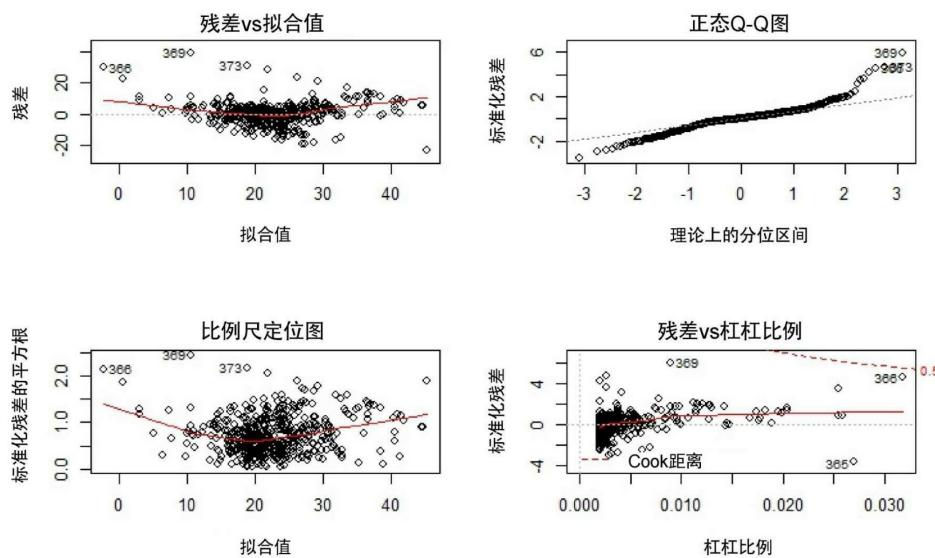


图5-2 medv和rm的线性回归模型诊断图

让我们仔细地看看这些图表，一名数据科学家可以根据这些来决定模型在预测时的行为。第一张图是标准的残差曲线图，展示了拟合值下的残差。一些趋于异常值的点在图中标注了出来（观察点366的值为29.75，点369的值为39.43，点373的值为31.19）。如果这张图上的点有明显的模式，那么线性模型可能是不合适的。第二张图是残差的正态百分位数图。我们希望看到残差是正态分布的。最后一张图展示了残差vs. 杠杆比率。在这张图中有标记的点表示我们可能想要审查的可疑情况，它们也许会对回归关系有不好的影响。点366就是这样一个值得后续深入检查的点。

下面，我们将展示第366条观测，它对应的medv=27.5, rm=3.56。

但是注意到，这条观测的拟合值是-2.258，所以残差高达29.758。模型的标准化残差可以使用summary(lm1)\$sigma来获取，然后可以用于计算第366条观测的标准化残差，这个值应该是比较小的。

```
> Boston[366,] # mdev=27.5 rm=3.56
  crim zn indus chas nox rm age dis rad tax ptratio
  366 4.55587 0 18.1 0 0.718 3.561 87.9 1.6132 24 666 20.2
  black lstat medv
  366 354.7 7.12 27.5
> lm1$fitted[366]
[1] -2.258
> lm1$residuals[366]
[1] 29.758
> summary(lm1)$sigma #RSE-residual standard error
[1] 6.61616
> lm1$residuals[366]/summary(lm1)$sigma
 366
4.497777
```

常用于计算影响的一个标准是Cook距离（Cook's Distance），如图5-3所示。下面的plot()函数可以使所有观测的Cook距离可视化。这个测量方法在执行回归分析时，能用于估计数据点的影响。特别地，它可以指出一些值得检查准确性的数据点，并告诉我们在数据集的哪些区域适合多获取一些数据点。Cook距离测量的是删除一个已有观测带来的影响。有很大残差（异常值）和/或有高杠杆比率（识别出那些远离对应的平均预测值的观测点）的数据点可能会扭曲回归的结果和精度。具有较大Cook距离的点需要在分析过程中执行更深入的检查。

```
> par(mfrow=c(1,1))
> plot(cooks.distance(lm1))
```

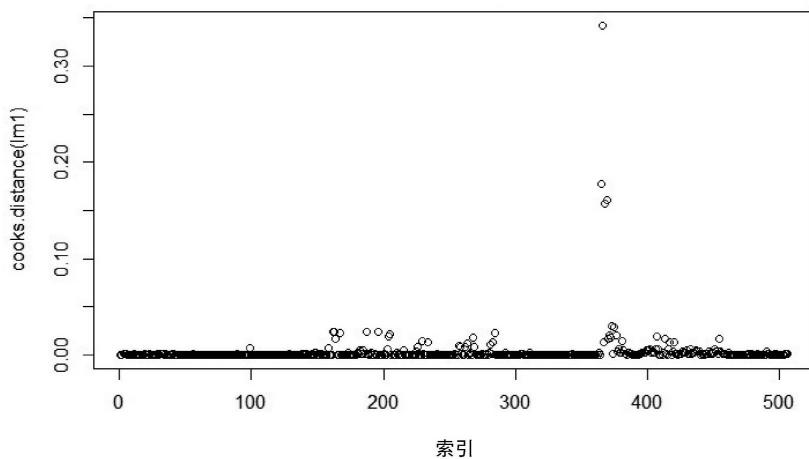


图5-3 线性模型的Cook距离图表

我们也可以用`residuals()`函数从线性回归拟合直接计算残差，如图5-4所示，y轴是残差，x轴是预测响应（拟合值）。这种类型的残差图也可以用于多元线性回归，我们将在下一节讲到。为了在一元线性回归中简化这个图表，你可以绘制基于预测（拟合）值下的残差。这个图可用于检测非线性、非均匀误差方差和异常值。任何刚好落在估计回归线上的数据点的残差都是0。因此，残差=0的线就相当于估计回归线。

一个“表现好”的残差图应该展现以下的行为：

- 残差应该在0线附近“随机跳动”。这意味着线性关系的假设是合理的。
- 残差在0线附近应大致形成一个“水平带”。这意味着误差方差是相等的。
- 在残差的基础随机模型中，没有一个残差是“突出”的。这意味着没有异常值。

总的来说，你希望残差-拟合值的图看起来像图5-4这样。只要记住，解释这些图是主观性很强的，所以不必着眼于每一个细微差别，过度解释一些很棘手的点。尤其要注意的是，遇到较小的数据集时，不要在残差-拟合值图表中倾注太多的注意力。有时候，数据集太小了，不值得花时间去解释残差-拟合值图。

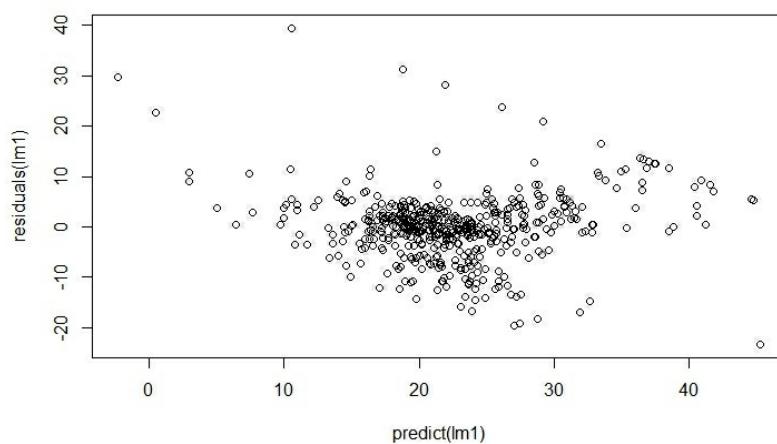


图5-4 线性回归拟合的残差-拟合值图

残差图是鉴别非线性的一个有效的可视化工具。理想情况下，残差图不会有明显的模式。基于lm1的残差图，可能会有证据证明数据之间存在非线性关系。

```
> par(mfrow=c(1,1))
> plot(predict(lm1), residuals(lm1))
```

在学习多元线性回归之前，考虑一些含有异常值的情况。数据点异常值是指响应变量的真实值和由线性模型预测出来的值相差很远。画出最小二乘回归线之后，你可以很容易检测到异常值。就响应变量和预测因子而言，它们明显远离回归线。你可以试着把包含异常值数据点的观测移除，看看对回归系数和生成的回归线有什么影响。虽然移除异常值对于最小二乘法拟合来说几乎没有效果，它能给标准化残差和R²带来显著的变化。你也可以用残差图来鉴定异常值，然后谨慎地决定残差多大时可以把一个点看做是异常点。如果你认为异常值是因为数据收集过程中的差错导致的，那么也可以考虑移除这条观测。

5.2 多元线性回归

尽管一元线性回归常常能发挥作用，但是它只能基于一元特征变量做预测。然而，现实生活中的问题通常有多个预测因子。现在让我们扩展线性回归的概念，学习多元特征变量。

我们可以像在一元线性回归案例中那样，使用最小二乘法来找到使得残差的平方和最小的回归系数。我们将再次使用R中的lm()算法来找到多元最小二乘回归系数估计。通过前面小节学习的计算线性模型数据的方法，R可以轻易地处理多元线性回归案例。

在执行多元线性回归时，我们通常需要回答一些关于模型的重要问题：（1）响应变量和预测因子之间有关系吗？（2）哪些是最重要的预测因子？（3）考虑一组预测值，哪些响应变量应该被预测，预测的精度是多少？还有（4）这个模型拟合数据的程度有多好？

我们可以选取car包中的Prestige数据集，看一个多元线性回归的例子。我们先安装car包，然后加载库和Prestige数据集。

```
> install.packages("car")
> library(car)
> data(Prestige)
```

像往常一样，使用summary()和head()函数来熟悉数据集。

```
> summary(Prestige)
   education      income       women      prestige
Min.    : 6.380  Min.   : 611  Min.   : 0.000  Min.   :14.80
1st Qu.: 8.445  1st Qu.: 4106  1st Qu.: 3.592  1st Qu.:35.23
Median :10.540  Median : 5930  Median :13.600  Median :43.60
Mean    :10.738  Mean   : 6798  Mean   :28.979  Mean   :46.83
3rd Qu.:12.648  3rd Qu.: 8187  3rd Qu.:52.203  3rd Qu.:59.27
Max.    :15.970  Max.   :25879  Max.   :97.510  Max.   :87.20
   census        type
Min.   :1113  bc :44
1st Qu.:3120  prof:31
Median :5135  wc :23
Mean   :5402  NA's: 4
3rd Qu.:8312
Max.   :9517
> head(Prestige)
   education income women prestige census type
gov.administrators 13.11 12351 11.16     68.8 1113 prof
```

general.managers	12.26	25879	4.02	69.1	1130	prof
accountants	12.77	9271	15.70	63.4	1171	prof
purchasing.officers	11.42	8865	9.11	56.8	1175	prof
chemists	14.62	8403	11.68	73.5	2111	prof
physicists	15.64	11030	5.13	77.6	2113	prof

这个Prestige数据集包含102条加拿大职业的观测，有6个特征变量。我们将使用多元线性回归以及作为预测因子的其他特征变量来预测响应变量prestige，这是一个基于社会调查的职业得分。

理解接下来的代码很重要，因为它展现了机器学习的主要部件——定义算法使用的训练集（`training set`）和测试集（`test set`）。把数据集拆分成训练部分和测试部分是算法学习的方法。将含有已知响应变量的数据集拆成两个部分。训练集用于通过计算回归系数训练算法。然后在测试集上使用训练好的模型来预测已知的响应变量。最后一步是将预测响应和观测到的（真实）响应作对比，看看它们之间有多接近。得到的差异是测试误差的衡量标准。基于测试误差，你可以返回去对模型进行优化，然后重复这些过程。一旦得到满意的模型，你就可以用全新的数据来看模型的“推广”能力。

让我们在这里花一些时间，更深入地钻研推广的概念。在机器学习中，你需要考虑“样本中”误差和“样本外”误差的概念。样本误差是在构建预测因子时，相同数据集带来的误差率。这种误差有时候称为再代入误差（`resubstitution error`）。样本误差通常都比新数据集中的误差小一些。这是因为在处理特定的训练数据集时，你的预测因子有时会调整噪音。所以在尝试一个新的数据集时，噪音会变得不同，会造成精度下降。与此相反，样本外误差（`out of sample error`）是你在使用新数据集时的误差率。有时这称为泛化误差。这个度量标准能够真实估计机器学习在新数据上表现得多好。样本外误差是你要关心的数值，而样本误差通常都小于样本外误差。

在拆分数据集时，通常要将大部分观测放入训练集。剩下的就是测试集。数据科学家通常六四分。在下面的代码中，我们将创建两个新的数据框，`trainPrestige`和`testPrestige`。同时，由于变量`prof`中有一些NA值，我们也会从Prestige数据集中移除这些观测。

```
> Prestige_noNA <- na.omit(Prestige)
> n <- nrow(Prestige_noNA) # Number of observations
> ntrain <- round(n*0.6) # 60% for training set
```

```

> set.seed(333)      # Set seed for reproducible results
> tindex <- sample(n, ntrain) # Create an index
> trainPrestige <- Prestige_noNA[tindex,] # Create training set
> testPrestige <- Prestige_noNA[-tindex,] # Create test set

```

让我们在训练集上做一些快速的探索性数据分析，以便更进一步熟悉数据。可以基于多个其他预测因子绘制响应变量的图表。如图5-5所示，变量education展现了一个趋势。

```

> plot(trainPrestige$prestige, trainPrestige$education) #Trend
> plot(trainPrestige$prestige, trainPrestige$income)   #No trend
> plot(trainPrestige$prestige, trainPrestige$women)    #No trend

```

在下面的代码中，在线性模型函数lm()中使用trainPrestige来进行线性回归，用于计算回归系数和其他模型参数。注意在lm()中使用的公式是prestige~.这种形式，这表示将其他所有变量都当做预测因子。summary()函数包含了每一个预测因子的结果，就像一元线性回归案例中的一元预测因子一样的形式。特别地，有一个关联的假设几率展现了每一个估算系数的置信度。例如，没有标记任何星号的特征完全可以忽略。在我们研究的模型中，education和income对预测prestige有很大的影响。

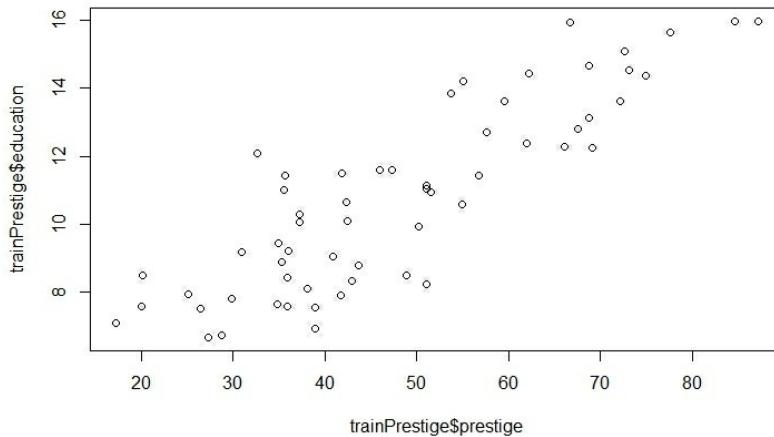


图5-5 训练集的探索性数据分析展现了一个趋势

```

> lm2 <- lm(prestige~., data=trainPrestige)
> summary(lm2)
Call:

```

```

lm(formula = prestige ~ ., data = trainPrestige)
Residuals:
    Min      1Q  Median      3Q     Max 
-13.7864 -4.0290  0.8807  4.5369 16.9482 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -1.544e+01 9.901e+00 -1.560 0.12492  
education    4.562e+00 8.320e-01  5.483 1.24e-06 *** 
income       9.607e-04 3.204e-04  2.999 0.00415 **  
women        7.252e-03 4.543e-02  0.160 0.87379  
census        1.031e-03 7.390e-04  1.396 0.16876  
typeprof     5.981e+00 5.773e+00  1.036 0.30495  
typewc       -1.137e+00 3.962e+00 -0.287 0.77531  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
Residual standard error: 7.145 on 52 degrees of freedom
Multiple R-squared:  0.8406, Adjusted R-squared:  0.8222 
F-statistic: 45.71 on 6 and 52 DF,  p-value: < 2.2e-16

```

一旦得到一个拟合后的线性模型，你应该运行一些诊断图来完全理解结果。一张重要的图表是拟合值（模型对训练集的预测值）-残差（拟合模型后剩下的变动幅度），用来确认在0残差线附近的分布情况。图5-6展示了一个云分布，证实了这个模型是正确的。因为残差图中几乎没有模式，所以模型中没有非线性的成分。如果你发现这个图中有一群明显的异常值，你可能需要探索一个或多个预测因子，才能解释这些异常值。

```
> plot(lm2$fitted, lm2$residuals)
```

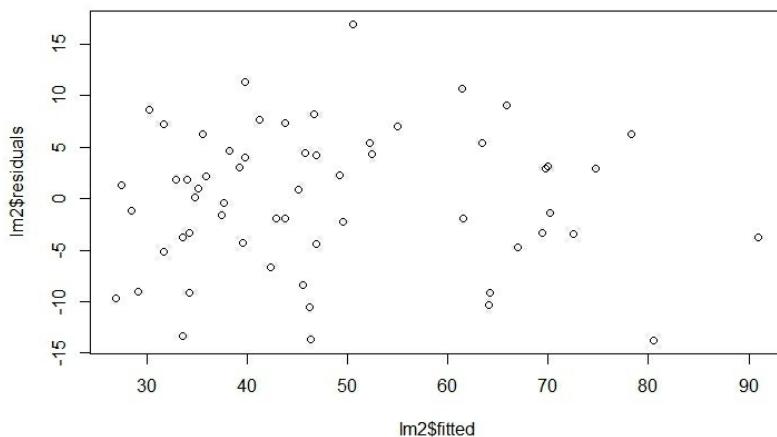


图5-6 预测值-残差图确认分布得很好

另一种诊断技术是绘制残差-索引图。索引指的是数据的行号，表明数据创建的特定顺序，用于检查是否有与索引相关的趋势。我们可以用下面的plot()命令来执行这个操作。图5-7展示的是没有趋势的情况。

如果你发现了行号的明显趋势或是一组异常值（例如，在一些可识别的索引下，一些残差聚成一团），这可能意味着模型漏掉了预测因子。总的来说，残差（真实值和拟合值之间的差值）图不应该跟数据集中观测的顺序有任何关系，除非数据行排列的顺序跟时间、年龄或者其他连续变量相关。

```
> plot(lm2$residuals, pch=19)
```

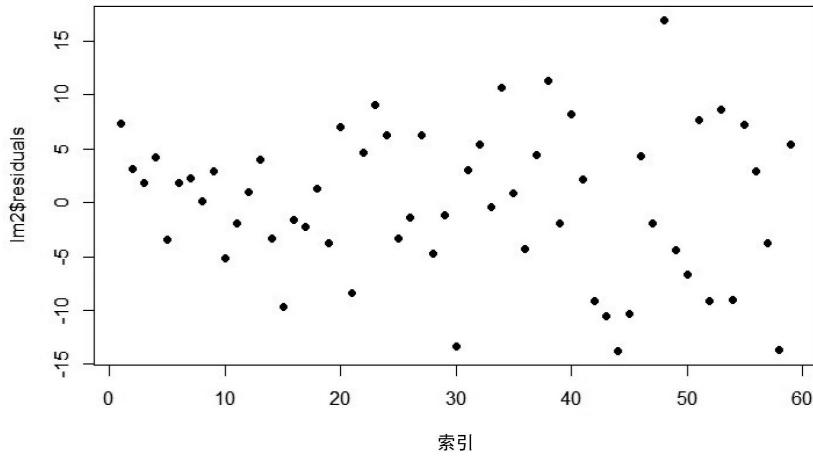


图5-7 索引图展示了没有趋势

接下来，我们用testPrestige来预测响应变量prestige（它的值是已知的）。我们的想法是预测一些已知的变量来确定测试误差，这样当你使用响应变量未知的新数据时，预测会更精确。

```
> predict2 <- predict(lm2, newdata=testPrestige)
```

为了看看我们对测试集的预测有多准确，可以使用cor()函数来建立统计相关性。这里，我们将prestige的预测值和实际值作对比，得到的值是0.915，它很接近1，所以我们可以推断它们肯定是成线性相关。

```
> cor(predict2, testPrestige$prestige)
[1] 0.9151361
```

而另一种诊断技术，是使用`qqnorm()`和`qqline()`函数展示分位数-分位数（Q-Q）图，来确认残差是否成正态分布。图5-8展示了残差是如何匹配正态分布的。Q-Q图将样本数据分布（在这个例子中，线性模型的残差）和理论分布（这里是一个正态分布）进行对比。如果残差分布方式跟正态分布一致，所有的点都将落在一条从左下到右上的45°直线上。线的系统偏差代表著违背正态分布，例如重尾或是歪斜。如果残差是正态分布的，它们应该在接近线的周围分散。

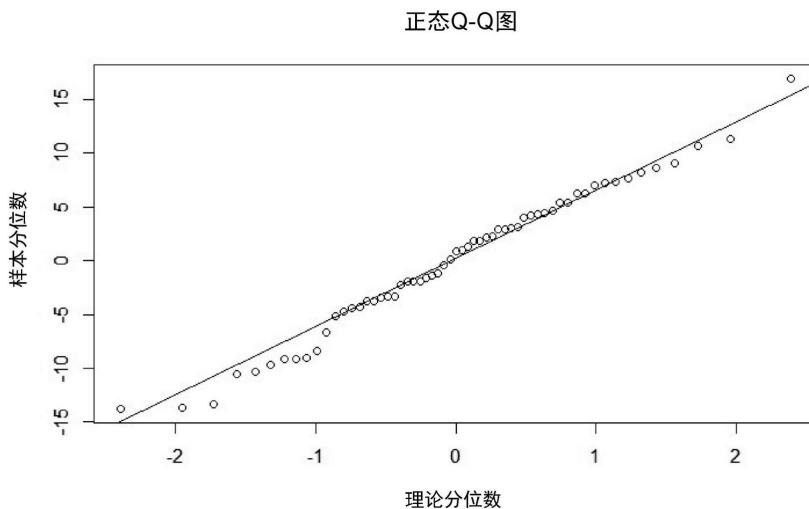


图5-8 Q-Q图展示了残差和正态分布的吻合程度

记住，残差是实验误差的估计，通过从预测响应减去观测响应获得。检查残差是所有机器学习的关键部分。通过认真观察，残差可以告诉我们假设是否是有道理的，选择的模型是否是合适的。残差可以被当做是不能通过拟合模型解释的变异元素。因为这是一种误差形式，所以我们希望它们（大概地）成正态分布，并（近似地）成均值为0、方差恒定的独立分布。

```
> rs <- residuals(lm1)
> qqnorm(rs)  # Quantile-quantile plot
> qqline(rs)
```

作为一种后期检查，我们创造的最后一一种诊断多元线性回归模型的方法是用模型中未使用的变量生成一张图表。在下面的代码中，生成的图表如图5-9所示。我们绘制了测试集中变量prestige的实际值-预测值图。理想状态下，我们希望看到一条45°的直线，prestige值与我们预测

的完全相等（虽然通常情况下都不是这样的）。在测试集中，你可以尝试鉴定一些可能之前错过的趋势，所以我们在图中加入了一个新的变量，也就是type。在这里，type是一个分类变量，表示职业的类型。它有以下几种可能的值：bc、prof和wc。

```
> plot(testPrestige$prestige, predict2, pch=c(testPrestige$type))
> legend('topleft', legend=c("bc", "prof", "wc"), pch=c(1,2,3),
  bty='o')
```

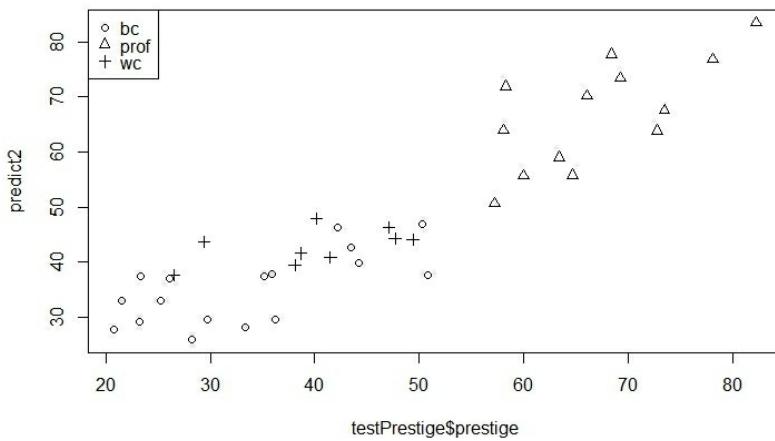


图5-9 用于分析多元线性回归模型的最终诊断图

5.3 多项式回归

另一种有效的监督机器学习的工具叫多项式回归，通过添加额外的预测因子来扩展线性模型。这些额外的预测因子通过提升每个原始预测因子的能力来获取。这个算法能处理响应变量和预测因子之间的非线性关系。在对这种非线性关系进行探索性数据分析时，你可能发现散点图揭示的模式看起来不像是线性的。如果是这种情况，多项式回归可以用于探索不同曲度等级下的预测因子。就像我们下面看到的，多项式回归以一种分层的方式使用，使得模型最能代表数据集。例如，我们使用Boston数据集，然后检查图5-10的散点图，图中包含了两个变量：nox和dis。这个图显示这些数据可能不是线性的。显而易见，当dis值介于6和12之间时，对应的nox值相对平缓。然而，当dis值小于6时，nox呈现出非线性变化。这些特征说明数据是曲线型的。

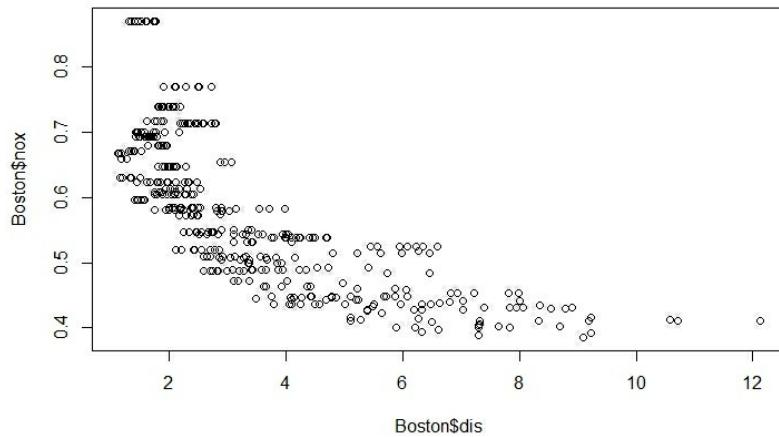


图5-10 散点图显示数据是曲线型

处理这种关系的常用方法是用一个预测变量 X 考虑统计回归情形。用三个变量 X 、 X^2 和 X^3 作为预测因子，扩展成三次回归。这叫做三次多项式。多项式逼近提供了一种非线性拟合数据的简便方式。因为很多统计模型仅仅是近似，原则上来说，你可以通过提升拟合多项式的次数来获得更精确的模型。然而，在某一时刻模型会过拟合（这是第7章的主题），所以使用次数高的多项式预测一些新的、未来的数据，事实上会

导致结果恶化。

对于足够高的次数来说，多项式回归能帮助我们生成极端的非线性曲线。然而一般来说，使用超过三次或四次的多项式是很罕见的，因为多项式的次数越高，生成的曲线会过于灵活，呈现出一些很奇怪的形状，可能会过于吻合训练集。

让我们使用Boston数据集来看一个多项式回归的例子。为了用dis的一次、二次、三次多项式来预测nox，先用lm()函数来拟合一个线性模型。一次多项式就是一元线性回归，回归曲线图是一条直线，如图5-11所示。

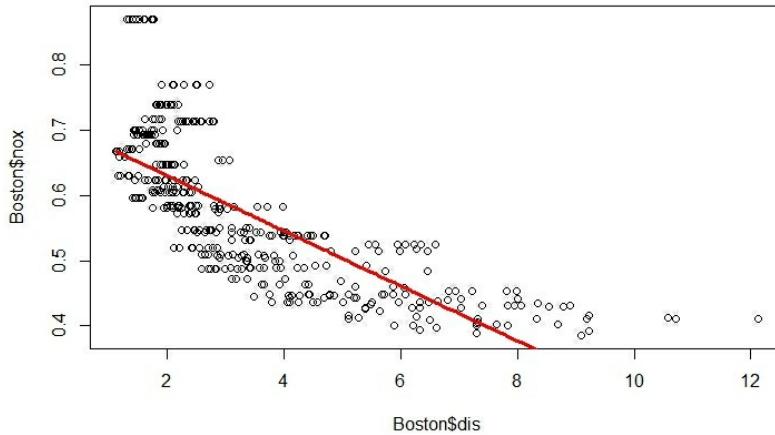


图5-11 一次多项式回归曲线——线性拟合

很显然，nox和dis之间存在显著的关系，但是这个关系看起来是非线性的，数据展示了曲线关系。

```
> library(MASS)
> data(Boston)
> names(Boston)
> fit_d1 <- lm(nox~dis, data=Boston)
> summary(fit_d1)
Call:
  lm(formula = nox~dis, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.12239 -0.05212 -0.01257  0.04391  0.23041 

Coefficients:
```

```

Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.715343 0.006796 105.26 <2e-16 ***
dis -0.042331 0.001566 -27.03 <2e-16 ***
...
Signif. codes: 0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
Residual standard error: 0.07412 on 504 degrees of freedom
Multiple R-squared: 0.5917, Adjusted R-squared: 0.5909
F-statistic: 730.4 on 1 and 504 DF, p-value: < 2.2e-16
> plot(Boston$dis, Boston$nox)
> lines(Boston$dis, fit_d1$fitted, col=2, lwd=3)

```

对于更高次数的回归方程式，我们将使用poly()函数来提升dis的次数。使用poly(dis, 2, raw=TRUE)可以生成一个长度为506的二次矩阵，这能产生二次多项式。参数raw=TRUE表示使用原始的非正交多项式。

另一种创造多项式基函数的方法是使用R中的I()包装函数，就像dis + I(dis^2)这样，来得到一个等价多项式。I()函数是必要的，因为公式中使用的插入符号没有常规的数学意义，即自乘。不论哪种情况，我们都创造了一个非线性拟合。熟悉的非线性二次型曲线如图5-12所示。

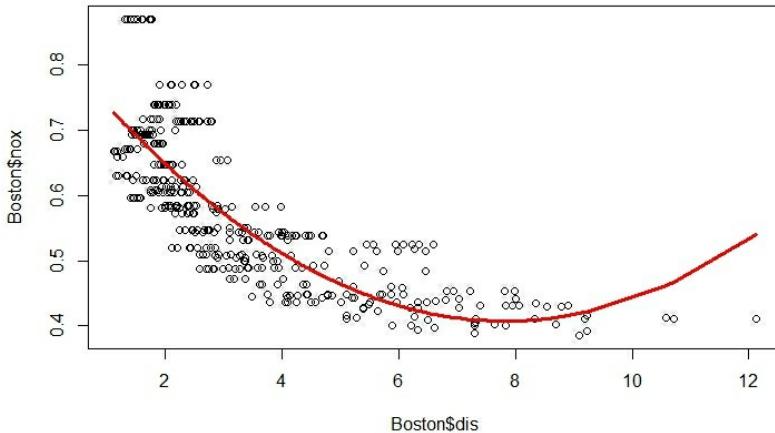


图5-12 二次多项式回归曲线——二次拟合

注意，二次拟合比只有线性项的拟合效果好一点。同时也要注意，二次拟合的R²值为0.6999，作为对比，线性拟合的R²值为0.5917。

```

> fit_d2 <- lm(nox ~ poly(dis, 2, raw=TRUE), data=Boston)
> summary(fit_d2)
Call:
lm(formula = nox ~ poly(dis, 2, raw = TRUE), data = Boston)
Residuals:

```

```

      Min        1Q     Median       3Q      Max
-0.129559 -0.044514 -0.007753  0.025778  0.201882

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.843991  0.011196  75.39 <2e-16 ***
poly(dis, 2, raw = TRUE)1 -0.111628  0.005320 -20.98 <2e-16 ***
poly(dis, 2, raw = TRUE)2  0.007135  0.000530  13.46 <2e-16 ***
---
Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
Residual standard error: 0.06361 on 503 degrees of freedom
Multiple R-squared:  0.6999, Adjusted R-squared:  0.6987
F-statistic: 586.4 on 2 and 503 DF,  p-value: < 2.2e-16
> plot(Boston$dis, Boston$nox)
> lines(sort(Boston$dis), fit_d2$fitted.values[order(Boston$dis)],
  col = 2, lwd = 3)

```

现在让我们用三次多项式拟合线性模型。正如上面所说，一图胜千言，基于数据的多项式拟合生成的非线性曲线图表，是评估拟合质量的一种强大方式。图5-13展示了优良的拟合，但是让我们看看lm()返回的每一个多项式次数下的统计量。

- 拟合的决定系数（也被称为R²或是R平方值）表明用模型解释的数据集中异常值的百分比。R²越接近1，拟合模型越能完整地“解释”数据。注意到，随着多项式次数的提高，调整后的决定系数也提高：0.5909、0.6987和0.7131。
- 拟合的F-统计量（F-statistic）比较模型解释的数据变异分数和模型无法解释的数据变异分数的“大小”。这一比较的基础是模型变异和误差（残差）的比值。F-统计量的值“大”，通常> 6，表示拟合是有意义的。
- 回归系数的t-统计量（也称为“t值”）表示任何一个回归系数的显著性水平。t-统计量定义为回归系数和标准误差的比值。如果一个回归系数的t值为2或者更高，那么这个回归系数通常是显著的。

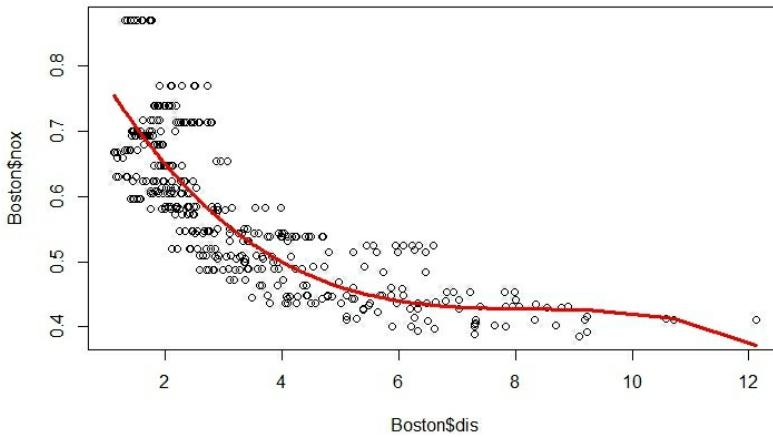


图5-13 三次多项式回归曲线——三次拟合

```

> fit_d3 <- lm(nox~poly(dis, 3, raw=TRUE), data=Boston)
> summary(fit_d3)
> coef(summary(fit_d3))
Call:
lm(formula = nox~poly(dis, 3, raw = TRUE), data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.121130 -0.040619 -0.009738  0.023385  0.194904 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.9341281  0.0207076 45.110 < 2e-16 ***
poly(dis, 3, raw = TRUE)1 -0.1820817  0.0146973 -12.389 < 2e-16 ***
poly(dis, 3, raw = TRUE)2  0.0219277  0.0029329   7.476 3.43e-13 ***
poly(dis, 3, raw = TRUE)3 -0.0008850  0.0001727  -5.124 4.27e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06207 on 502 degrees of freedom
Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131 
F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16

> plot(Boston$dis, Boston$nox)
> lines(sort(Boston$dis), fit_d3$fitted.values[order(Boston$dis)],
  col = 2, lwd = 3)

```

我们可以看到，在方程中加入更高阶的项能提高模型水平，那么我们为什么不继续增加更多的更高阶项呢？高阶拟合可能会导致曲线波动过多，也就是说，我们不清楚加入额外的项是否能对数据进行更好的拟合。

在使用多项式回归时，必须确定多项式的次数。除了上面列出的统计规律之外，我们也可以使用假设检验。在我们的例子中，我们有从线性到三次多项式的拟合模型，力图用最简单的模型来最有效地解释nox和dis之间的关系。我们可以使用anova()函数来比较三个模型的假设检验。方差分析（analysis of variance）用于测试模型M₁是否能有效解释数据（零假设），或者是否需要一个更复杂的M₂模型（择一假设）。

在调用anova()时，我们给模型传递逐渐提高复杂度的参数：fit_d1、fit_d2和fit_d3。生成的方差分析表展现了不同模型对照的结果。在这里，假设几率是有用的。我们从头到尾仔细研究Pr(>F)列的值。在第二行中，即Model 2，如果假设几率的值大于0.05，那么我们不接受零假设：通过在多项式中加入其他项，“模型没有显著改善”。在这种情况下，值非常小，2.2e-16，意味着有改善。下一步，我们来到第三行，即Model 3，重复这个处理过程。再次，我们看到在一个很小的假设几率下，效果有提升，这意味着三次多项式对于这些非线性数据来说有最好的拟合效果，它能提供最强的预测能力。

```
> anova(fit_d1, fit_d2, fit_d3)
Analysis of Variance Table
Model 1: nox~dis
Model 2: nox~poly(dis, 2, raw = TRUE)
Model 3: nox~poly(dis, 3, raw = TRUE)
  Res.Df   RSS Df Sum of Sq    F    Pr(>F)
1     504 2.7686
2     503 2.0353  1  0.73330 190.329 < 2.2e-16 ***
3     502 1.9341  1  0.10116  26.255 4.275e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' 1
```

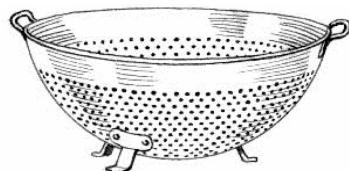
5.4 小结

本章起着使用回归工具进行监督学习前的启动点的作用。回归是历史最悠久也是人们理解最深入的算法，它在众多领域都有着广泛的应用。在本章中，我们讨论了3种回归形式：一元线性回归、多元线性回归和多项式回归。在第6章中，注意力将转向分类方法。在后面的第7章中，我们会探索评估统计学习算法（包括回归）的工具。但这不是回归的结束。还有其他类型的回归算法，例如岭回归（ridge regression）和套索（lasso）等收缩（shrinkage）方法，用于降低回归系数趋近0时产生的误差。不过，这些回归形式超出了本书的范围。

下面是本章小结：

- 我们从使用R中的`lm()`函数提供的最小二乘计算工具开始，探索一元线性回归。
- 在`summary()`函数的帮助下，我们评估了一系列统计措施，例如模型的拟合值、回归系数、残差、标准误差、t值、假设几率和F-统计量。
- 我们发现一旦训练了模型，就可以用新数据进行预测。
- 我们探索了一系列评估模型拟合质量的重要图表：残差-拟合值图、Q-Q图、Cook距离图等。
- 然后我们学习了如何将一元线性回归算法扩展到使用多元线性回归的多元预测因子。在这里，我们使用了一种机器学习常用的技术，将数据集拆分成训练集和测试集。
- 我们也用一元线性回归中的相同方法来评估模型的拟合程度，同时也看了一些新的诊断图，例如索引图。
- 为了处理响应变量和预测因子之间的非线性关系，我们的注意力转向了多项式回归，并探索了不同阶次的多项式如何提升拟合的质量。

第6章 分类



在本章中，我们会继续探索使用机器学习工具解决数据科学问题的方法。我们在第5章学习了回归方法，作为扩展，现在我们要转向另一个非常重要的监督学习类型：分类（Classification）。在线性回归中，我们假定响应变量（要预测的变量）是定量的。但是在很多数据科学问题中，响应变量是定性的——也被称作分类变量。例如，性别是定性的，它的值是男性或者女性。预测定性响应的过程称为分类。

有很多常用的分类方法，或者称为分类器，数据科学家可以用来预测分类响应变量。在本章中，我们将通过讨论最常用的一些分类算法来大大地扩展你的机器学习工具箱，这些方法包括：

- 逻辑回归；
- 分类树；
- K-最近邻；
- 支持向量机；
- 神经网络；
- 朴素贝叶斯；
- 随机森林；
- 梯度提升树。

就像我们在第5章讨论回归一样，我们将使用训练集来建立一个分类器。我们的目标是分类器在训练集上能表现出色，并且用没有训练过的测试集上也得到正确的结果。

6.1 一个简单的例子

让我们从一个非常简单的例子开始讨论分类。使用kernlab包中的spam数据集来建立并测试一个垃圾邮件分类算法，就跟现如今大多数邮件应用里包含的功能一样。spam数据集包含了若干变量，代表一封邮件中出现特定词语和标点的频率。举个例子，如果邮件中出现了“免费”“贷款”或是“钱”等字眼，那么这可能是一封垃圾邮件。spam数据集包含了4 601条观测记录（个人电子邮件）以及57个特征变量加上一个分类标签（响应变量）type，type的值为“垃圾邮件”或是“非垃圾邮件”。

在下面的R代码中，我们会考虑字符“\$”出现的频率（利用charDollar特征变量），并使用密度图来进行探索性数据分析。包含“\$”通常是证明垃圾邮件的一个条件。在图6-1中，x轴表示“\$”在邮件中出现的频率，y轴表示密度，即邮件中出现特定频率的次数。

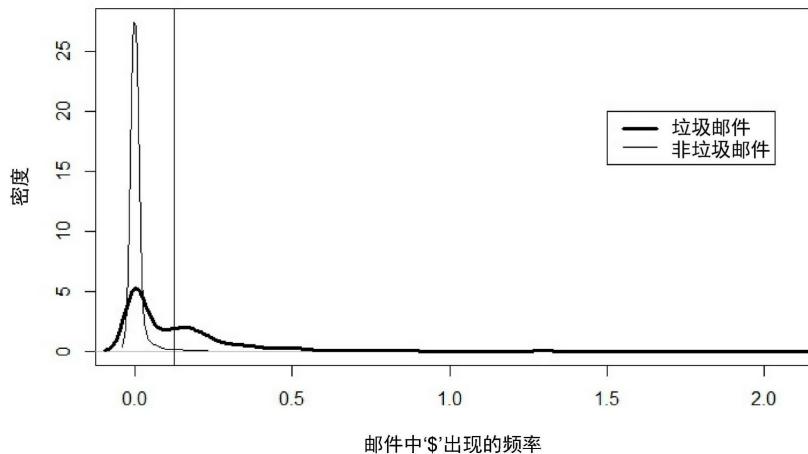


图6-1 带有边界值的基于分类目的的垃圾邮件分类器的密度图

大多数的垃圾邮件中出现了粗实线，通常也会出现更多字母“\$”。另一方面，我们希望收到的非垃圾邮件的峰值接近0，这意味着这些邮件中出现“\$”的次数更少。更进一步，含有大量“\$”字符的邮件是非垃圾邮件的概率是相当小的。

```
> plot(density(spam$charDollar[spam$type=="nonspam"]), lwd=0.5,  
main="", xlab="Frequency of '$' in E-mail")  
> lines(density(spam$charDollar[spam$type=="spam"]), lwd=3)
```

```
# > 0.125 -> spam, <= 0.125 -> non-spam  
> abline(v=0.125, col="black")  
> legend(1.5,20,legend=c("Spam","Nonspam"), lwd=c(3,0.5), lty=1)
```

我们可以开发一个简单的算法来定义一个边界，如果“\$”出现的频率大于这个边界值，那么我们预测这是一封垃圾邮件。否则，预测这是一封非垃圾邮件。让我们回到探索性图表，看看使用的数据。我们选择了0.125作为频率边界值。大于0.125，判定为垃圾邮件；小于0.125判定为非垃圾邮件。在图6-1中，我们可以看到非垃圾邮件的尖峰小于0.125，而垃圾邮件的其中一个大尖峰大于边界值。

现在我们可以评估这个简单的分类算法，看看它的预测效果有多好。为此，我们为spam数据集中的每一封邮件都做了预测。在下面的代码中，基于选择的边界值，生成的spam_classifier向量包含了数据集中每一封邮件的分类，即“垃圾邮件”或是“非垃圾邮件”。然后，我们使用spam_classifier向量为这些分类做了一个表格，并除以数据集中的观测数目。这个表格可以这样解释：当一封邮件是非垃圾邮件时，我们的分类准确度是59.1%；当邮件是垃圾邮件时，准确度是16.6%。所以这个算法的总体准确度是59.1 + 16.6，也就是说，这个算法75.7%的概率是准确的。这就是我们评估简单分类器算法的方法，虽然这是对综合误差的一个乐观估计。

```
> spam_classifier <- ifelse(spam$charDollar > 0.125, "spam",  
"nonspam")  
> table(spam_classifier, spam$type)/nrow(spam)  
spam_classifier   nonspam      spam  
nonspam           0.5911758  0.2279939  
spam              0.0147794  0.1660509
```

6.2 逻辑回归

我们要探索的第一个分类算法叫做逻辑回归（logistic regression）。这个监督学习工具本质上用于转化线性回归的输出，以便它能拟合到二元响应中。换句话说，它模拟了二元输出的“概率”。所以逻辑回归针对响应归属于特定分类的概率进行建模，而不是直接对响应变量进行建模。像6.1节提到的垃圾邮件问题一样，很多问题都适合这种表述方法——二元响应，诸如垃圾邮件或者非垃圾邮件，网上交易是否是欺诈，或者一个肿瘤是否是恶性的。换言之，逻辑回归通常适用于响应变量是二元的、预测变量是定量的，并与响应变量的概率或者几率有联系的情况。这种算法也可以用于分类变量和多元预测因子的情况。逻辑回归是用于解决分类问题的、最流行的机器学习工具。

作为一个案例，让我们在iris数据集上使用逻辑回归算法。我们尝试预测一个观测结果是否属于“杂色”（也就是说，杂色或是纯色）。在下面的代码中，我们从生成供算法使用的训练集train_iris和测试集test_iris开始，来解决这个问题。我们在本章中会一直使用这两个数据集。鉴于逻辑回归利用了二元响应，我们需要在iris数据框中创建新的一列isVersicolor。这一列根据观测数据是否是“杂色”，填写true或是false。

```
> data(iris)
> n <- nrow(iris) # Number of observations
> ntrain <- round(n*0.6) # 60% for training set
> set.seed(333) # Set seed for reproducible results
> tindex <- sample(n, ntrain) # Create an index
> train_iris <- iris[tindex,] # Create training set
> test_iris <- iris[-tindex,] # Create test set
> newcol <- data.frame(isVersicolor=(train_iris$Species==
"versicolor"))
> train_iris <- cbind(train_iris, newcol)
> head(train_iris)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	isVersico
71	5.9	3.2	4.8	1.8	versicol	TRUE
13	4.8	3.0	1.4	0.1	setosa	FALSE
145	6.7	3.3	5.7	2.5	virginica	FALSE
84	6.0	2.7	5.1	1.6	versicolor	TRUE
3	4.7	3.2	1.3	0.2	setosa	FALSE
105	6.5	3.0	5.8	2.2	virginica	FALSE

现在，我们使用拟合广义线性模型的glm()函数，广义线性模型是包含逻辑回归的一类模型。glm()是stats包的一部分，该包是基础R中自带的，所以无需安装。glm()的语法跟lm()差不多，我们已经很熟悉了。首先，指定算法中运用的公式。在这个例子中，响应变量是我们刚创建的二进制变量isVersicolor。接下来，用一元预测因子Sepal.Width调用glm()。在生成的模型表中，第一个参数是Sepal.Width和isVersicolor图的截距，第二个参数是对应的斜率。在这里，我们看到对象glm1得到了一个广义线性模型，在二项式误差分布的帮助下，isVersicolor建模为一个一元定量预测变量Sepal.Width的函数。

```
> glm1 <- glm(isVersicolor~Sepal.Width, data=train_iris, family=binomial)
> summary(glm1)
Call:
  glm(formula = isVersicolor~Sepal.Width, family = binomial,
       data = train_iris)
Deviance Residuals:
    Min      1Q  Median      3Q      Max 
-1.9933 -0.8609 -0.4757  0.9359  2.1143 
Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  9.1013    2.5534   3.564 0.000365 ***
Sepal.Width -3.3010    0.8656  -3.813 0.000137 ***  
---
Signif. codes:  0 '****'0.001 '**'0.01 '*'0.05 '. '0.1' '1
(Dispersion parameter for binomial family taken to be 1)
Null deviance: 113.136 on 89 degrees of freedom
Residual deviance: 90.326 on 88 degrees of freedom
AIC: 94.326
Number of Fisher Scoring iterations: 5
```

观察响应变量和预测变量的散点图是很有趣的。接着，使用predict()函数在广义线性模型上叠加另一条曲线。图6-2描绘了算法的二叉树特性。这叫做sigmoid函数曲线。注意，y轴上isVersicolor的值介于0到1之间，这代表了训练集中不同的Sepal.Width值对应的概率。

```
> plot(train_iris$Sepal.Width, train_iris$isVersicolor)
> curve(predict(glm1, data.frame(Sepal.Width=x), type=
  "response"), add=TRUE)
```

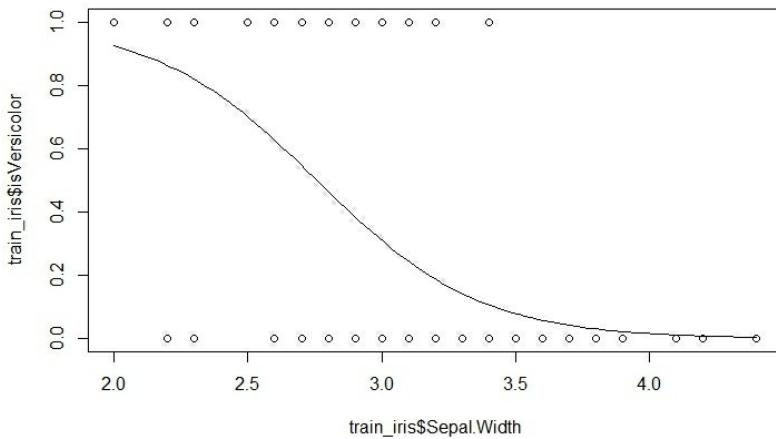


图6-2 逻辑回归图展示了sigmoid函数的形状

最后，我们可以使用predict()函数来决定一个新观测的分类概率。我们将预测Sepal.Width值为2.4的分类概率。在这个例子中，响应变量是Versicolor类型的概率为76.4%。

```
> newdata <- data.frame(Sepal.Width=2.4)
> predict(glm1, newdata, type="response")
  1
0.7647645
```

接下来，我们可以在逻辑回归中使用多元预测因子。这次的预测因子是数据集中的4个定量变量。在调用glm()时，我们会传递方程、训练集以及和前面相同的参数family=binomial，以便让R执行逻辑回归（还有其他可供选择的广义数据模型）并创建glm2对象。

接着，我们可以使用summary(glm2)来研究算法的结果。模型表看起来跟多元回归一样。可以看到，Sepal.Width关联到最小的假设几率，这意味着Sepal.Width和isVersicolor之间存在关联。也可以使用coef()函数来获取这个拟合模型的回归系数，用summary()函数获取这个拟合模型的组件，例如回归系数的假设几率。

```
> formula <- isVersicolor ~ Sepal.Length + Sepal.Width +
  Petal.Length + Petal.Width
> glm2 <- glm(formula, data=train_iris, family="binomial")
> summary(glm2)
Call:
  glm(formula = formula, family = "binomial", data = train_iris)
```

```

Deviance Residuals:
    Min      1Q  Median      3Q     Max 
-2.0732 -0.7529 -0.4250  0.9386  2.2185 

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) 5.9490     3.3423   1.780  0.07509 .
Sepal.Length 0.4966     0.8340   0.595  0.55156 
Sepal.Width  -3.2680    1.0456  -3.125  0.00178 ** 
Petal.Length 0.5930     0.7837   0.757  0.44920 
Petal.Width  -1.7861    1.3396  -1.333  0.18241 
---
Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '1
(Dispersion parameter for binomial family taken to be 1)
Null deviance: 113.136  on 89  degrees of freedom
Residual deviance: 87.066  on 85  degrees of freedom
AIC: 97.066

Number of Fisher Scoring iterations: 5
> coef(glm2)
(Intercept) Sepal.Length Sepal.Width Petal.Length Petal.Width
  5.9489966  0.4966006 -3.2679800   0.5930357 -1.7861451
> summary(glm2)$coef
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) 5.9489966  3.3422812  1.7799211 0.075088880
Sepal.Length 0.4966006  0.8340266  0.5954254 0.551559180
Sepal.Width -3.2679800  1.0456208 -3.1253970 0.001775653
Petal.Length 0.5930357  0.7836647  0.7567467 0.449201602
Petal.Width -1.7861451  1.3395827 -1.3333593 0.182413915

```

glm2模型成功训练之后，我们可以使用predict()函数对测试集进行预测。参数type=“response”表示R输出预测因子对应的响应概率。

```

> prob <- predict(glm2, newdata=test_iris, type="response")
> round(prob,3)
 1      5      6      8      9     11     14     15     19     20     21 
 0.077 0.054 0.021 0.105 0.295 0.051 0.223 0.020 0.040 0.027 0.138 
 23     29     31     36     39     41     45     46     49     51     57 
 0.036 0.108 0.230 0.158 0.222 0.059 0.029 0.236 0.048 0.322 0.145 
 58     60     64     65     67     68     69     73     74     75     78 
 0.670 0.382 0.447 0.282 0.253 0.657 0.862 0.757 0.616 0.470 0.355 
 79     81     82     85     94     96     97     99     106    107    109 
 0.364 0.755 0.776 0.234 0.748 0.337 0.371 0.531 0.521 0.461 0.791 
 111    114    116    119    125    127    129    131    133    134    137 
 0.138 0.501 0.092 0.779 0.133 0.380 0.389 0.668 0.347 0.568 0.047 
 141    144    147    149    150
 0.139 0.149 0.618 0.048 0.247

```

6.3 分类树

接下来，我们要学习分类树并考虑如何把这一机器学习工具应用到典型的分类问题中，特别是预测定性变量的问题中。分类树被看做是非线性模型。所有基于树形结构的工具都会生成一个或多个树对象，代表从树根到树冠的一系列分叉。在一个流行的实现方法中（R中的tree包），每个分支的目的都是找到一个预测变量（和那个变量已知的阈值），这将给解释异常值方面带来极大的变化。树形工具用穷举法进行这项工作，举出每个预测因子的所有可能的阈值。一旦创建出一个分叉，这一过程将在每一个新分叉重复进行，直到所有异常值（或者一些低阈值）都能解释清楚，或是在剩下子集中的样本非常少。我们可以以下面这种方式总结树形算法：

1. 在一个大组中考虑所有的预测因子。
2. 找出第一个预测因子，它能最好地将响应拆分成两个不同的同类组。
3. 把数据分割成两组，称作“叶子”。执行分割的位置称为“节点”。
4. 在每一个分叉中，我们重新检索所有的预测因子，包括刚刚拆分的预测因子，尝试找出该组内的另一个预测因子，或者把响应拆分成更多的同类组。
5. 重复这一过程，直至分组足够小或者足够纯净（也就是说，同类的），才能停止算法。

为了理解分类树算法，我们可以使用tree库来分析iris数据集。请记住，R中还有很多其他的树包，例如rpart包和party包，但在本节中，我们将使用tree。它基于我们熟悉的异常统计，所以比较容易使用。在iris数据集中，Species是一个分类响应变量，另一个连续变量是预测因子。第一步，我们需要安装并加载tree库。

```
> install.packages("tree")
> library(tree)
```

让我们进行一些快速的探索性数据分析来熟悉训练集。我们可以看到这个数据集中有90条观测值和5个变量，其中4个变量是预测因子，另一个是响应变量。

```
> str(train_iris)
'data.frame': 90 obs. of 5 variables:
 $ Sepal.Length: num 5.9 4.8 6.7 6 4.7 6.5 6.3 5 4.9 5.8 ...
 $ Sepal.Width : num 3.2 3 3.3 2.7 3.2 3 2.3 3.5 3.1 2.7 ...
 $ Petal.Length: num 4.8 1.4 5.7 5.1 1.3 5.8 4.4 1.6 1.5 5.1 ...
 $ Petal.Width : num 1.8 0.1 2.5 1.6 0.2 2.2 1.3 0.6 0.1 1.9 ...
 $ Species      : Factor w/ 3 levels "setosa","versicolor",...: 2 1 3 2 1 3 2 1 1 3
```

接下来，为了预测Species，我们用训练集train_iris对分类树算法进行训练。注意到，Species指定为响应变量，数据集中剩余的其他变量充当预测因子，在调用tree()函数时，使用Species~来标记。tree()函数的语法跟lm()函数很相似。分类树的返回对象是ct1。

```
> ct1 <- tree(Species~, data=train_iris)
```

决策树在机器学习中如此流行的一个原因是，它们是可视化的，而且容易解释。作为证明，让我们用plot()函数画出树形结构图，用text()函数展示图6-3所示的节点标签。这种类型的可视化经常称为树状图（dendrogram）。沿着树的分支，我们可以看到当Petal.Length < 2.7时，训练集的类型总是setosa。

```
> plot(ct1)
> text(ct1)
```

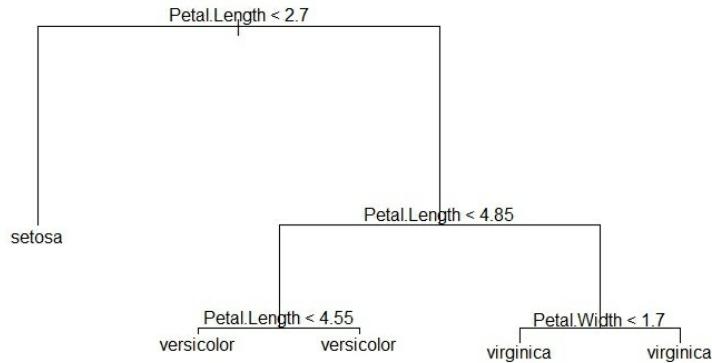


图6-3 用iris数据集预测Species的分类树

如下所示，打印出分类树对象ct1以提供这些信息：每一个分支的阈值、节点样本大小、关联的残差异常，以及预测值。你也能得到样本中每个节点中存在或是不存在的分数（即每个响应类的频率）。例如，在下面的结果中，我们可以看到有一个分叉显示Petal.Length < 2.7，符合这一条件的30条观测的类型都是setosa。

```

> ct1
node), split, n, deviance, yval, (yprob)
* denotes terminal node
1) root 90 197.700 virginica ( 0.33333 0.32222 0.34444 )
2) Petal.Length < 2.7 30 0.000 setosa ( 1.00000 0.00000 0.00000 ) *
  3) Petal.Length > 2.7 60 83.110 virginica ( 0.00000 0.48333
  0.51667 )
  6) Petal.Length < 4.85 28 8.628 versicolor ( 0.00000
  0.96429 0.03571 )
12) Petal.Length < 4.55 21 0.000 versicolor ( 0.00000
  1.00000 0.00000 ) *
  13) Petal.Length > 4.55 7 5.742 versicolor ( 0.00000
  0.85714 0.14286 ) *
  7) Petal.Length > 4.85 32 14.960 virginica ( 0.00000
  0.06250 0.93750 )
14) Petal.Width < 1.7 5 6.730 virginica ( 0.00000 0.400000.60000 ) *
  15) Petal.Width > 1.7 27 0.000 virginica ( 0.00000 0.00000
  1.00000 ) *

```

我们可以使用summary()函数来列出作为树的内部节点存在的变量、终端节点的数目和训练误差率。我们看到，只有两个变量用于预测类型（Petal.Length和Petal.Width），并且这个模型中只有3%的预测值分错了类。

```

> summary(ct1)
Classification tree:
tree(formula = Species~ ., data = train_iris)
Variables actually used in tree construction:
[1] "Petal.Length" "Petal.Width"
Number of terminal nodes: 5
Residual mean deviance: 0.1467 = 12.47 / 85
Misclassification error rate: 0.03333 = 3 / 90

```

现在，我们可以检验训练好的分类树在测试集上表现如何。再一次，我们使用了`prediction()`函数。在下面的代码中，我们用拟合树对象计算一个预测类标签的数组。在这里，`prediction`包含了`test_iris`中响应变量`Species`的预测值。

```

> prediction <- predict(ct1, newdata=test_iris, type='class')
> prediction
[1] setosa      setosa      setosa      setosa      setosa      setosa
[7] setosa      setosa      setosa      setosa      setosa      setosa
[13] setosa     setosa      setosa      setosa      setosa      setosa
[19] setosa     setosa      versicolor versicolor versicolor versicolor
[25] versicolor versicolor versicolor versicolor versicolor virginica
[31] versicolor versicolor virginica versicolor versicolor versicolor
[37] versicolor versicolor versicolor versicolor versicolor virginica
[43] versicolor virginica virginica virginica virginica virginica
[49] virginica  versicolor virginica virginica virginica virginica
[55] virginica  virginica virginica virginica virginica virginica
Levels: setosa versicolor virginica

```

最后，我们使用`table()`来展示一个混淆矩阵，以判断树预测响应的准确程度。我们可以看到，测试集中93%的观测被正确分类。记住，测试集只占`iris`全部150条观测中的60条。

```

> table(prediction, test_iris$Species)
prediction   setosa versicolor virginica
setosa       20      0        0
versicolor    0     19        2
virginica    0      2       17
> (20+19+17)/60
[1] 0.9333333

```

6.4 朴素贝叶斯

接下来，我们将目光转向朴素贝叶斯（Naïve Bayes）分类器，这是一种流行的监督机器学习算法。这种学习方法基于条件概率，换句话说，通过给定已知的其他东西，来推断一件事情发生的可能性。朴素贝叶斯应用了贝叶斯定理（来自贝叶斯统计）和强（朴素）独立性假设。尽管原理很简单，朴素贝叶斯常常表现得比复杂的分类方法更好。

简言之，朴素贝叶斯分类器假定在一个类中，某个特征存在（或是不存在）与其他任何特征存在（或是不存在）没有任何关系。举个例子，如果一只动物有毛皮、四条腿、尾巴，还能吠叫，我们可能认为它是狗。即使这些特征互相依赖或者依靠其他特征而存在，朴素贝叶斯分类器认为所有这些属性独立地促成这只动物是狗的概率。

朴素贝叶斯分类器的一个优势是，它只需要少量的训练数据进行分类所需的参数（变量的均值和方差）估计。因为假定变量是独立的，所以只需要确定每一类变量的方差，而不需要整个协方差矩阵。

朴素贝叶斯的优势是：

- 对不相关特征不敏感；
- 一次扫描就能快速训练；
- 快速分类；
- 能够处理任意数量的预测因子，不论它们是连续的还是分类的；
- 能够很好地处理数据流；
- 尤其适合高维。

朴素贝叶斯的劣势是：

- 假定了特征之间相互独立。

为了阐述朴素贝叶斯，我们将使用e1071包中的naiveBayes()函数。

e1071是机器学习算法的宝藏，在本章的后面，我们还将使用这个包来演示支持向量机。对于这个例子来说，我们将使用iris数据集。

naiveBayes()通过贝叶斯规则，使用独立的预测变量，计算出一个分类变量的条件后验概率。

现在，我们可以从安装并加载e1071包开始演示例子。接着，我们可以将Species作为响应变量，其他变量作为预测因子来调用naiveBayes()。为此，我们将使用iris训练数据集train_iris。一旦创建了朴素贝叶斯对象nb1，我们就可以展示它的内容，以更好地理解算法提供的内容。naiveBayes类中，像nb1这样的对象包括两个组件：apriori包含因变量的分布类，tables是每个预测变量对应的表格的组合。对于连续变量来说，目标类的表格可以提供变量的平均值和标准差。就像下面那样，我们可以很方便地引用对象的组成部分。

```
> install.packages("e1071")
> library(e1071)
> nb1 <- naiveBayes(Species~, data=train_iris)
> nb1
Naive Bayes Classifier for Discrete Predictors
Call:
  naiveBayes.default(x = X, y = Y, laplace = laplace)
A-priori probabilities:
Y
  setosa versicolor virginica
0.3333333 0.3222222 0.3444444

Conditional probabilities:
  Sepal.Length
Y      [,1]      [,2]
setosa 4.983333 0.3141308
versicolor 6.024138 0.4740819
virginica 6.654839 0.6297294

  Sepal.Width
Y      [,1]      [,2]
setosa 3.403333 0.4089375
versicolor 2.779310 0.3244548
virginica 3.009677 0.3279785

  Petal.Length
Y      [,1]      [,2]
setosa 1.480000 0.1349329
versicolor 4.306897 0.3890749
virginica 5.577419 0.5321088

  Petal.Width
Y      [,1]      [,2]
setosa 0.2533333 0.1224276
versicolor 1.3379310 0.1859604
virginica 2.0129032 0.2883770
```

现在，让我们在测试集上使用predict()函数，用刚训练好的朴素贝叶斯模型来做预测。为了检查它的精确度，我们可以使用table()函数展示一个混合矩阵，其中包含响应变量的预测值和实际值。示例的结果表展示了预测结果相当准确，只有3条观测被分错了类。我们可以通过用正确分类的总数（矩阵中对角线的数量和）除以观测的总数，快速地得到一个精确度指标。最后得到95%，这是一个相当不错的成绩。

这个例子展示了朴素贝叶斯分类器在数据集中犯的错误很少。这些数据虽然简单，但是在混合矩阵中可以看到，都是线性不可分的。分类器的所有误分类都是在versicolor和virginica类型中产生的。

```
> prediction <- predict(nb1, test_iris[,-5])
> xtab <- table(prediction, test_iris$Species)
> xtab
prediction   setosa versicolor virginica
  setosa        20         0         0
  versicolor     0        20         2
  virginica      0         1        17
```

为了看到在对象nb1的内部发生了什么，我们使用如下命令来提供数据中的类分布——类的先验概率（“A priori”在拉丁文中的含义是“先验”）：

```
> nb1$apriori
Y
  setosa versicolor virginica
 30      29          31
```

因为这里的预测变量都是连续的，朴素贝叶斯分类器为每个预测变量生成了3个高斯（正态）分布：类变量Species的每个值都有一个分布。如果你使用下面的命令，就可以看到三个类关联高斯分布的平均值（第一列）和标准差（第二列）。

```
> nb1$tables$Petal.Length
      Petal.Length
Y           [,1]      [,2]
  setosa    1.480000 0.1349329
  versicolor 4.306897 0.3890749
  virginica  5.577419 0.5321088
```

通过使用上表中给出的值，我们可以用下面的R命令画出这3个分

布，如图6-4所示。注意，setosa类型（平滑曲线）趋向于拥有更小的花瓣（均值=1.48），且在花瓣长度上的变化率更小（标准差只有0.1349329）。

```
> plot(function(x) dnorm(x, 1.48, 0.1349329), 0, 8, lty=1, main="Petal length dis·
> curve(dnorm(x, 4.306897, 0.3890749), add=TRUE, lty=2)
> curve(dnorm(x, 5.577419, 0.5321088 ), add=TRUE, lty=5)
> legend('topright', legend=c("setosa", "versicolor", "verginica"),
lty=c(1,2,5), bty='o')
```

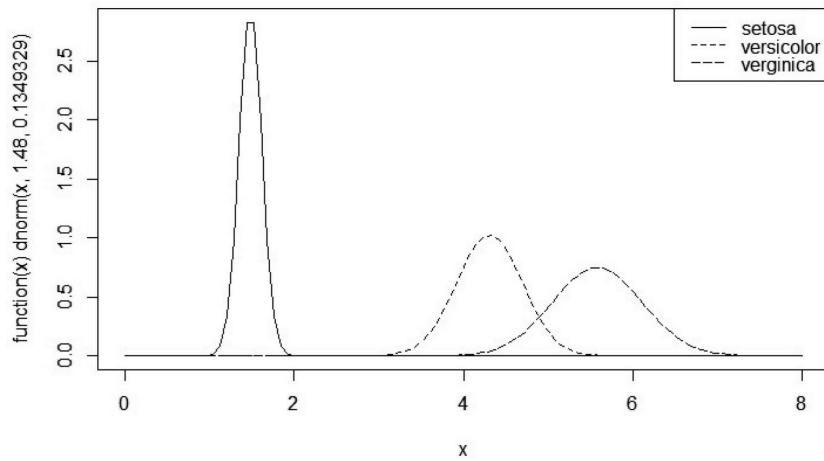


图6-4 基于物种的花瓣长度分布

6.5 K-最近邻

K-最近邻（K-nearest neighbors, KNN）分类器是另一种常用的监督机器学习算法。它可以说是本书中最直截了当的算法。举个简单的例子，根据一个朋友喜爱的食物种类，考虑一个“精神算法”，使用类似的数据向他推荐新的食物。这一直觉基本上就是1-最近邻算法。继续这个场景，K-最近邻算法对直觉进行的一般化，在做出一个推荐之前，你可以使用超过一个数据点。在选择食物的例子中，K-最近邻算法就像是让你朋友做出推荐。我们从询问口味相似的朋友开始，征求他们的建议。如果他们中的大部分人推荐了相同的食物，那么就可以推断，我们也会喜欢这种食物。

KNN算法的目的是将数据点分离成多个类别的数据集，来预测新的样本点的分类。KNN通常用于不知道给定预测因子下响应变量条件分布的情况下，这种情况下，无法使用贝叶斯分类器预测定性响应。很多方法能够尝试估计给定预测因子下响应变量条件分布，然后将给定的观测分到估计概率最高的类别中。KNN就是这样的方法。

为了演示KNN，我们将使用class包中的knn()函数。与基于学习的模型形成对照，KNN也称为基于实例的学习，因为它没有在真正意义上学习任何模型。训练过程本质上来说就是记忆所有训练数据的过程。

KNN的优势主要是简单，因为无需训练模型。不像我们之前遇到过的模型拟合算法，要进行两步的处理，首先拟合模型再使用模型进行预测。KNN使用一次函数调用就能进行预测。当新的数据送达时，KNN就能自动完成增量式学习；同时，旧数据也可以被移除。KNN算法的弱点是无法很好地处理高维数据。

为了可视化展现KNN的工作方式，让我们用iris数据集举一个简单的例子。特别地，我们将使用二维空间的散点图，包含两个预测因子PetalWidth和PetalLength，来观察我们使用的训练集中的观测点。下面的R代码生成了图6-5，展现了响应变量Species的不同值，Species类型用不同的符号进行标注。

```
> plot(train_iris$Petal.Length, train_iris$Petal.Width,  
      pch=c(train_iris$Species))
```

```
> legend('topleft', legend=c("setosa", "versicolor", "virginica"),
  pch=c(1,2,3), bty='o')
```

为了确定最近邻，我们需要一个距离函数来定义从一个数据点到另一个数据点之间的距离。在确定“近”时，数值变量最常用的度量标准是欧几里得（Euclidean）距离。为了预测新的数据点，在图中用符号“?”将其标注出来，我们通过训练集确定K（一个调谐参数，设定为正整数）的最近邻，然后让它们对最后的预测提出建议。在这个例子中，我们使用K=5，所以最近的5条观测为3个versicolor和2个virginica。所以，基于5个最近邻的建议，通过KNN对新数据点进行分类的结果为versicolor。

KNN算法处理了测试集中的每一行数据，找到K个最近的训练集数组，其中包含带有随机打破关系的多数票决定的分类。如果第K个最近向量存在并列，那么所有的观测都要参与投票。投票也可以通过K邻点到新数据点的距离进行加权。

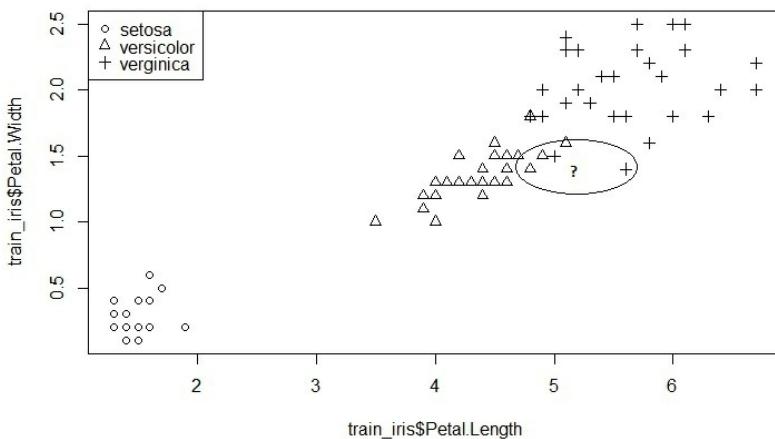


图6-5 K=5时找到最近邻

加一个重要的边注，因为KNN分类器是通过识别离测试数据最近的观测来预测分类的，所以变量的比例尺很重要。变量的值越大，它将对观测距离产生的影响也越大，而变量值越小，影响也会较小。例如，就KNN而言，与数量为2的房间数目差异相比，以\$10 000为单位的房屋价格差异就显得十分巨大了。处理这个问题的一种较好的方法是，对数据

进行标准化，所有的变量都转换成均值为0、方差为1的形式。这样所有的变量就能在一个可比较的尺度上了。可以看3.13小节来复习scale()函数的更多用法细节。

这里有一些需要记住的KNN算法的特征：

- 在二元匹配问题中，选择一个奇数的K值。
- K不能是分类数目的倍数。
- KNN算法的主要缺陷在于为每个样本搜寻最近邻的复杂程度。这意味着该算法需要特定级别的CPU，且容易消耗内存。
- KNN对异常值不灵敏，这使得它能适应分类数据中的任何错误。

现在，让我们看看KNN是如何进行预测的，这次我们使用iris数据集作为例子。首先，安装并加载class库，这个包中有分类算法。接下来，将训练集train_iris分割成预测因子和响应变量，对测试集test_iris也执行这样的操作。knn()函数需要数据框，数据框的第一个参数是训练集的预测因子，第二个参数是测试集的预测因子，第三个参数是训练集的响应值。第四个参数是K，就像我们上面讨论的那样，它控制着KNN算法。最后得到的是一个包含测试集分类结果的因子变量，在这个例子中是prediction。

```
> install.packages("class")
> library(class)
> train_x <- train_iris[,-5] # Training set predictors
> train_y <- train_iris[,5] # Training set response
> test_x <- test_iris[,-5] # Test set predictors
> test_y <- test_iris[,5] # Test set response
> prediction <- knn(train_x, test_x, train_y, k=5)
```

为了测试KNN算法的正确率，我们可以使用table()函数来展示一个混淆矩阵，参数为预测响应值和真实响应值。这个例子生成的结果表展示了预测结果十分准确，只有一条观测错误分类。用所有正确分类的数量除以观测的总数，可以快速得出一个准确率指标。计算得到98.3%，这个结果非常不错。

```
> table(prediction, test_iris$Species)
prediction   setosa versicolor virginica
  setosa        20         0         0
  versicolor     0        21         1
  virginica      0         0        18
```

```
(20+21+18)/nrow(test_iris)
[1] 0.9833333
sum(prediction != test_y) # Number of misclassifications
[1] 1
length(test_y) # Number of observations
[1] 60
```

6.6 支持向量机

我们要讨论的下一个分类算法是支持向量机（SVM，Support Vector Machine）。支持向量机在数据科学社区相当流行，它也在各种各样的问题领域中有很好的表现。真正理解SVM算法需要彻底理解这个统计学习工具的数学原理。不过，这超出了本书的讨论范围，所以我们将用非数学的观点来介绍SVM。

SVM主要是一个分类工具，通过在多维空间构造超平面（hyperplane），对不同的类型标签进行拆分，从而执行分类任务。当你的数据恰好有两种类别时，你可以使用SVM。例如，二元分类问题（虽然对模型进行扩展后可以处理超过二元的问题）。SVM通过寻找最佳的超平面，将一个类别中的所有数据点与其他类型分离出来，由此进行分类。换句话说，通过类别标签对观测进行分离。想法是通过线性边界和非线性类型边界，调节类型的拆分情况。对于SVM来说，最好的超平面意味着两种类别之间存在最大的间隔。间隔指的是与超平面平行，且其中不存在任何内部数据点的切片的最大宽度。支持向量是距离分离超平面最近的数据点，这些点位于切片的边界处。它们支持最优分类超平面，从这种意义上来说，如果这些点略微移动，那么超平面也会随之移动。

图6-6使用一个线性分类器的案例展示了这些定义，有两个特征变量，实心圆代表了归属于一种类型的数据点，空心圆代表了另一种类型的数据点。在这个二维的例子中，分离超平面只是一条直线。在三维中，分离超平面将会是普遍意义上的平面。在这里，数据是线性可分的，因为我们可以看到有很多其他直线可以分离这两种类型。

大多数的分类问题没有这么简单，为了构造最佳分离，需要综合决定边界结构，也就是说，基于现有的观测（训练集），正确地对新对象（测试集）进行分类。基于绘制分割线来区分不同类别的对象的分类方法，叫作超平面分类。如果数据分布基本是非线性的，窍门是把数据转换到更高维的空间，希望数据在高维空间能线性可分。在这种情况下，同一类型的原始数据点使用一套数学函数进行映射（即重新排列），这

些函数称为核函数。重排数据点的过程被称为映射（转换）。注意，在这个新的排列中，映射的数据点是线性可分的，因此无需再构造复杂的曲线，我们要做的就是找到一条可以分离类型的最佳直线。

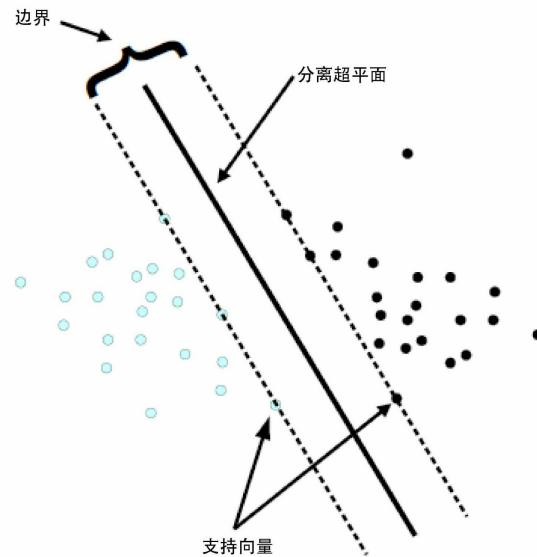


图6-6 二维支持向量机的示意图

为了演示SVM算法，我们将在iris数据集上使用e1071包中的svm()函数。下面的代码使用训练集train_iris训练了一个模型。向svm()传递了一些参数，其中包括Species~.，这意味着我们想要用数据集中所有的预测因子来对响应变量Species进行分类，即Sepal.Length、Sepal.Width、Petal.Length和Petal.Width。我们也可以指定svm()的用法。默认type=“C-classification”是一个分类机，svm()同样也可以作为回归机使用。kernel参数有许多不同的类型，包括线性、多项式、辐射型和S型。对于这个多元分类问题，我们使用kernel=“radial”（缺省值）。你可以使用两个附加的参数调整SVM的操作：gamma和cost。参数gamma供核函数使用，cost能帮助我们规定间隔的违反成本。因此当成本很小时，间隔会变得很宽，支持向量也会很多。你可以多试验一些gamma和cost的典型值，来获得最佳的分类准确率。

```
> library(e1071)
> svm1 <- svm(Species~., data=train_iris, method="C-classification",
  kernel="radial", gamma=0.1, cost=10)
```

下面展示的summary()函数提供了一些关于如何训练模型的有用信

息。我们可以看到，算法发现22个支持向量以如下的形式分布：10个在setosa类中，3个在versicolor类中，还有9个在virginica类中。

```
> summary(svm1)
Call:
  svm(formula = Species ~ ., data = train_iris, type = "C-
classification",
       kernel = "radial", gamma = 0.1, cost = 10)
Parameters:
  SVM-Type: C-classification
  SVM-Kernel: radial
  cost: 10
  gamma: 0.1
  Number of Support Vectors: 22
  ( 10 3 9 )
  Number of Classes: 3
  Levels:
    setosa versicolor virginica
```

在拟合模型svm1\$SV值的帮助下，我们也可以展示算法计算出的支持向量。展示内容包括数据集中观测的索引和支持向量中预测因子的回归系数（位于原比例尺空间）。我在下面只展示了前5个支持向量，而不是全部的22个。

```
> svm1$SV #Show support vectors
   Sepal.Length Sepal.Width Petal.Length Petal.Width
71  0.006532661  0.30579724  0.5624153  0.7747208
84  0.124120557 -0.84094241  0.7315157  0.5125972
88  0.476884245 -1.75833413  0.3369481  0.1194119
86  0.124120557  0.76449310  0.3933149  0.5125972
53  1.182411621  0.07644931  0.6187821  0.3815354
```

也有特殊的plot()函数供SVM使用。想法是对支持向量（用“x”表示）、判定边界和模型的间隔进行可视化。使用下面的R代码，生成如图6-7所示的拟合模型图。这个图能帮助我们想象数据的二维投影（使用Petal.Width和Petal.Length预测因子），并突出分类和支持向量。

Species类以不同的阴影表示。参数slice用于指定一系列维度(dimensionality)不变的命名值（只需在超过两个变量的情况下使用）。

```
> plot(svm1, train_iris, Petal.Width ~
Petal.Length, slice= list(Sepal.Width=3, Sepal.Length=4))
```

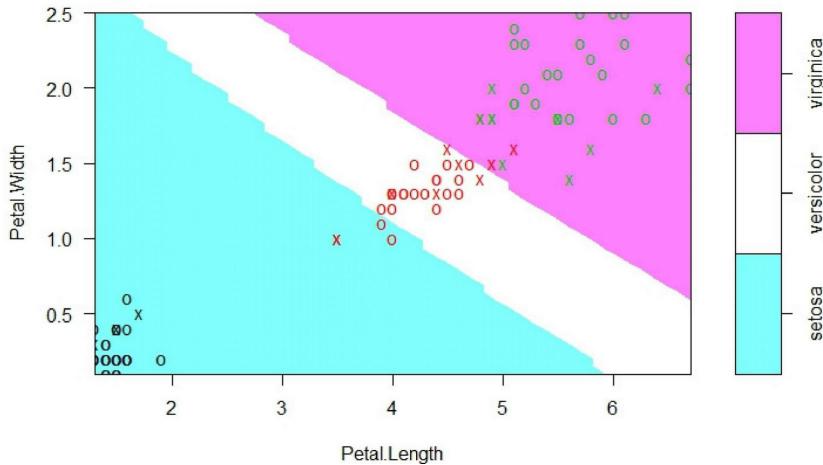


图6-7 SVM图展示了模型的支持向量、判定边界和间隔

接下来，我们要使用训练好的SVM模型和predict()函数对测试集test_iris进行预测。生成的是一个因子变量prediction，包含了对测试集中每条观测做出的预测。我们也可以检查矩阵，看看是否只有一个误分类。

```
> prediction <- predict(svm1, test_iris)
> xtab <- table(test_iris$Species, prediction)
> xtab
      prediction
      setosa versicolor virginica
setosa      20        0        0
versicolor     0       20        1
virginica      0        0       19
```

就像我们在6.5节做的一样，可以用如下R代码检查算法的准确性。第1个数值展示了训练后的算法使用测试集做预测的准确度。98.3%的准确度非常不错。接下来我们可以展示误分类的数目，在这个例子中，测试集中只有一条观测错误分类了。

```
> (20+20+19)/nrow(test_iris) # Prediction accuracy
[1] 0.9833333
> sum(prediction != test_y) # Number of misclassifications
[1] 1
```

6.7 神经网络

在本节中我们将学习神经网络（neural network）机器学习算法（更正式的名字是人工神经网络）。神经网络的概念事实上是一种过时的旧观念，但是现在它被认为是处理很多机器学习难题的最先进技术。但是为什么我们需要另一种算法呢？使用神经网络的动机是由于强有力的非线性模型的需求。例如，你可能要解决一个非线性分类问题，如果你尝试使用逻辑回归，只有在特征变量较少的情况下才能得到解决方案。然而，逻辑回归并不是一个学习复杂非线性模型的好方法。神经网络要好得多，特别是当特征变量的数目较大时。神经网络是计算机视觉流行的机器学习工具，计算机视觉是一个分类算法，用来检查图像并告诉我们其中是否包含特定对象。

神经网络最初的目标是用机器来模仿已知的一流的学习机器：人类的大脑。大脑由一系列紧密结合的神经元组成。虽然特定的某个神经元在结构上可能特别简单，但是互相连通的神经元组成的密集的网络可以完成复杂的学习任务，例如分类。大脑中含有大约 10^{11} 个神经元，平均每个神经元和10 000个其他神经元相连，总计有 10^{15} 个突触连接。神经网络代表了一种尝试，模仿在神经元网络下出现的非线性学习的最基本类型。

这个工具在20世纪80年代和90年代初期广泛使用，但是由于种种原因，在20世纪90年代末期，神经网络的热度减退了。近年来，神经网络又经历了一次重要的复苏。热度减退的一个原因是，神经网络是一个运算量非常大的算法。但最近，计算机已经足够快了，能运行大规模的神经网络。自从2006年开始，更先进的神经网络用来实现深层结构的学习或者更简单的深度学习。已经证明这种算法可以有效地发现数据的底层结构，并已经成功应用到各式各样的问题中，范围从图像分类到自然语言处理和语音识别。

神经网络的动机如图6-8所示。这个人工神经元（artificial neuron）模型用在大多数神经网络算法中。从上游神经元（或者数据集中）收集输入，用组合函数组合在一起，然后输入到通常是非线性的激励函数

中，生成一个输出响应，接着输送到下游的其他神经元中。

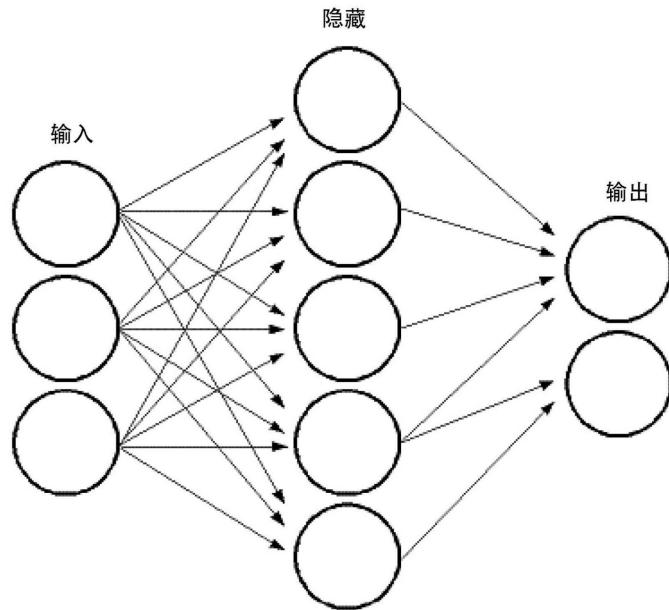


图6-8 神经网络的动机

神经网络在噪声数据方面相当健壮。因为网络包含很多节点或者人工神经元，加上分配到每个连接的权重，网络可以学习解决这些数据集中没有信息甚至是不正确的样本。决策树能为非专业人员提供易于理解的直观法则，而神经网络相对来说难以跟人们解释。此外，神经网络通常需要比决策树更长的训练时间。

神经网络由人造神经元（也被称作节点）网络组成，这些神经元是分层、前馈且完全连接的。网络的前馈（feedforward）特性制约着网络以一个方向流动，不允许循环或者迭代。神经网络包含两层或两层以上，不过典型的配置是3层（如图6-8所示）：一个输入层（input layer）、一个隐藏层（hidden layer）和一个输出层（output layer）。隐藏层的数量可能大于一，但是三层对于大多数场合都足够用了。神经网络完全连接的意思是，某一层中的每一个神经元都和下一层的每一个节点相连，而不是和本层的其他神经元相连。神经元中的每一个连接都有对应的权重。

在神经网络中能够调节的参数包括隐藏层的数量、每一层中神经元的数量和学习率。输入节点的数目通常取决于数据集的大小和数据分布

的类型。隐藏层的数目和每一隐藏层中神经元的数目都是可配置的。输出层中神经元的数目取决于要解决的特定分类任务。

在配置神经网络时，一个很重要的问题是决定隐藏层中需要多少神经元。隐藏层中的神经元越多，识别复杂模式的网络的能力和弹性就越强，因此你可能忍不住在隐藏层中加入大量神经元。然而，这可能并不是一个好的策略，因为一个过于大型的隐藏层会导致过拟合，危害算法处理新数据的能力。如果发现出现了过拟合，可以减少隐藏层中神经元的数目；反过来，如果发现训练准确度低得无法接受，你可以增加隐藏层中神经元的数目。

神经网络用迭代反馈机制进行学习，错误的训练数据输出用来调整相应的输入权重。通过使用一种叫做反向传播的学习算法，这个调整会抛回给上一层。

为了示范如何使用神经网络，我们会再次用到iris数据集来对响应变量Species进行分类。下面的代码安装并加载了neuralnet包，然后用训练集train_iris生成一个副本nn1_iristrain，这样，我们就可以配置算法使用的数据框了。

```
> install.packages("neuralnet")
> library(neuralnet)
> nn1_iristrain <- train_iris
```

从本质上来说，下面的代码创建了新的二元变量，每条观测都有3个值，用于指示Species的值是否等于setosa、versicolor或者virginica其中一个。然后我们给每一个新列都分配了名称，也就是第6列的名称为“setosa”，第7列为“versicolor”，第8列为“virginica”。为了对代码的输出进行可视化，我们使用head()来查看数据框中最后4列。

```
> nn1_iristrain <- cbind(nn1_iristrain, train_iris$Species == "setosa")
> nn1_iristrain <- cbind(nn1_iristrain, train_iris$Species == "versicolor")
> nn1_iristrain <- cbind(nn1_iristrain, train_iris$Species == "virginica")
> names(nn1_iristrain)[6] <- "setosa"
> names(nn1_iristrain)[7] <- "versicolor"
> names(nn1_iristrain)[8] <- "virginica"
> head(nn1_iristrain[,5:8])
   Species setosa versicolor virginica
71  versicolor FALSE      TRUE     FALSE
13    setosa   TRUE     FALSE     FALSE
145   virginica FALSE     FALSE      TRUE
84  versicolor FALSE      TRUE     FALSE
```

```

3      setosa   TRUE     FALSE    FALSE
105  virginica FALSE    FALSE    TRUE

```

接下来，我们要基于重新配置的训练集中的全部4个特征变量对模型进行训练。根据所需的复杂度，我们也有机会定义所需隐藏层和隐藏神经元的数目。添加隐藏层或者隐藏神经元之后，计算函数的复杂度也会随之增长。缺省值是一层隐藏层和一个隐藏神经元。我们使用参数 `hidden=c(4)` 来表示一层隐藏层和四个隐藏神经元。你可以用不同数目的隐藏层和每层上不同的神经元数目来进行试验。举个例子，你可以尝试 `hidden=c(2,4,2)`，这个参数表示三层隐藏层，每层上分别有2、4、2个神经元。当执行算法时，你会发现即使是非常小的数据集，也需要一段时间才能完成。这是因为这个算法需要相当密集的计算。隐藏层越多，花费的时间也越多。在某些情况下，算法可能不会收敛，你必须得回过头修改隐藏层和/或每一层上神经元的数目。

在下面的 `neuralnet()` 调用中，我们对 `algorithm` 参数使用了默认的“`rprop+`”值。这指的是传统的反向传播。取决于该问题域中神经网络所需的预测准确率，还有很多其他算法可供选择尝试。

```

> nn1 <- neuralnet(setosa+versicolor+virginica~
Sepal.Length+ Sepal.Width+Petal.Length+Petal.Width, data=nn1_iristrain, hidden=c(

```

训练好的神经网络对象 `nn1` 是类 `nn` 中的一个对象。你会发现这类对象中包含许多关于训练过程和训练后神经网络的基本信息组件。具体包括重现结果所需要知道的所有信息，例如初始权重。你也可以用 `print(nn1)` 函数发现，训练过程需要 20533 个步骤才能收敛。

```

> print(nn1)
Call: neuralnet(formula = setosa + versicolor + virginica~
Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data = nn1_iristrain,
1 repetition was calculated.
Error Reached Threshold Steps
1 0.9297848198 0.009291227594 20533

```

其他能在神经网络对象 `nn1` 中找到的组件列在下面：

- `nn1$response`——从 `data` 参数中提取，是我们之前设置的 3 个二进制变量。

- nn1\$covariate——从data参数中提取的变量，主要是Sepal.Length、Sepal.Width、Petal.Length和Petal.Width的值。
- nn1\$data——data参数的副本。
- nn1\$net.result——神经网络的输出，即拟合值。
- nn1\$weights——用于复制目的，包含神经网络拟合权重的列表。
- nn1\$generalized.weights——包含神经网络拟合权重的列表。
- nn1\$result.matrix——包含达到阈值的误差的矩阵
- nn1\$startweights——包含初始权重的列表。

Neuralnet包中也有plot()函数能对神经网络进行可视化。图6-9是关于对象nn1的图。它显示了训练好的神经网络的结构，即网络拓扑。默认情况下，这个图包括训练后的突触权重、所有的偏差值以及训练过程中的一些基本信息，例如总误差和达到收敛所需要的步数。

```
> plot(nn1)
```

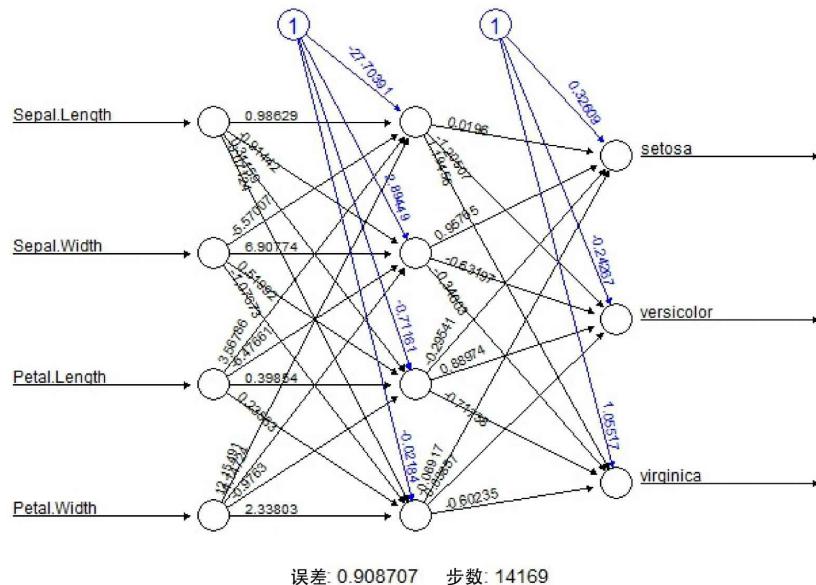


图6-9 一个训练好的隐藏层中含有四个隐藏神经元的神经网络图

接下来，让我们使用训练好的神经网络来预测测试集中的响应变量。我们可以在nn1对象上使用neuralnet()生成的compute()函数。给定一个训练好的神经网络，这个函数能计算出任意预测向量（在这个例子中是test_iris）对应的所有神经元的输出。记住确保训练集数据框中预测因子的顺序和原始神经网络中的顺序相同。

```
> prediction <- compute(nn1, test_iris[-5])
> prediction <- prediction$net.result
```

下一步是建立一个特征向量，其中包含神经网络预测的响应变量值（Species）。然后用table()函数创建一个混合矩阵，通过比较预测响应和实际值，来检查模型的准确性。我们可以直接观察矩阵，看是否只有少量误分类。

```
> pred_idx <- function(x) {return(which(x==max(x)))}
> idx <- apply(prediction, c(1), pred_idx)
> prediction_nn <- c('setosa', 'versicolor', 'virginica')[idx]

> xtab <- table(prediction_nn, test_iris$Species)
> xtab
prediction_nn setosa versicolor virginica
  setosa      20          0          0
  versicolor    0         18          2
  virginica     0          3         17
```

6.8 集成

集成方法（Ensemble method）已经成为数据科学家工具箱中一种重要的补充。这个工具非常流行，以至于成为许多机器学习比赛（例如 Kaggle）参赛者的选择策略。这些比赛中的很多获胜方案都基于集成方法策略的变种。从根本上说，集成方法是能够实现增强预测性能的统计学习算法。通过构造一系列分类器，通常是小型决策树，然后为预测进行加权投票，以便对新的数据点分类。换句话说，集成方法不是挑选一个模型，而是用一定的方式将多个模型组合起来，用以拟合训练数据。其中有两种基本的方法：套袋法（bagging）和提升法（boosting）。

在套袋法算法中，我们使用“bootstrap”工具创建了多份原始训练集的副本，为每个副本拟合一个单独的决策树，然后把所有的树组合起来以便创造一个单独的预测模型。在复合模型训练好之后，使用投票场景来预测未来数据。提升法算法的工作方式也差不多，只不过树是顺序生成的。也就是说，每一棵树生长所需的信息都来自于之前生长好的树。提升法不使用bootstrap采样，而是让每棵树用原始数据集的改进版本来拟合。

为了使集成处理概念化，可以类比为什么一群人总是能做出比一个人更好的决策，特别是当群组成员各自带着自己的偏见的时候。在机器学习中同样如此。集成的一个很好的特性是：你总是能使用更简单的分类器，并依然能获得很棒的性能。从本质上来说，集成是一种监督学习工具，将很多弱分类器（weak learner）组合在一起，力图生成一个强分类器（strong learner）。

体现套袋法性能的一个很好的例子是，向Who want to be millionaire的观众求助。每一个观众都有一些关于正确答案的线索，猜到正确答案的概率会比随机选一个高。即使每个观众都有很大的可能选择了错误答案，但是多数观众选择错误答案的概率还是很小的。这样的类比同样也能指出套袋法的一些局限性。首先，单独的分类器必须要有信息。如果观众中没有人具备这个领域的知识，全靠随机猜测的话，求助观众是没有用的。第二，如果一些预测器已经有自己强有力的预测因子的话，

套袋法即使能提供帮助，也没有多少帮助。如果你之前就知道观众中有一个人很有可能能答对，你宁可直接问这一个人而不是让所有观众投票。套袋法适用于有很多平庸的预测器且很难找到其中例外的预测器的情况。

比起评估单个模型的预测结果，评估集成方法生成的预测结果通常来说需要更多的密集计算。所以从某种意义上来说，集成可以看做是执行大量额外计算来弥补较差的学习算法。优秀的机器学习算法，例如分类树，常常和集成方法一起使用。关于集成方法如何解决数据科学难题的一个很好的例子是我们下一节要讨论的随机森林算法。

就像前面提到的，在机器学习中常用的一种集成方法叫做套袋法（也称作bootstrap aggregating）。在这种方法中，集成的每个模型投票权重都相同。为了提升模型的方差，套袋法用训练集中随机取出的子集来训练集成中的每个模型。这种方法同样也会降低方差并有助于避免过拟合。作为一个例子，随机森林算法将一组随机决策树（即森林）组合在一起，用套袋法来实现非常高的分类准确度。如果你有足够的树，随机树会被当做噪声淘汰，只有优质的树能影响最终的分类结果。

另一方面，提升法是一种用新模型着重训练之前模型中误分类的训练实例，从而增量地创造集成的集成方法。提升法过程从用原始数据拟合一个简单的分类器开始，每条观测的权重都相同。在拟合完成第一个模型之后，计算出分类错误，给数据重新分配权重。之前被错误分类的样本比正确分类的样本有更高的权重。观测值在每一步中不断地重新分配权重，上一个模型错误分类的观测获得了更高的权重。提升法通过增量改善聚合分类器的决策边界来工作，而套袋法是通过降低方差来工作。最后的预测结果将由在迭代过程中生成的每一棵树投票，并由树的精确度计算权重得出。在某些情况下，提升法生成的预测准确率要高于套袋法，但是它同样可能造成对训练数据过拟合的结果。

交叉融合法（blending）是最简单也是最直观的组合模型方法。交叉融合法获取几个复合模型的预测结果，然后把它们加入到一个更大的模型中，例如二层线性或者逻辑回归。交叉融合法可以用在任何类型的复合模型中，但是当复合模型数量更少且更复杂时，它比提升法或套袋法更合适。交叉融合法的性能依赖于模型组合的方式。如果使用一元线性回归，相当于取所有预测的加权平均值，只通过降低方差来工作。如

果用逻辑回归、神经网络甚至是带有交互的线性回归来组合模型，那么组合模型将互相产生乘法效应。

使用集成方法是否合适以及选取哪种工具，都取决于你尝试解决的具体问题。把许多模型扔给问题并把预测结果组合起来很难替代行业知识，但是当行业知识或者显著特征的可用性很弱的时候，这个方法可能会很有帮助。集成方法做成的模型同样也较难解释，但是依然比诸如神经网络这样的黑盒模型容易解释。

熟悉集成方法能力的一个很好的方式是用R中的randomForest()算法来试验分类和回归，这是Leo Breiman和Adele Cutler创建的Fortran中的一部分。在6.9节中，我们将讨论如何用随机森林算法执行分类。

6.9 随机森林

就像在前面一节提到的，随机森林（random forest）算法是最常用、最广泛使用的套袋法工具之一。除了在树的每个决策节点上从训练集中选取训练数据之外，算法还会随机选择一些特征变量，用它们学习得到一棵决策树。最后，森林中的每一棵树投票以选出最热门的类型。

实际上，随机森林是一种递归分区工具，特别适合解决含有少量观测和大量特征变量的问题。它们涉及集成，也就是说，用数据的随机子集计算得出的分类树，在每棵分类树的分叉上使用随机受限的子集和选定的预测因子。在更简单的模型中，一个预测因子的效果通常都会被其他更重要的竞争对手所掩盖，但是随机森林能更好地检查每个预测因子所产生的贡献和行为。此外，已经证明分类树集合的结果比单独一棵分类树的预测结果要好得多。

随机森林真的是一种“随机”的统计方法，每次运行的模型结果都不尽相同。因此，证明森林的稳定性十分重要，可以从至少一个不同的种子值开始，增加森林的大小，直至一个足够大的数目。

例如，使用随机森林来对iris数据集进行分类。首先，我们需要安装randomForest包，然后打开这个库以便使用。

```
> install.packages("randomForest")
> library(randomForest)
```

接下来，我们可以用randomForest()函数调用算法。要用到的参数是formula，描述将要拟合的模型，即Species~..。这说明我们要用数据集中的4个预测因子对Species进行分类。我们也要用data=train_iris指定训练模型用到的数据框。参数ntree=500指定了生成树的数目。为了确保每一行输入至少能被预测几次，这个值不能设置的太小。我们使用mtry=2来指定随机抽样变量数目，这些变量作为每个分支的候选。分类的默认值是变量数目的平方根。最后，我们使用importance=TRUE让算法评估预测因子的重要性。

```
> rf <- randomForest(Species
~., data=train_iris, ntree=500, mtry=2, importance=TRUE)
```

现在，我们可以使用训练好的模型rf对测试集test_iris进行分类。为此，我们将使用predict()函数。然后，用table()生成一个混合矩阵，来对比预测分类和真实分类。我们希望误分类的数目比较少。在这个例子中，只有3个误分类。

```
> prediction <- predict(rf, newdata=test_iris, type="class")
> table(prediction, test_iris$Species)
prediction   setosa versicolor virginica
  setosa       20        0        0
  versicolor     0       20        2
  virginica      0        1       17
```

因为我们之前要求算法评价预测因子的重要性，可以通过使用rf\$importance或importance(rf)函数来获取存放在随机森林对象rf中的这一信息。rf中的importance组件包含了一个矩阵，矩阵的行数等于分类数量加2——在这个例子中是5行。前3行是计算特定类的平均精确度减少。第4行是所有类的平均精确度减少。最后一行是平均基尼系数减少（一种测量模型中各种类型总方差的衡量标准）。

```
> importance(rf)
    setosa    versicolor    virginica  MeanDecreaseAccuracy
Sepal.Length 0.04522895  0.021642010 0.071957664  0.046288212
Sepal.Width  0.00523258 -0.003301299 0.006098693  0.002920681
Petal.Length 0.31637557  0.290288009 0.354026892  0.316221928
Petal.Width  0.30126501  0.249734399 0.286229751  0.275835179
  MeanDecreaseGini
Sepal.Length      6.852732
Sepal.Width       1.407675
Petal.Length     27.888180
Petal.Width      23.093445
```

随机森林对象中的print()函数提供了概括分析。这个总结重申了分析细节并告诉我们所有案例和3种类型的OOB（袋外数据）错误率。

```
print(rf)
Call:
randomForest(formula = Species
~ ., data = train_iris, ntree = 500, mtry = 2, importance = TRUE)
Type of random forest: classification
Number of trees: 500
No. of variables tried at each split: 2

OOB estimate of error rate: 6.67%
Confusion matrix:
  setosa versicolor virginica class.error
setosa      30        0        0  0.00000000
versicolor    0       26        3  0.10344828
virginica     0        3       28  0.09677419
```

`varImpPlot()`函数提供了通过随机森林测量得出的变量重要性的点阵图。这个图能帮助你了解哪些变量是重要的，哪些变量是弱势的。随机森林算法能让你知道每个变量对降低节点杂质的平均贡献程度。一个变量的贡献程度越高，它就越有用。图6-10的变量重要性图展示了准确度和基尼系数的重要性排序（关于分类）是相同的。我们可以看到，`Petal.Length`最重要，`Petal.Width`次重要。

```
> varImpPlot(rf)
```

最后，重要性图由`varUsed()`的计数支持。也就是说，查找每个预测变量在随机森林中实际使用了多少次。这个函数返回整型向量，里面包含了森林中变量使用的频次。在这个例子中，按照重要性排序是`Petal.Length` (528)、`Petal.Width` (414)、`Sepal.Length` (1010)，以及`Sepal.Width` (903)。

```
> varUsed(rf, by.tree=FALSE, count=TRUE)
[1] 528 414 1010 903
```

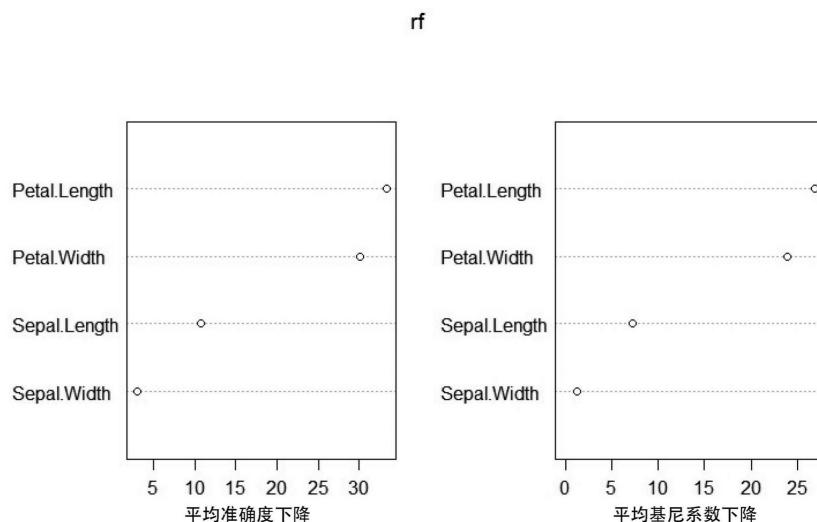


图6-10 随机森林变量重要性点阵图

6.10 梯度提升机

我们讨论的最后一个分类算法是梯度提升机（Gradient Boosting Machine, GBM），它是最流行的提升方法之一。GBM的概念最初是由 Jerome Friedman设计的，R的实现基于这个原始作品。提升是另一种提高决策树预测能力的方式。从技术上来说，GBM可以呈现其他形式，但是决策树是占主导地位的用法，而且R的实现在内部以树的形式来展示。提升特征来源于将多个弱模型在算法上组合起来。GBM被看做是“梯度提升”，因为这个算法能迭代地解决残差。套袋法涉及使用bootstrap工具创建多个原始训练集的副本，用单个决策树拟合每个副本中，然后把所有树组合在一起以便创建一个预测模型。提升的操作跟这个类似，除了树是连续生长的以外——每棵树使用已经生长好的树的信息来生长。显而易见的是，提升不需要bootstrap取样；作为替代，每棵树都用原始数据集的修改版本进行拟合。

GBM有以下特点：

- 可以和随机森林这样的高性能算法竞争。
- 能保持可靠的预测表现，预测结果比简单模型差的情况非常罕见，并且能避免无意义的预测。
- 常常被Kaggle竞赛的获胜者使用，包括Heritage健康奖。
- 能明确地处理缺失数据，例如，NA。
- 无需进行特征缩放。
- 能处理的因子水平比随机森林更高（1024 vs. 32）。
- 没有已知的对特征变量数目的限制。

用于分类的提升方法包括3个调节参数，如下所示，将这些参数传递到gbm()算法中。你可以选择差不多相同的参数值来开始，除非怀疑数据集中有特殊的特征。

- 收缩参数（shrinkage）：一个小的正数，控制提升学习的速率。从另一个角度来说，收缩可以看做是描述算法在减少梯度的过程有多快/强的一个正则化参数。值为0.05就算比较强了，默认值为

0.001。如果值小于0.1，倾向于产生较好的结果。降低收缩系数通常能改善结果，但是需要更多的树。因此shrinkage和n.trees应该协同调整。实际使用过程中，值为0.01或者0.001都是很常见的，哪个值是最佳选择，则取决于需要解决的问题。

- 树的数量（n.trees）：拟合树的总量。选择这个参数时要特别注意，因为这个值如果过大，可能会造成过拟合（与套袋法和随机森林不同）。如果发生过拟合，常常会运行得很慢。如果已经发生或是怀疑发生了过拟合的话，你可以缩减这个值，再进行预测。
- 每棵树的分叉数目（interaction.depth）：这个参数控制着提升集成的复杂程度。更通俗地说，这个参数控制着提升模型的交互顺序。在实践中，取值为1通常就有比较好的效果。

提升模型和随机森林的一个主要不同在于，通常会提升较小的树，因为某棵树的生长要考虑到其他已经生长好的树。使用较小的树对解释能力也有帮助。

为了演示提升，我们将使用gbm包和其中的gbm()算法。我们从安装、加载库开始。

```
> install.packages("gbm")
> library(gbm)
```

现在我们调用gbm()算法，使用剩下的iris数据集变量作为预测因子，来预测变量Species。我们用参数n.trees=2000指定树的数量。虽然gbm()提供了很多分布选择（损失函数），当拥有超过两个分类时，我们将使用“多项式”。基于上面的推荐，参数shrinkage的值设置为0.01。下一步，我们可以打印出模型的结果。

```
> gbm1 <- gbm(Species ~ ., distribution = "multinomial",
+ data = train_iris, n.trees = 2000, shrinkage = 0.01)
> gbm1
gbm(formula = Species ~ ., distribution = "multinomial",
  data = train_iris,
  n.trees = 2000, shrinkage = 0.01)
A gradient boosted model with multinomial loss function.
2000 iterations were performed.
There were four predictors, of which four had non-zero influence.
```

在对象gbm1的帮助下，我们可以用predict.gbm()对test_iris数据集进

行预测。注意，这个函数返回了一个类概率的矩阵，每一行都是一个类型。

```
> prediction <- predict.gbm(gbm1, test_iris, type="response", n.trees=1000)
```

最后，我们为GBM算法使用summary.gbm()函数，生成的图如图6-11所示，展现了每个预测因子的相对影响。在这里，Petal.Length和Petal.Width有最大的影响。

```
> summary.gbm(gbm1)
      var   rel.inf
Petal.Length Petal.Length 69.845854
Petal.Width   Petal.Width  21.582802
Sepal.Length Sepal.Length  4.440778
Sepal.Width   Sepal.Width  4.130565
```

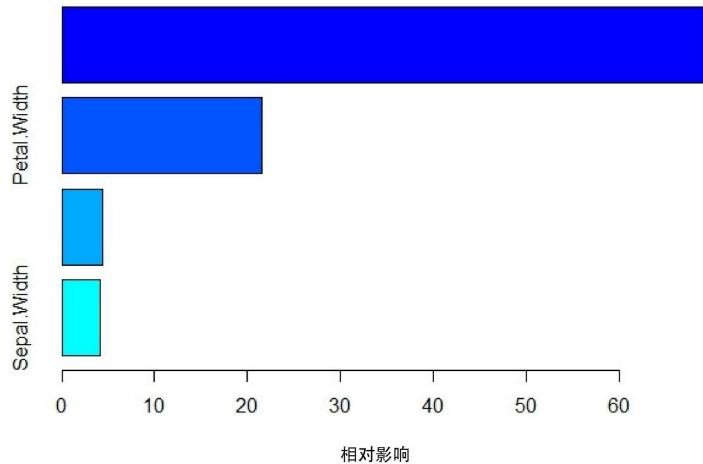


图6-11 GBM summary()的结果展示了相对影响

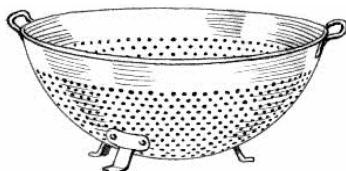
6.11 小结

在本章中，我们继续学习了监督机器学习的几个常用分类工具——识别一条新观测属于哪个类别（亚群体）的问题。分类可能是最容易辨认的统计学习工具，因为它用在非常广泛的问题领域中，例如，垃圾邮件检测、客户流失预测、欺诈检测等等。在本章中，我们考虑了一些流行的分类形式：逻辑回归、分类树、朴素贝叶斯、K-最近邻、支持向量机、神经网络、集成方法、随机森林和梯度提升机。在第7章中，我们将探索用于评估统计学习算法性能的工具，包括评估回归和分类的方法。

下面是本章内容小结：

- 我们从一元分类开始，使用kernlab包中的spam数据集作为案例，在没有建立算法的情况下，介绍了分类的概念。
- 使用广义线性模型的算法——glm()，探索了逻辑回归，对某个响应属于特定类别的概率进行建模。
- 我们使用了tree包中的tree()算法来对分类树进行试验，然后使用训练好的分类器对测试集进行预测。
- 学习了贝叶斯分类器，一种基于条件概率的学习工具。我们用e1071包中的naiveBayes()函数训练模型，然后用它来做预测。
- 我们学习了如何使用K-最近邻算法，一个基于实例学习的算法，使用了class包中的knn()函数。
- 我们提供了对统计学习中支持向量机的一个非数学介绍，然后看了一个使用e1071包中svm()算法的例子。
- 我们探索了使用神经网络来解决非线性分类问题的方法，用neuralnet包中的neuralnet()函数作为案例。
- 最后介绍了两个流行的集成工具，随机森林和梯度提升机。

第7章 评估模型性能



在本章中，我们要探索对机器学习算法交付结果的评估过程，并会提供各种各样的工具以谋求更好的模型性能。当评价一个模型的性能如何时，通常考虑的是拟合程度（ R^2 、调整 R^2 、均方根误差等），但是我们真正想知道的是这个模型在新数据上表现如何。这个概念的基础来自于科学方法，观察产生描述，而实验能产生解释——抱着解释和预测的目标来使用统计学习模型。当预测成功之后，就能产生支撑理论的证据；若预测失败，那么就生成拒绝这个理论的证据。

本章中讨论的方法将用于完成第1章中明确的“数据科学”过程（如图1-2所示）。但是在你投入到项目展示或者“讲数据故事”之前，还得用获得的结果评估模型性能，这时你可能需要重新进行一些数据科学处理步骤：数据处理、特征选择、EDA和统计分析，甚至要重新选择模型（因为你的评估结果可能告诉你，开始选择的算法是错误的）。

事实上，数据科学过程不会停止。你应该不断地用获得的新数据重新评估模型，然后逐步用增强的调节参数来重新训练算法，由此可以获得更好的预测能力。

在模型选择方面，既然已经学习了数据科学过程的其他部分，那么我们现在可以从一个全新的角度审视这个步骤。我们通常有很多备选模型（回归、决策树、神经网络、SVM、随机森林等），为了达到最佳的预测效果，每种模型可能都需要调整。为了选择最佳（最合适）的模型，我们需要客观地评估不同模型的表现。性能评价指标通常是测试误差率，也称作泛化误差（generalization error），指在独立的测试样本中

期望的预测误差。与此相反，训练误差率（training error rate）是训练样本的期望预测误差。这并不是一个客观的评价指标，因为一个模型可能对训练样本过拟合，而在新数据上表现很差。

一旦我们决定了最佳模型，我们想通过用新样本做预测来估算它的测试误差率，这能为评价该模型的性能提供一个客观的指标。这次估计的测试误差率通常比数据科学过程原始模型选择阶段所得到的数据要高，但是这个数据会更贴近真实结果。这是因为在所有备选模型中根据估算测试误差选择最佳模型的行为，等同于在生成测试误差的确认数据阶段冒着过拟合（下一节讨论的主题）的风险训练模型。

在下一小节中，我们将探索各种可以添加到数据科学工具箱中的模型性能工具。在获得更多机器学习经验之后，希望你能获取更多用于处理新情形的工具。最后，你将拥有很多技术来确保算法能在巅峰预测水平下运行。

7.1 过拟合

围绕着机器学习算法性能产生的最常见问题可能就是过拟合（overfitting）了。统计学习的特定应用可能会导致某些模型因为过拟合而表现得很差，在数据科学处理阶段就是要评估是否出现了过拟合。从某种意义上来说，模型过拟合的原因是它拟合了数据集中的一部分噪声，而不是真正的潜在信号。更具体地说，过拟合是算法为了准确度而针对特定的某个训练集训练得太好的情况。如果过于适应一个训练集，那么这个算法就不能很好地推广到其他情形中，对任何新数据它的预测能力都会下降。换句话说，当一个给定的算法产生的训练集误差很小，但是测试集误差很大，那么我们就说这个算法对数据产生了过拟合。

我们可以在两个关键概念下讨论过拟合：样本内误差（in sample error）（你在训练集中得到的误差率，即从用于训练模型的数据集中得到的）和样本外误差（你在新的数据集中得到的误差率，即从测试集中得到的）。在机器学习文献中，样本内误差有时候也称作再代入误差。样本内误差通常要比在新样本中得到的误差乐观一些。原因是你的算法会自我“调整”来适应训练数据集中的噪声。所以当你得到新的数据集时，噪声不同，从而准确度也会有小幅度的降低。相比之下，样本外误差是你从新数据集——测试集——中得到的误差。它有时候也叫做泛化误差。理念是一旦在已知的训练集上创立了一个模型，你就会希望在用其他方式收集（不同时间、不同的观测集）的新数据集上对它进行测试，来观察这个机器学习算法在新数据上表现得如何，即这个算法的推广能力怎么样。

在训练集/测试集里面，有几个需要注意的基本问题。第一，你最关注的是样本外误差。如果你只知道机器学习建立过程中的误差，那么要知道这个值比较乐观，很可能不能反映模型在实际中的表现。第二，样本内误差通常要比样本外误差小。最后，这些特性是由于“过拟合”而产生的，也就是说，你可能把算法和训练数据匹配得太好了，因为训练数据中包含了一些噪声，而不是潜在的趋势。因此，当你把数据用于实践中时，可能并不会符合普遍的现实。有时候，你可能希望放弃一些训练集上的精度，这样你才能在新数据上有更好的推广精度。

这个问题的另一个方面是预测准确率，当一个模型不够复杂，甚至都无法很好的拟合训练集时，我们称这个问题为欠拟合（underfit），算法有高偏差（high bias）（在下一节中，我们会更深入地讨论偏差和方差）。这两种表述都意味着我们没有很好地拟合训练数据。

让我们迅速地看一个偏差的例子，考虑这样一种情况，为了预测基于房屋大小（平方英尺）的住房价格，用一条直线拟合数据点。这里有一个非常强的偏见，或者说是很强的偏差，就是房屋价格和大小之间成线性变化。尽管数据可能跟这个假设相反，但是这个偏差会让算法拟合成一条直线，最终会导致对数据的拟合程度很差。就像在第5章看到的，我们可以用多项式回归来用一条曲线拟合所有的训练样本。这看起来能相当好的拟合训练集（实际上，它用力过猛了），但是这对于房屋价格来说可能不是一个优秀的模型。再说一次，这个问题叫做过拟合，这个算法应该有高方差（high variance）。对于方差的直觉来源于如果我们拟合这样一个高阶的多项式，而这个多项式可以拟合几乎所有的函数，那么留给可能性假设的空间太大、变化太多了。并且我们也没有足够多的数据来驱使它给我们一个较好的假设。在这两种极端的中间某个位置，可以看做是“刚刚好”。

如果特征变量太多，也可能出现过拟合的情况。得到的假设可能能完美拟合训练集，但是不能推广到新的例子中（即，在新的样本中预测价格）。

过拟合也可能影响分类问题，例如使用逻辑回归算法。在这个例子中，当你的决策边界无法很好地分隔类别时，可能出现欠拟合。在另一个极端，你可能发现决策边界为了拟合每一个训练案例而扭曲，这就产生了过拟合。

为了用R演示过拟合的情形，使用kernlab包中的spam数据集。就像我们在第5章看到的那样，这个数据集建立并检验了一个垃圾邮件分类算法。spam数据集包含了一系列变量，描绘了邮件信息中垃圾邮件的特征。这个数据集中包含57个特征变量，加上一个类别标签，其值为“垃圾邮件”或是“非垃圾邮件”。一个有用的特征变量是capitalAve，如果这个值很高的话，表示可能是一封垃圾邮件。我们会用简单的规则定义两个手工算法：一个过拟合，另一个不过拟合。作为开始，我们将加载kernlab包（如果你还没有安装的话，要先安装）。然后选择任意的种子

值来重复结果。在sample()函数的帮助下，我们获得了一个10条观测的小样本，然后将这个子集分配到数据框sampleSpam中。

```
> install.packages("kernlab")
> library(kernlab)
> data(spam) # 4601 observations x 58 variables
> set.seed(333)
> sampleIndex <- sample(dim(spam)[1], size=10)
> sampleSpam <- spam[sampleIndex, ]
```

接下来，执行一些快速的探索性数据分析来熟悉样本数据集。可以生成有用的sampleSpam\$capitalAve图来了解数据的分布。为此，我们需要一种方法标记每个数据点是垃圾邮件或者非垃圾邮件，使用plot()中的pch参数来达成这个目标。我们将“非垃圾邮件”标记为1，“垃圾邮件”标记为2。这些值可以方便地翻译为pch中的标记，圆形或者三角形。

```
> spamSymbol <- (sampleSpam$type=="spam") + 1
> plot(sampleSpam$capitalAve, pch=spamSymbol)
> legend('topright', legend=c("nonspam", "spam"), pch=c(1, 2))
```

从图7-1中可以看到，除了最后一个圈出来的数据点之外，垃圾邮件和非垃圾邮件数据点的分布相当不错。下面是样本中每一个capitalAve的值：

```
> sampleSpam$capitalAve
[1] 1.000 11.320 1.000 1.840 7.300 1.635 2.666 3.545 5.163 2.444
```

现在，让我们考虑用一些简单的规则来建立一个手工算法对数据点进行分类。前两个规则非常普遍，能设法分辨出所有的垃圾邮件，除了圈出来的值为2.444的那个问题样本之外。出现了过拟合的问题就是为了能够完美预测训练集而调整算法。所以，我们加入其他一些规则来把问题数据点考虑在内。

```
> alg1 <- function(x){
  pred <- rep(NA, length(x))
  pred[x>2.7] <- "spam"
  pred[x<2.4] <- "nonspam"
  # Additional rules result in overfitting
  pred[x<=2.45 & x>=2.4] <- "spam"
  pred[x<=2.7 & x>2.45] <- "nonspam"
  return(pred)
}
```

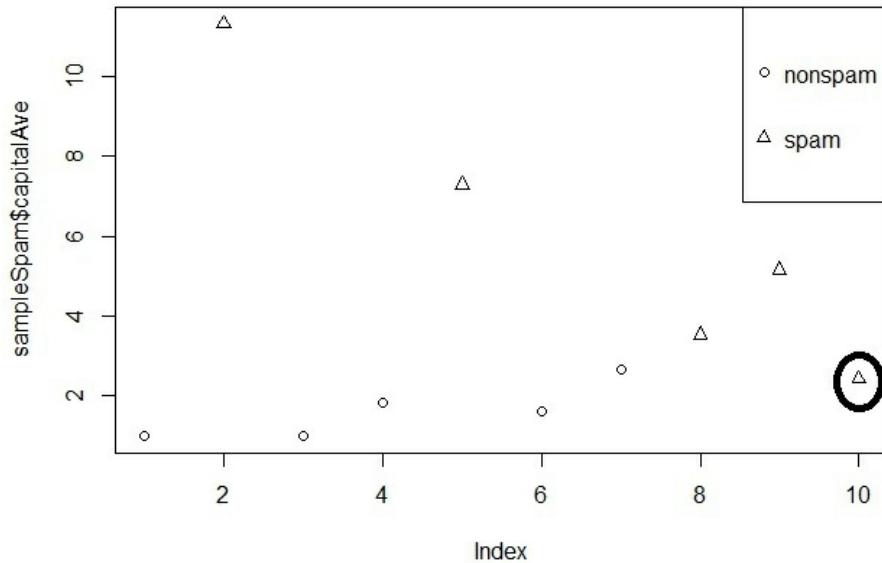


图7-1 训练集中变量capitalAve的探索性数据分析图。
如果我们处理了圈出来的这个数据点，就会出现过拟合。

对比预测标签和真实标签的混合矩阵，5封垃圾邮件和5封非垃圾邮件都被完美地分类了。没有误分类的数据点。

```
> table(alg1(sampleSpam$capitalAve), sampleSpam$type)
      nonspam  spam
nonspam       5    0
spam          0    5
```

现在我们定义第二个算法，这个算法没有过拟合训练集。这一次，混合矩阵展示了有一个预料之中的误分类。

```
> alg2 <- function(x){
  pred <- rep(NA, length(x))
  pred[x>2.8] <- "spam"
  pred[x<=2.8] <- "nonspam"
  return(pred)
}
> table(alg2(sampleSpam$capitalAve), sampleSpam$type)
      nonspam  spam
nonspam       5    1
spam          0    4
```

接下来，我们可以看看两个算法推广到整个垃圾邮件数据集时的表现如何。果然，由于针对小训练集训练得过于完美，`alg1()`犯了很多错误。我们可以轻易地计算出错误的数量（1 235）。现在，在整个垃圾邮件数据集上运行`alg2()`，可以看到它误分类了1 206条数据。这应该不奇怪。比起更普遍的算法，过拟合算法在整个数据集上犯了更多的错误。

```
> table(alg1(spam$capitalAve), spam$type)
   nonspam  spam
nonspam    2141  588
spam       647 1225
> sum(alg1(spam$capitalAve) != spam$type) # Number of errors
[1] 1235
> table(alg2(spam$capitalAve), spam$type)
   nonspam  spam
nonspam    2224  642
spam       564 1171
> sum(alg2(spam$capitalAve) != spam$type) # Number of errors
[1] 1206
```

如果我们认为出现了过拟合的情况，需要如何处理？一些统计问题中有很多特征，这使得难以对数据进行可视化（即绘图）。如果我们拥有很多变量和很少的训练数据，那么过拟合可能成为一个问题。

为了处理过拟合问题，有两种主要的选择。第一种是尝试减少特征的数目。这里，我们可以手动选择留下哪些特征，或者使用一个模型选择算法来自动决定保留哪些特征，剔除哪些特征。减少特征数目的方法能有效减少过拟合。这个方法的缺点是，抛弃一些特征意味着你也抛弃了一些关于问题的信息——有些特征可能会对做预测有帮助。

第二个选择是正则化（regularization）。在这里，我们保留所有的特征，但是减小特征变量值的量级。这个方法在我们有很多特征且每个特征都对预测有贡献时比较有效。在正则化的帮助下，最小二乘估计的估计系数收缩到0附近。这个效果常称为收缩，有降低方差的效果。一些正则化方法可能导致估计的一些系数正好是0，这样能执行间接的特征选择。正则化能通过抑制之前训练数据中的噪音来防止过拟合。

在R中，两种常用的使回归系数收缩到0的正则化工具是岭回归和套索。使用`glmnet()`算法来利用正则化拟合线性模型，可以通过改变`alpha`参数来选择想要的工具：如果`alpha=0`，那么用岭回归模型进行拟合；反之如果`alpha=1`，那么用套索模型进行拟合。`glmnet()`中还有一个叫做

lambda的调整参数，可以控制正则化的影响。你可以在训练集中试验正则化参数lambda的不同取值，通过观察产生的样本误差来看在数据集上的执行效果。在本章后面部分，我们将学习如何计算误差率。注意，lambda值过大、中间取值、过小会产生高偏差（欠拟合）和高方差（过拟合）问题，我们将在下一节中讨论这一点。通常来说，选择一个合适的lambda中间值，能使偏差和方程达到较好的平衡。你可以将lambda值从0开始迭代，间隔为0.02，直到10，看看在每个值下算法的运行效果。使用能将样本外误差降到最小的正则化参数，可以重新进行模型选择。

7.2 偏差和方差

在机器学习中，我们假设问题下面的数据符合某种数学模型。在训练过程中，我们努力将训练数据拟合进这种假设模型中，并确定使误差值降到最低的最优模型参数。如果你运行了一个学习算法，但是它没有像你预期的那样运行得很好（由于较高的交叉验证或是测试误差），几乎所有发生这种情况的原因都是出现了高偏差或者高方差问题——换句话说，也就是欠拟合问题或是过拟合问题。确定是高偏差、高方差、还是二者兼而有之，是很重要的。知道现在发生了什么，能指引你选择哪些工具来提高算法的性能。

其中一个误差原因是，我们假设的模型从根本上就是错误的，例如，尝试使用一个线性模型来对非线性关系进行建模。简单的模型容易欠拟合，因为它们不够灵活，无法模拟真实的关系。这称为高偏差问题，在这类问题中我们使用了过于简化的模型来代表底层数据。另一个误差来源是模型参数太刻意地拟合到训练数据，无法很好地推广到底层数据模式中（即，产生很高的泛化误差）。一般来说，过于复杂的模型会出现高方差问题，这会导致过拟合。当你评估模型的算法时，关键点在于平衡偏差和方差。

这类偏差-方差权衡（bias-variance tradeoff）问题是基于机器学习的数据科学项目的一个重要方面。为了在简化关于这个主题的讨论的同时还能提供有益的思考，让我们考虑一个无需数学方程式（虽然使用方程式非常过瘾）的偏差-方差权衡解释。

为了近似真实，一个学习算法使用的数学模型中的“误差”可以拆分成两个主要成分：可消除误差（irreducible error）和不可消除误差（reducible error）。不可消误差是噪声造成的，无法通过建模消除，它与系统的自然变异有关。在另一方面，可消除误差，顾名思义，可以消除也应该消除到最小值，从而使模型的精度达到最高。

可消除误差可以进一步分解为“由平方偏差导致的误差”（error due to squared bias）和“由方差导致的误差”（error due to variance）。数据科学家的目标是同时获得尽可能小的方差和偏差，以便得到可实现的最精确模型。注意，方差本身就是一个非负数值，平方偏差也是一个非负

数。然而，在选择不同灵活性或者弹性的模型时，需要做一个折中，选取合适的训练集来将这些误差降至最低。

由平方偏差导致的误差是模型对训练数据的预测区别于真实数值或者目标的数量。本质上来看，偏差指的是由一个接近实际生活的问题（可能是非常复杂的）和一个过于简单的模型引进的错误。由于这个特别的定义，我们倾向于认为偏差是在数据科学过程中的模型选择阶段引入的。数据科学家能够重复模型建立过程（通过重采样）来获得平均预测值。如果这些平均预测值跟实际值大不相同，那么偏差就会很高。

由方差导致的误差是用一个训练集得出的预测结果和用所有训练集得出的预测结果之间的区别的总量。在理想情况下，对于不同的训练集，预测值应该相差不大。然而，如果一个模型的方差较高，那么训练数据中的小变化就会引发预测值的大变化。正如偏差一样，你可以多次重复整个模型建立过程。从本质上看，方差测量的是由不同训练集得出的预测的矛盾程度，不考虑预测值是否正确。另一个很重要的点是，高度相关的特征变量会引起共线性（collinearity）问题，反过来会导致方差大大增加。

具有较小方差、较大偏差的模型对真实目标欠拟合。在另一方面，具有较大方差、较小偏差的模型对真实目标过拟合。如上文所述，如果真实目标是高度非线性的，而你选择了一个线性模型来近似，由于线性模型无法捕捉到非线性特征，这时就会引入偏差。换句话说，你的线性模型在训练集上对非线性的目标函数欠拟合。同样地，如果目标是线性的，而你选择了一个非线性模型来近似，那么将会引入偏差，因为非线性模型无法在需要的地方实现线性。事实上，非线性模型在训练集上对线性目标函数过拟合。

偏差和方差的“权衡”可以以这种方式来看待：一个有着较低偏差的学习算法必须是“灵活的”，这样才能较好地拟合数据。但是如果学习算法过于灵活，它将拟合每一个训练数据集，从而导致高方差。一般情况下，我们使用的模型越灵活，方差会增加，偏差会降低。很多监督学习工具的关键特性是，用一种嵌入式的方式来控制偏差-方差的平衡，或是自动控制，或是提供一些特殊的参数以供数据科学家调整。

在致力于偏差-方差权衡时，你需要建立一个度量衡来测定模型的

准确度。机器学习中有两个常用的量：训练错误率和测试错误率。举个例子，对于线性回归模型，你可以计算不同数据集〔用来训练模型的训练集（正如第5章提到的，通常是现有数据的60%甚至70%），和用于检查模型精确度的测试集（通常是现有数据的40%或者30%）〕的均方误差（MSE, mean square error）。出于完备性考虑，在训练之后还需要进行附加的交叉验证步骤。所以通常的分割方法是60%用于训练，20%用于交叉验证，20%用于测试。训练集用于拟合模型。交叉验证集用于估算预测误差，以便选择合适的模型。一旦模型选择完成，测试集用于评估模型（和模型误差）。

在图7-2中，我们看到的是模型的性能曲线。依据横轴的模型复杂度，纵轴展示预测能力。在这里，我们描绘的场景是用若干不同次数的多项式函数（从1次到12次）来近似目标函数。图7-2展示的是计算出了每个估计函数的平方偏差、方差和测试集误差。

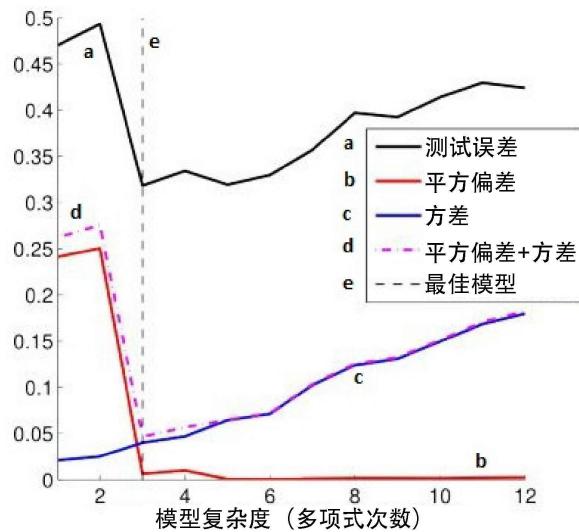


图7-2 展示了测试误差，平方偏差，和方差的模型复杂度图

图7-2展示的是，随着模型复杂度的升高，方差缓慢增加，平方偏差降低。关键在于根据模型复杂度权衡偏差和方差。也就是说，过于复杂的模型通常有较高的方差和较低的偏差，过于简单的模型通常具有较高的偏差和较低的方差。我们的目标就是找到一个同时拥有较低偏差和较低方差的模型。在图7-2中，用虚垂直线标出的是最好的模型，它在数据集上的预测误差最小。我们可以看到最好的估计函数是3次多项式。注意到，虚垂直线也位于平方偏差与方差之和最小的位置。

有经验的数据科学家需要通过精细的模型评估过程来使企业数据价值最大化，而偏差-方差权衡就是一个最好的例子。正如这里所展示的，在使用统计学习解决问题的过程中需要知识、直觉和策略。

7.3 干扰因子

韦氏词典把“混淆”（confound）定义为“无法分辨之间的差异：弄混”。把这个定义翻译到数据科学领域，我们可以发现，干扰因子是一个同时与响应变量和预测变量都有关系的变量。由于这种相互关系，干扰因子会对机器学习问题的精确度有很大的影响。例如，在一个一般的回归问题中，干扰因子可以改变回归线，甚至改变线的形状。

如果两个变量的效果不能分离的话，就说这两个变量是混杂的。在完成一个数据科学项目时，如果除了预测因子之外还有其他的变量能引发研究的内容时，就可能会出现这样的问题。引起干扰的变量会降低内部分析的有效性，因为不能确保完全是预测变量的效果。这个变量和预测因子一起改变，但这不是在预料之中的。因此，最终的结果不能完全归因于预测变量，也可能有其他变量（干扰因子）的贡献。

在某些情况下，干扰因子很难避免。例如，考虑实验者效应。大多数自愿参加用户研究的参与者会希望研究员取得成功。如果知道实验者喜欢哪个条件，他们可能更为乐观地评估它。为了避免产生干扰变量，把可能产生的误差考虑在内是十分重要的，确保每个参与者随机分配到不同的试验环境中，并确认预测变量是唯一一个能够引发这种效果的元素。

让我们考虑一个广告对销量影响的例子。难点在于，有非常多的干扰变量（例如季节或者天气状况）会同时增加广告的曝光度和顾客的购物行为。考虑这样的情况，一个广告经理被问到为什么她的广告是有效的。作为回应，她拿出了一张图表，展示了每年12月，她增加了广告花费。果真，购买量也提升了。当然，在这个例子中，季节可以加到模型里。然而，一般情况下还有其他的干扰变量会同时影响到广告的曝光度和购物倾向，这会使已有关系的因果解释出现问题。

有几种方式可以避免干扰变量对精确度产生的影响。最好的办法是在设计项目时就避免干扰因子。干扰因子有时可以通过对数据集的仔细检查而发现，举例来说，使用探索性数据分析（EDA）工具。也可以采取其他防范措施，这将帮助数据科学家得出正确的结论。第一，可以使用人口统计数据来估计可能的干扰因子。在执行一个消费者调查时，收

集一些额外的人口数据对客观评价可能出现的干扰因子会有帮助。在不同的条件下，这种变量的系统差异会表明这是一个干扰变量。然而，如果实验组没有按这个标准进行区分，分析就会得到加强。例如，在评估一个减肥系统时，研究人员可以收集教育水平和对于健康生活态度的相关信息，然后可以比较这些变量的实验组中是否存在系统差异。

另一种避免基于干扰变量做预测的方法是在研究中加入配套的输出评价。如果使用了这种配套措施，输出的矛盾可能能够指示存在着干扰变量。其他设计和分析可以考虑这些潜在的干扰，甚至可以加以使用。多元统计分析可以用于考虑干扰变量的影响。

考虑统计学习模型中的干扰变量会对结果造成怎么样的影响，这在什么时候都不算晚。如果你觉得算法的结果跟预期的不同，就值得花时间后退一步寻找干扰因子。

7.4 数据泄漏

作为一名数据科学家，你应该时刻注意可能造成机器学习算法过度展现泛化误差的情况，因为这可能使算法无法解决实际问题。当你用于训练机器学习算法的数据中刚好存在待预测的信息时，其中一种潜在的问题叫作数据泄漏（data leakage）。由于不好的泛化来源和对预期表现的过高估计，这在很多层面上都是不可取的。数据泄漏出现得通常很巧妙、很无意，并且它可能造成过拟合。很多数据科学家把数据泄漏看成是首要的机器学习错误之一。

数据泄漏可能有多种表现形式，包括：

- 数据从测试集泄漏到训练集；
- 正确预测或是参考标准泄漏到测试数据中；
- 未来信息泄漏到过去；
- 故意加入的对数据的反向混淆、随机化或者匿名化；
- 在算法的预期用途之外使用数据样本的信息；
- 出现任何以上所述情况的外部数据与训练集结合。

总而言之，机器学习算法的数据泄漏有两大来源：特征变量和训练集。一个数据泄漏的简单案例就是一个模型把响应变量作为预测因子，由此得出结论。举个例子，“晴天的天气是晴朗的”。

再举一个更具体的例子，在一个软件即服务（SaaS, Software as a Service）公司中，使用“客服代表姓名”特征变量来创建一个客户流失算法。在客户流失时，用接待他们的销售代表的名字来做算法，这看起来可能相当正确，直到你发现某一个销售代表被分配去接待那些已经表明可能会流失的用户。在这个案例中，得到的算法能够高度预测顾客是否流失，但是在新顾客身上起不到什么预测作用。这是一个极端的例子，其他更多的数据泄漏案例发生得很巧妙而且难以觉察。现实中，有很多包含数据泄漏的算法在生产系统中已经运行了很多年，才检测到数据创建或者训练脚本中的漏洞。

预先识别数据泄漏并校正，是改进这个机器学习问题定义的重要组

成部分。很多形式的泄漏是很巧妙的，最好的探测方法是提取特征并在该问题上使用最先进的算法训练。这里有几个策略来寻找并模拟数据泄漏：

- 探索性数据分析（EDA）是用于识别数据泄漏的一个有力工具。在探索性数据分析的帮助下，你可以通过统计和可视化工具更加熟悉原始数据。这种类型的检查揭示的数据泄漏模式是令人惊讶的。
- 如果你的算法表现好得难以置信，数据泄漏可能是造成这一现象的原因。你需要事先权衡或者针对手头的问题记录某一性能水平下的结果。用这个预期性能的分歧来检测算法，以使得算法的合理性更接近事实。
- 对算法执行实际测试。任何有意义的数据泄漏都会反映在估计性能和实际性能的差别上。这可能是识别数据泄漏的最好的方式，但这也是最昂贵的手段。分离出数据泄漏造成的性能矛盾通常也是很困难的，因为原因也可能是传统的过拟合、样本偏差等。

一旦识别出数据泄漏，下一步就是考虑如何修复它（或者考虑你想如何尝试修复它）。一些问题与数据泄漏共存，不尝试修复它们也是可以接受的。但是如果你决定要修复泄露，必须要小心不要让情况恶化。通常情况下，有了一个泄漏的特征变量，就会有其他的。移除检测到的明显泄漏可能会使未检测到的泄漏效果更加恶化，尝试修正明显的泄漏，也可能创造出新的泄漏。做法是尝试得到特定观测和/或特征变量的合理性，努力堵住泄漏，并希望能完全封好。校正数据泄漏是研究的活跃领域，在不久的将来，很可能会产生很有价值的结果。

7.5 测定回归性能

在评价一个回归模型的性能时，我们需要一个度量标准来测定预测的精确性。确定拟合效果的最常用标准是均方根误差（RMSE）。

RMSE测量的是估计响应变量和实际响应变量之间的差值。它定义为均方误差（MSE）的平方根。反过来，MSE定义为数据集中每一条观测的估计响应减去实际响应的方差总和，除以观测的总数。因为平方的效果，MSE对于较大的差异更为不利。

对于机器学习算法来说，包括比线性回归复杂得多的算法，RMSE是一种非常流行的测定性能的工具。事实上，像Kaggle举行的著名数据科学比赛，甚至是有着重大影响的Netflix杯，都将RMSE作为评估算法性能的决定性度量标准。

就像第5章提到的，常用的做法是将数据集分割成两个部分。创建一个包含70%观测的训练集，剩下的30%作为测试集。如果数据集中存在某种种类或者顺序的话，不能将数据集拆分成前70%当做训练集，后30%当做测试集。首先应该对数据集进行随机排序或者混排，再填充进训练集和数据集中，这样可能会更好。不过，如果你的数据已经是随机排序了，那么你可以简单地把数据集的前70%划分为训练集，后30%划分为测试集。

下一步是使用训练集来训练算法，然后生成线性模型参数，这些在第5章中有提到。用这些参数，接下来，你可以用MSE或者RMSE计算训练误差率，展现算法预测响应变量的水平。最后，在测试集上使用训练好的算法来得到测试集误差。

为了演示这个模型性能评估的过程，让我们再次使用Prestige数据集。可以从car包中再次调用这个数据集。就像第5章所做的那样，我们会移除残缺数据，然后把数据集分割成训练集和测试集。

```
> library("car")
> data(Prestige) # 102x6
> Prestige_noNA <- na.omit(Prestige)
> n <- nrow(Prestige_noNA) # Number of observations = 102
> ntrain <- round(n*0.7) # 70% for training set
> set.seed(333)          # Set seed for reproducible results
> tindex <- sample(n, ntrain) # Create an index
```

```
> prestige_train <- Prestige_noNA[tindex,] # Create training set  
> prestige_test <- Prestige_noNA[-tindex,] # Create test set
```

为了方便起见，可以定义一个可复用函数来计算RMSE。下面的函数rmse接收了两个参数y_hat和y，它们分别代表预测和实际响应变量。

```
# Calculate RMSE  
rmse <- function(y_hat, y)  
{  
  return(sqrt(mean((y_hat-y)^2)))  
}
```

接下来，用训练模型拟合一个线性模型。模型拟合完成之后，我们就可以用它来计算训练集的RMSE。对于传递给rmse()的y_hat参数，我们将使用在模型lm1下（也可以使用lm1\$fitted）predict()函数的输出。对于参数y，使用训练集的真实响应变量，即prestige_train\$prestige。我们可以得到RSME的值为6.46309。接下来，针对测试集执行类似步骤来计算RMSE。在这里，y_hat的值使用predict(lm1, newdata=prestige_test)，y的值使用prestige_test\$prestige中的真实值。可以看到RSME值为7.705871，这比训练集的RMSE略微大一些。这个结果是预料之中的。

```
> lm1 <- lm(prestige~., data=prestige_train)  
> rmse_train <- rmse(predict(lm1), prestige_train$prestige)  
> rmse_train  
[1] 6.46309  
> rmse_test <- rmse(predict(lm1, newdata=prestige_test),  
prestige_test$prestige)  
> rmse_test  
[1] 7.705871
```

对于偶然的大误差，RMSE比其他测定标准更为敏感，因为平方过程会给较大的误差分配不成比例的高权重。更进一步，RMSE只能在相同单位（例如，美元、公里、红酒的箱数等）的模型之间进行比较。如果一个模型的误差排除了通货膨胀造成的影响而另一个模型的误差没有进行相应的调整，或者一个模型的误差是绝对单位制而另一个模型是相对单位制，那么它们的误差测定不能直接比较。在这些情况下，在计算各种测度前，必须要把两个模型的误差都转换成可比较的单位。也就是说把一个模型的预测结果转换成与另一个模型结果相同的单位。

RMSE值的“好坏”没有绝对的评判标准，它取决于被测量变量的单位和预测准确度，测量的单位跟具体的应用有关。基于所选择的单位，你最好的模型的RMSE值可能非常大或者非常小（也可能处于中间）。“因为RMSE值小于（或者大于）x，所以这个模型很好（或者不好）”，这种表述是毫无意义的。除非在你的预测应用上提及了具体的精确度。

如果一个模型的RMSE值为6.25，不要拿它和RMSE值为6.39的模型作比较，因为它们的差别很小，RMSE值为6.25的模型不会显著强于6.39的模型。用百分比来考虑这个问题会比较有效：如果一个模型的RMSE值比另一个模型低35%，那么通常会有非常显著的区别。如果低12%，区别比较明显。但是如果只好了3%，很可能没有明显的差异。当你权衡模型的复杂度和误差时，这些区别尤为重要：为了降低RMSE值的几个百分点，很可能不值得在回归模型中加入另一个特征变量。RMSE是一个指示相关模型质量的优秀工具，但前提是这个值是可信的。要注意如果存在证据证明模型过拟合，那么RMSE和所有其他的误差测度可能都需要打个折扣。

RMSE令人诟病的一点是，它无法立刻清楚地展示一般情况下模型表现的特征。RMSE值为0表示表现完美，但是机器学习的现实目标不是追求完美。类似地，当一个模型表现很差时，通常也很难通过RMSE值看出来。例如，如果大多数的房屋价格在\$250000附近，你预测的是\$500000，那么你会得到一个非常大的RMSE值。RMSE值没有极限，这样就很难判断模型的表现是否在合理范围内。这个问题的一个解决方法是使用第5章提到的 R^2 来度量。 R^2 展示的是使用模型比仅仅使用平均值好多少。为了便于解释， R^2 通常介于0到1之间。如果你的模型结果和使用平均值一样，那么 R^2 值为0。如果完美预测了每一个数据点，那么 R^2 值为1。

因为 R^2 值总是介于0和1之间，数据科学家通常会将它乘以100，并解释为方差的百分比。即使你缺乏经验，不知道问题领域中的标准RMSE值，也可以方便直观地知道这个模型有多精确。

为了计算 R^2 ，首先要计算使用了平均值作为输出的模型的剩余标准差（RSE）。在这之后，步骤就很简单了，如下面的函数定义所示：

```
> rsquared <- function(y_hat, y){  
  mu <- mean(y)  
  rse <- mean((y_hat - y)^2) / mean((mu - y)^2)  
  rsquared <- (1-rse)*100  
  return(rsquared)  
}
```

在下面的R代码中，我们可以使用rsquared()来计算训练集和测试集的R²值。

```
> y_hat <- lm1$fitted.values # Calculate R2 for training  
> y <- prestige_train$prestige  
> rsquared(y_hat, y)  
[1] 85.09975  
> y_hat <- predict(lm1, newdata=prestige_test) #Calc R2 for test  
> y <- prestige_test$prestige  
> rsquared(y_hat, y)  
[1] 80.71466
```

7.6 测定分类性能

在评估分类模型的性能时，我们需要一个测量标准来衡量预测的精确性。回想一下，在分类中有一个目标类别变量，例如垃圾邮件或是非垃圾邮件。有很多常用的度量衡可以确定一个分类器的精确度，但是通常情况下，我们使用的是误分类率（misclassification rate）。遵循的步骤和回归相同，把数据集拆分成70%的训练集和30%的测试集。然后我们计算测试集误差或者称为误分类错误，也就是说，你要么做出了正确的预测，要么做出了错误的预测（给一条观测贴错标签）。误分类错误是测试集中误分类的观测所计算出的分数比例。这是用于度量误分类错误的测试集误差的定义。

考虑一个二元分类问题，有两级分类（如垃圾邮件或是非垃圾邮件、欺诈或是非欺诈、流失或者未流失、点击了广告或是未点击广告，等等），我们可以把问题分解成以下衡量标准：

- 真阳性 (TP, true positive)：模型预测为真，实际输出也为真。
- 真阴性 (TN, true negative)：模型预测为假，实际输出也为假。
- 假阳性 (FP, false positive)：模型预测为真，但实际输出为假。
- 假阴性 (FN, false negative)：模型预测为假，但实际输出为真。

用上述这些标准，我们可以计算出各种各样的、可以用于阐明模型预测能力的数值。下面定义的分类误差率（误分类率）代表了在一个二元分类问题中，模型对观测分类错误的比例。这是用于衡量所有误分类的指标：即假阴性加上假阳性，除以观测总数。我们的目标是使用这个值最小的模型。

- 分类错误率 $=(FN + FP) / (TP + TN + FP + FN)$ ，也等于 $(1 - \text{准确度})$ 。
- 准确度 $=(TP + TN) / (TP + TN + FP + FN)$ ，是所有预测中正确分类的数目比例。
- 敏感度（又称为召回率或是真阳性率） $= TP / (TP + FN)$ ，衡量正确

识别的阳性比例。

- 特异度（又称为真阴性率）= $TN / (FP + TN)$, 衡量正确识别的阴性比例。
- 精密度（又称为阳性预测值）= $TP / (TP + FP)$, 是真阳性在所有预测为阳性值中的比例。
- 假阳性率 = $FP / (FP + TN)$ 是阴性被错误预测的比例。
- 假阴性率 = $FN / (FN + TP)$ 是阳性被错误预测的比例。

我们可以用一个非常简单的场景来演示用于确定分类算法误分类率的方法。假设我们有两个数组，一个包含了多类别分类器的预测类，另一个包含实际值。换句话说，如果我们的模型使用了有3个可能类别的分类响应变量。我们会使用sample()函数创建一个模拟数据集，第一次调用生成预测值y_hat，第二次调用生成真实值y。然后展示出一个比较预测值和真实值的混淆矩阵（confusion matrix）。正确分类的数值出现在矩阵的对角线中，而错误分类的值出现在除对角线之外的其他位置。通过使用公式 $1 - \text{sum}(\text{diag}(\text{cm})) / \text{sum}(\text{cm})$ 可以很容易地计算出误分类率。就是把对角线上的数值相加，除以矩阵中所有元素的总和，最后用1减去这个值，得到0.58。这个误分类率比较差，但是考虑到我们使用的是一个模拟随机数据集，也就不意外了。

```
> y_hat <- sample(0:2, 50, replace=TRUE)
> y <- sample(0:2, 50, replace=TRUE)
> cm <- table(y_hat, y) # Show confusion matrix
> cm
      y
y_hat  0  1  2
      0  6  4  4
      1  4 11  5
      2  8  4  4
> misclassification_error_rate <- 1 - sum(diag(cm)) / sum(cm)
> misclassification_error_rate
[1] 0.58
```

现在让我们把注意力转向衡量分类算法性能的另一个例子，这次使用真实数据集。机器学习中一个流行的数据集来源是UC Irvine Machine Learning Repository。这个资源中有很多数据集，你可以下载下来用于各种统计学习算法的实验。这次我们将使用Wine Quality数据集，其中包含了11个预测变量和一个响应变量quality。这个响应变量是一个类别变量，数值范围从3到8。这个数据集一个包含1599条观测（葡萄酒）。

下面的R代码使用了随机森林算法在质量方面对葡萄酒进行了分类，我们想通过计算误分类率来衡量模型的准确度。首先，加载randomForest库，再把数据集（CSV文件的格式）下载到工作目录中。然后把CSV文件读入数据框df中。同时，将quality从整型转换为因子。

```
> library(randomForest)
> download.file("http://archive.ics.uci.edu/ml/machine-learning-
databases/wine-quality/winequality-red.csv", "wine.csv")
> df <- read.csv("wine.csv", sep=";", header=TRUE)
> df$quality <- factor(df$quality)
```

现在将数据集拆分成包含70%记录（1119条观测）的训练集和30%（480条记录）的测试集——分别是wine_train和wine_test。

```
> n <- nrow(df)
> ntrain <- round(n*0.7)
> set.seed(333)
> tindex <- sample(n, ntrain)
> wine_train <- df[tindex,] # Create training set
> wine_test <- df[-tindex,] # Create test set
```

最后，使用randomForest()算法和训练集来拟合模型rf。用真实数据集的响应值和拟合模型rf中的预测值，在table()函数的帮助下计算混淆矩阵。最后，就像之前例子中一样，计算得出了误分类率为0.3604167。36%的误差率并不好，你的任务就是尽可能地把它降到最小值。在下一节中，我们会学习交叉验证技术，这些技术可能会帮助我们提高分类器的准确度。

```
> rf <- randomForest(quality~ ., data=wine_train, ntree=20, nodesize=5, mtry=9)
> table(wine_test$quality, predict(rf, wine_test))
   3   4   5   6   7   8
3  0   1   2   2   0   0
4  0   2  11   8   1   0
5  0   2 159  35   7   0
6  0   0  45 114  24   1
7  0   0   5  20  34   1
8  0   0   0   2   4   0
> sum(wine_test$quality!=predict(rf, wine_test)) / nrow(wine_test)
[1] 0.3604167
```

7.7 交叉验证

在本章的前面小节中，我们已经看了测试误差率（样本外误差）和训练误差率（样本内误差）的区别。测试误差是用机器学习算法预测新观测（即没有用于训练模型的观测）的响应的平均误差。如果测试误差很低，那么数据科学家就有理由使用这个特定的算法。如果有可使用的测试集的话，就能很容易地计算出测试误差。反过来，将算法应用在训练过程中使用过的观测中，就能很快计算出训练误差。然而，训练误差率通常跟测试误差率大有不同。更具体地说，训练误差率比测试误差率低得多。我们已经知道，仅仅因为学习算法能很好地拟合训练集，不能推出这是一个好模型，因为可能出现过拟合。这就是不能用训练集误差来预测模型好坏程度的原因，无法推广到非训练集的数据。

交叉验证（cross validation）的过程可以直接用现有的训练数据来估计测试误差率。交叉验证是一类工具的集合，取出拟合过程中的训练数据的子集，将它应用在算法中，从而估计测试误差率。交叉验证涉及反复地从训练集中抽样，然后用每个样本重新拟合一个统计学习模型，以便获得对拟合模型的进一步认识。这一认识无法从使用原始训练集拟合模型时获得。交叉验证也称为是重采样方法。

为了演示这些概念，让我们考虑一个模型选择问题。例如，在多项式回归的情况下，我们尝试选择多项式的次数来拟合数据：一个线性函数（1次）、二次函数（2次）、三次函数（3次）等。基本上，我们要选择多项式方程中所有的系数，再加上一个额外的参数：多项式的次数。更进一步，我们想选择一个模型（即多项式的次数），拟合那个模型，并同时得到一些估计量，用于评估拟合模型推广到新观测的效果。接下来，我们要为拟合到训练集的多项式找出每一项的系数。然后针对每个系数集计算测试误差率。在R中，这无非就是意味着在每个具有不同公式的模型中使用测试集，然后计算出对应的测试误差率。这能够评估每个模型在测试集上的表现。我们希望寻找最低的测试误差率，目的是选择“最好”的模型，即，这个模型在测试集上的表现可能是最好的。遗憾的是，这个数值并不能合理地判断泛化能力，因为它是对泛化误差的乐观估计。因为我们用测试集来拟合多项式的次数参数，它在这

个测试集上的表现很可能会比新数据集上的表现更好。

为了处理这个问题，让我们回到模型选择阶段。我们把数据集分割成三个部分，而不是训练集和测试集两个部分。按照惯例，第一个部分是训练集。第二部分叫做交叉验证集，最后一个部分叫做测试集。一般的分割比例是60%用于训练，20%用于交叉验证，20%用于测试（然而有些数据科学家使用50%，25%，25%来分割）。每一部分数据集都有它们自己的关联误差：训练误差率、交叉验证误差率和测试误差率。

现在让我们回到前面描述的问题：在模型选择中使用测试误差率不能合理地估计获胜模型推广到新数据的效果。这一次，我们将使用交叉验证集来选择模型。在R中，处理过程如下：

- 基于每个提出的多项式函数，拟合一个线性模型。如果我们的多项式从1次到5次，那就基于训练集分别拟合出5个线性模型。
- 用交叉验证集分别测试5个模型，查看每个模型的效果。然后计算出每个模型的交叉验证误差率。
- 选择交叉验证误差率最小的模型。假定你选择的三次多项式（次数为3）交叉验证误差率最低。这里，我们把三次多项式用交叉验证集拟合了。这意味着这个多项式次数不再拟合到测试集中。
- 现在我们可以使用剩下的测试集来估计模型的泛化误差。

在现在考虑交叉验证的过程中，重新回到偏差-误差权衡问题，我们可以观察到高偏差（欠拟合）出现在训练误差率和交叉验证误差率都很高的情况下，即交叉验证误差率可能接近甚至比训练误差率还要高一点。反过来，高偏差（过拟合）出现在训练误差率较低的情况下，即很好地拟合了训练数据集，但是交叉验证误差会比训练误差大得多。

继续前面例子中列出的主题，当我们尝试从许多不同的模型中选择在训练集中表现得最好的模型时，训练集的精确度（也称为再代入准确度）通常都会比较乐观。选择出的模型会适应训练集中的一些异常值，无法准确地代表在新样本中的预测精确度。

一个更好的估计指标来自于一个独立的数据集——也就是测试集精确度。但是有个问题，如果我们反复地用测试集来评估模型的样本外精确度，那么从某种意义上来说，测试集已经成为训练集的一部分，我们

依然没有外部的评估标准，即对测试集误差的独立评估。

为了估计测试集的精确度，我们需要用到和训练集有关的东西来较好地评估测试集精确度，这样，我们就能完全使用训练集来建立模型，同时只用测试集评估一次。数据科学家完成这一精度测量的方法是使用交叉验证。概念性的方法列在下面：

- 获取训练集，并将其分割成两个部分，可以看做是一个较小的训练集和一个测试集（交叉验证集）。
- 基于训练集（原始训练集的子集）建立一个模型。
- 基于测试集（原始训练集的子集，也称作交叉训练集）评估模型。
- 将这一过程重复多次，算出估计误差的平均值。

这一过程可以用来选择模型中要包含的特征变量。我们可以选定特征变量的数目，用这些特征变量组来拟合出若干模型，然后使用在交叉验证测试集中拟合得最好的模型。我们也可以使用特定的算法类型，然后选出在交叉验证测试集中表现得最好的一个。也可以针对预测模型调整参数。在所有这些过程中，我们完全没有用到原始的测试集。所以，当最后把测试集应用在预测算法中时，它能公正地评判样本外精确度。

多种不同的重采样方法可以用于创建交叉验证集。总的来说，这些估计模型性能的工具的运行机制很相似：样本的一个子集用于拟合模型，剩余部分用于评估模型的有效性。这个过程重复多次，最后能得到汇总、总结后的结果。下面列出的方法之间的区别集中在选择哪种二次抽样法。

- 随机二次抽样。用这种方法，对原始训练集进行随机抽样（random sampling），把它拆分成两个部分：一个训练集和一个交叉验证集（或者叫留出集）。新的训练集会比交叉验证集大一些。在完成分割之后，你可以用新的训练集来建立一个模型。然后用这个模型来预测交叉验证集，并评估准确性。得到的交叉验证集误差率（对于定量响应，对应的是均方根误差）提供了对测试误差率的估计。你可以用多个不同的随机二次抽样集来重复这一过程，然后算出误差的平均值。
- **K**-折交叉验证。我们把原始的训练集随机折成k个同样大小的组，

用于交叉验证。通常k选择5或是10，但是没有硬性规定。举个例子，假设k选定为3。在这里，你就要把原始的训练集折成3块相同大小的部分。第一块当成是交叉验证集，模型拟合到剩下的k-1块数据中。交叉验证误差率通过留出块来计算。这个过程重复k次。在每次迭代中，选择不同的块当做交叉验证集。这个过程最后能生成k个对测试集的估计。通过计算这些值的平均值，得到k-折交叉验证估计。对于k-折交叉验证来说，k的值越大，得到的偏差越小、方差越大。而k值越小，得到的偏差越大、方差越小。在这里，偏差指的是估计性能和真实性能之间的差值。如果你设定了一个比较大的k值，例如20，那么你就能非常精确地估计预测值和实际值之间的偏差，但是这也会带来很高的方差，即它会很大程度上取决于你选取的特定子集。对于小一点的k来说，假设是一个2-折交叉验证。你可能无需得到对样本外误差率的精确估计，因为你只留出了一个样本，用剩下的大部分数据来训练模型。但是方差也会比较小，因为可供选择的子集也比较少。

- 留一交叉验证（**LOOCV, Leave one out cross validation**）。类似于随机二次抽样法，LOOCV需要将观测集拆成两个部分。我们留出一个样本，然后基于剩下的所有样本建立一个模型，然后预测留出的样本值。迭代原始训练集中的所有样本，每次都留出一条观测。因为留出的观测没有用于拟合，精确度指标（MSE）能提供对样本外误差率的大致无偏估计。LOOCV的估计是单个测试误差估计的平均值。我们可以看到，LOOCV是k-折交叉验证的特殊情况，即k等于观测数目。LOOCV的偏差比随机二次抽样要小得多。在使用LOOCV时，我们反复地使用和整个数据集差不多同样大的训练集（每次留出一条观测数据）来拟合模型。与此相反，随机二次抽

样使用的训练集通常是原始数据集大小的一半。这意味着LOOCV方法不会像随机二次抽样那样过高估计测试误差率。

请注意，用随机取样做交叉验证必须是不放回的，因为我们在对数据集进行二次抽样。同样的，如果你用交叉验证来选择预测因子，必须在独立数据集中评估误差，这样才能得到真实的样本外误差值。从本质上讲，这意味着如果你使用交叉验证来选择模型，交叉验证误差率可能无法很好地表现真实的样本外误差率，因为你总是选择最好的模型。再强调一次，实现良好性能的最佳方式是将独立的测试集只应用到模型上

一次。

让我们看一个使用交叉验证法的综合例子，它通过比较估计的测试误差率来执行模型选择。为了让交叉验证过程比较简便，我们使用R的ipred包中的errorest()函数。这个函数用10-折交叉验证来估计给定模型的测试误差。在下面的R代码中，我们要在一系列相似的算法中使用errorest()。从安装并加载ipred（改进预测因子）包开始，并设定重复结果的种子值。

```
> install.packages("ipred")
> library(ipred)
> set.seed(314)
```

对于测试场景，我们将使用iris数据集，并用errorest()函数处理随机森林、朴素贝叶斯、K-最近邻、支持向量机和线性判别分析。errorest()函数需要一个公式、一个数据集和一个模型。它也需要一个可以返回预测类别的预测函数。errorest()的返回值是交叉验证后单独的一个估计测试误差。

```
> # Random forest algorithm
> library(randomForest)
> cv_error <- errorest(Species~., data=iris, model=randomForest)
> cv_error$error # class=cvclass
[1] 0.04666667
> # Naive Bayes algorithm
> library(e1071)
> predict_nb <- function(object, newdata) {
  predict(object, newdata[,-1])
}
> cv_error <- errorest(Species~., data=iris, model=naiveBayes, predict=predict_nb)
> cv_error$error
[1] 0.04666667
> # K-nearest neighbor
> library(class)
> predict_knn <- function(object, newdata){
  predict.ipredknn(object, newdata, type="class")
}
> cv_error <- errorest(Species~., data=iris, model=ipredknn, predict=predict_knn) # default k=5
> cv_error$error
[1] 0.03333333
> # Support vector machines
> library(e1071)
> cv_error <- errorest(Species~., data=iris, model=svm)
> cv_error$error
[1] 0.03333333
> # Linear Discriminant Analysis
> library(MASS)
> predict_lda <- function(object, newdata){
```

```

predict(object, newdata)$class
}
> cv_error <- errorest(Species~., data=iris, model=lda, predict=predict_lda)
> cv_error$error
[1] 0.02

```

在运行完所有的备选算法之后，我们可以检查交叉验证误差，然后选择最佳的一个。表7-1是实验的结果汇总：

表7-1 对一系列常见算法的CV误差分析

算法	10-折CV误差
随机森林	0.04666667
朴素贝叶斯	0.04666667
K-最近邻	0.03333333
支持向量机	0.03333333
线性判别分析	0.02

通过结果可以看到，当我们寻找最低的CV误差值时，很多模型在预测性能方面表现得都差不多。线性判别分析（LDA）看起来是个不错的选择。假设选择了LDA作为胜出模型，我们可以重复10-折交叉验证25次来得到交叉验证误差的标准差。

```

> set.seed(314)
> cv_result <- replicate(25, errorest(Species
~., data=iris, model=lda, predict=predict_lda)$error)
> cv_result # Numeric vector, length=25
> sd(cv_result)
[1] 0

```

得到的差值为0，这说明交叉验证误差应该很接真实测试误差。很多总结工具和图表工具可以用于评价选定的这个模型。对于分类响

应，我们可以用交叉验证过程中样本外预测的混合矩阵来展现预测性能。

```
> pred_species <- errorest(Species  
~., data=iris, model=lda, predict=predict_lda, est para=control.errorest(predicti  
> table(iris$Species, pred_species) # confusion matrix  
pred_species  
setosa versicolor virginica  
setosa      50        0        0  
versicolor    0       48        2  
virginica     0        1       49
```

7.8 其他机器学习诊断法

在本章中，在评价模型性能时，我们要讨论的最后一个主题是，当你试图把算法推广到训练集中不存在的新数据时，预测结果中产生了无法接受的很大的误差的情况下，诊断过程要遵循的一些基本原则。诊断法是一种检验方法，可以用来深刻理解什么对机器学习算法有效，而什么无效。并且能获取提高性能的指引。诊断方法需要花费时间来实施，但是这么做不会无功而返。

遗憾的是，很多时候策略选择倾向于传统的“直觉”，随机地选择下面的一个准则。这会使数据科学家通向许多没有成果的道路，最后完全就是浪费时间。一个更加结构化、有组织、考虑周到的方法，通常能带来更高的效率。

7.8.1 获取更多的训练观测数据

这条准则可能是最简单的：你怀疑算法的预测能力，试着获取更多的数据。数据科学界的一句箴言说，“更多的数据能打败一个聪明的算法”。虽然没有严格的规律，但是你总是能通过扩展操作的数据集来提升算法的精确度。举个例子，你可以想象通过执行额外的调查来获取更多的数据。

实际上，数据科学家可能花费大量时间获取10倍的训练数据，但是事实证明增加的数据实际上并没有用。这意味着我们无法通过这个数据收集工具来盲目地扩展预测能力。收集更多数据会导致在这个项目上花费的时间和金钱也增加，在做很多努力之前，我们需要仔细地评估现状。注重实效的途径是，画出不同数据集大小下的误差图。常用的做法是，对不同模型的训练集进行不同大小的采样，然后同时画出基于训练集大小的交叉验证误差和训练误差率。凭借这个图，我们可以更好地确定是否数据越多，误差率越低。

7.8.2 特征降维

当特征变量的数目较大时，你可能希望重新进行特征工程的过程（数据科学过程的早期阶段），来减少模型中用于预测的变量数目。花一些时间找到特征变量的较小子集则有助于避免过拟合。

7.8.3 添加新特征

虽然不能立刻直观地显示，添加一些之前没有考虑（或者之前不存在）的特征变量可以增强模型的预测性能。可能现有的特征集无法提供足够多的有用信息，所以你收集了不同的数据——可能是通过更为直接的调查——来使预测过程更清晰。遗憾的是，这可能是个大工程，没有具体方法可以预先确定花费的这些时间是否真的有价值。

7.8.4 添加多项式特征

你可能试着增加用于回归问题的多项式的次数，例如，为某个特征加入更多的n次项，或者为一组可能增强模型性能的特征增加乘积项。再次提醒，花时间在添加多项式次数上可能成功，也可能不成功。

7.8.5 对正则化参数进行微调

你也可以试着增加或减少正则化参数lambda，看会对算法性能造成什么影响。这个策略相当简单，因为它只涉及在R中调用函数时改变一个参数。

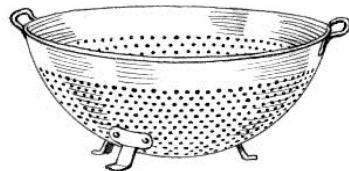
7.9 小结

监督机器学习用于预测我们在第5章学习的连续响应变量和第6章的分类响应变量。在本章中，我们试着测量这些统计学习模型的精确度，并试用了多种可用于提升精确度的工具。

下面是本章内容小结：

- 我们强调了机器学习中最常见的问题，包括过拟合，这种情况是算法针对特定的训练数据集训练得太好而导致的。
- 我们也复习了偏差和方差的概念，详细地了解如何权衡偏差-方差并平衡这两者。
- 我们看到了干扰因子的棘手影响，干扰因子就是同时与响应变量和预测变量有关的变量。
- 我们学习了数据泄漏，这是一个难以检测的问题，用于训练机器学习算法的数据中碰巧含有你试图预测的信息。
- 我们定义了确定回归模型精确度指标的计算方法。
- 我们定义了确定分类模型精确度指标的计算方法。
- 我们确定了交叉验证步骤如何用于减少统计学习模型的泛化误差率。
- 在本章的最后，我们提供了在预测性能较低时使用的一些附加工具。

第8章 非监督学习



在前面的各章中，我们已经看了许多监督学习下面的机器学习技术，例如回归和分类。这些工具用特征变量测定已经标记好的训练数据集，然后用于预测新的测试数据集。在本章中，我们将看到一种全新的机器学习：非监督学习。虽然它没有监督学习那么常见，但是这种新方法对于从未标记（unlabeled）数据集中发现之前不知道的信息有很大的潜能。这意味着非监督学习用到的数据集中不包含响应变量，因为我们没有试图去预测任何东西。非监督学习的主要用途是发现数据中的位置模式，举例来说，把相似的数据组合在一起，或是检测异常值。识别集群是非监督学习的一种典型应用场景。

非监督指的是试着了解数据底层的结构，而不是试图优化得到一个具体的、预先标记好的标准（例如创造一个预测模型）。非监督学习是探索性分析（第4章提到的内容）的很好的工具，但是它更加主观，因为没有预测响应变量这样的具体目标。同时，也很难评价非监督学习方法得出的结果，因为没有普适的处理程序来评估模型性能或者在独立数据集上验证结果。监督学习的大前提是，你可以通过模型，用测试数据集预测响应变量的效果来验证工作，这些测试数据集之前没有用于训练模型。但是对于非监督学习来说，因为没有正确的答案，也就没有方法可以验证工作。这就是为什么叫做非监督学习。

由于大数据在体量、种类、速度上的边界在不断扩张，非监督学习工具在很多应用领域中也越来越重要。例如，一个电子商务网站可以识别具有相同浏览和购物历史的顾客群体，也能为每一购物群体推荐他们感兴趣的产品。总之，任何时候有效地通过相似的人口统计特征、行为

和偏好把个体分组集合在一起，你就可以成功地提供定向市场营销，从而提高销量。

8.1 聚类

非监督学习的主要方法叫做聚类（Clustering），这是一套用处很广泛的工具，可以用于识别数据集中的数值群组或是聚类。聚类把这些“接近”的东西组织起来，然后把它们放入群组中。在使用聚类时，有很多问题需要考虑：

- 当我们谈论数据测量时，如何定义“接近”？
- 定义了“接近”之后，怎样把这些东西分组？
- 如何使分组可视化？
- 如何解释通过这个统计过程创建的分组，该分组可能令人难以置信，或者由于异常值而创建。

让我们从如何定义接近开始，这在聚类算法中是最重要的一步。距离测度必须适合你拥有的数据类型。如果不适合，那么你会得到一些不易解释的集群，也就是说，难以清楚地展现模式。对于连续变量来说，你可以使用欧几里得距离，或者不考虑最小距离，你可以着眼于相似性或是距离的二进制测量，例如曼哈顿距离。要使你分析的问题得到较好的聚类结果，选择哪种距离或是相似性的测量标准尤为重要。

首先，考虑欧几里得距离。可以结合图8-1来看这个度量标准，图中展示了纽约和波士顿之间的距离。像图中标注的那样， y 轴表示了两个城市的纬度值， x 轴表示的是经度值。更进一步， $Y_1 - Y_2$ 是两地纬度的差值， $X_1 - X_2$ 是经度的差值。我们希望定义的是这两个差值的某种组合，以用于测量纽约和波士顿之间的距离。如果你能回想起基本的几何学知识，这其实就是勾股定理（Pythagorean Theorem）的公式。当然，这个距离测量是针对二维的。我们可以将这个公式推广到多维来表示特征变量，用以解决聚类问题。在图8-1中展示的推广方程中，我们使用任意字母A~Z来表示这些变量。在执行聚类时，这个距离测量标准用在定量（连续）变量上。但是在超过三维的聚类可视化问题中，没有办法展现这样的结构。在这里，降维就派上用场了，你可以用较少的特征变量来创建一个新的数据集，大致和原始数据集等价。本章的后面部分会讨论这个工具。

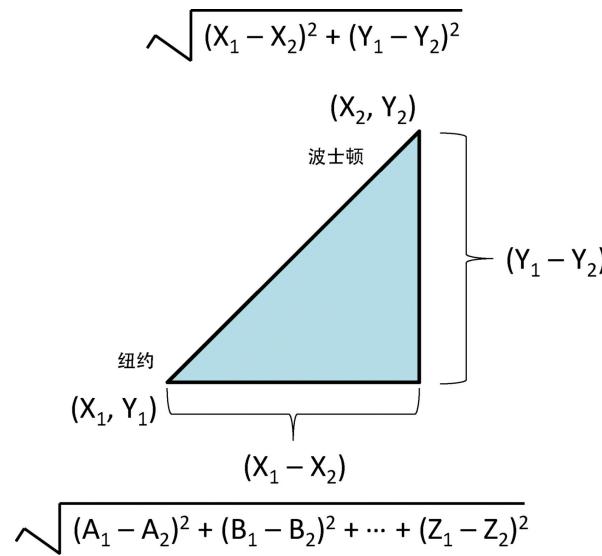


图8-1 计算欧几里得距离的例子

另一个用于计算距离的度量标准是曼哈顿距离（Manhattan distance），常常用在二元变量中。如图8-2所示，考虑图中标出的点A和B，把它们看做是城市中的建筑物。按照穿越的街区数目计算，这三组直线段的距离都相等。曼哈顿距离的公式展示在图8-2中。它看起来跟欧几里得距离很类似，但是无需对坐标差值求和再开方，你只要计算坐标差值的绝对值之和即可。

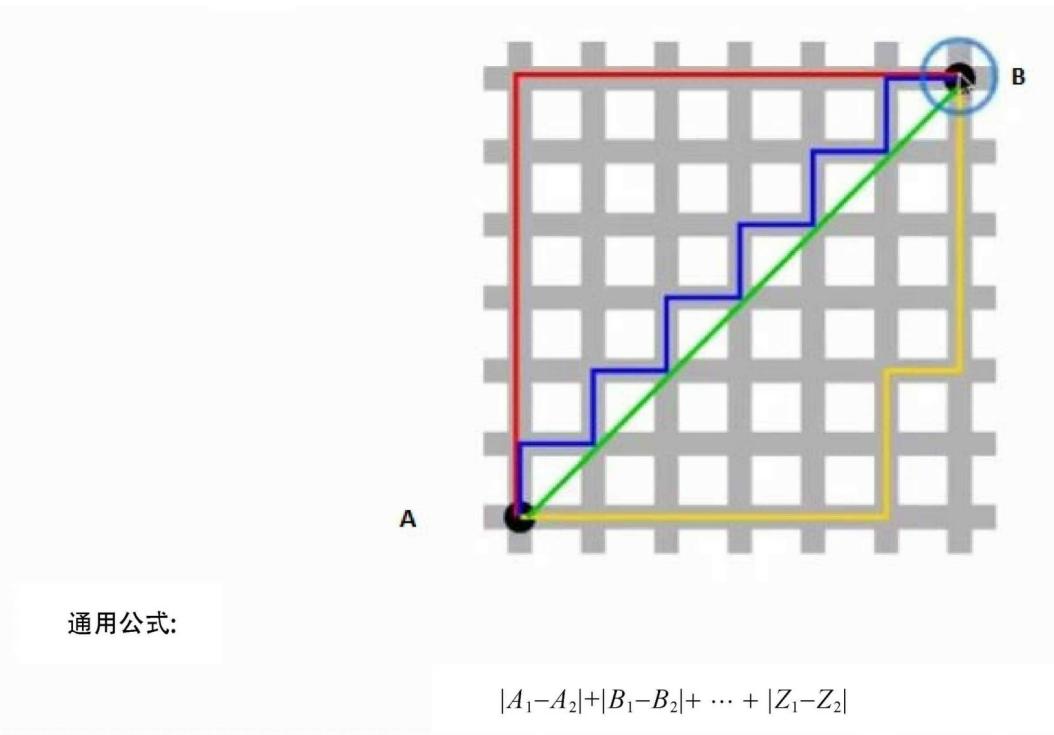


图8-2 计算曼哈顿距离的例子

8.2 模拟聚类

为了展示数据如何分成聚类，我们从一个使用小型仿真数据集的例子开始。R能很好地帮助我们形成用于统计技术的仿真数据。在这个例子中，我们将使用`rnorm()`函数来提供数据项。

```
> set.seed(1234) # Set seed to get same data set each time
> par(mar=c(0,0,0,0)) # Set plot margins
> # Define numeric vector, length=12
> # vector of means: 1 1 1 1 2 2 2 2 3 3 3 3
> x <- rnorm(12, mean=rep(1:3, each=4), sd=0.2)
> # Define numeric vector, length=12
> # vector of means: 1 1 1 1 2 2 2 2 1 1 1 1
> y <- rnorm(12, mean=rep(c(1,2,1), each=4), sd=0.2)
> # x and y coordinates to create 3 clusters
> plot(x,y, col="dark green", pch=19, cex=3)
> # Display integer labels to the upper-right of the dot
> text(x+0.05, y+0.05, labels=as.character(1:12))
```

上面的代码生成了两个向量，`x`和`y`，充当图表中的坐标轴。注意到，用这种方式使用`rnorm()`函数可以创建3个聚类。使用参数`mean`和参数`sd`，来确保有序对大概地集合在一起形成了聚类。生成的图如图8-3所示。你可以很容易地挑选出聚类：我已经在每个聚类附近手工地画了一个红色的椭圆。可以用不同的参数`mean`和`sd`值来改变`x`和`y`向量进行试验，可以看到点的分布不同，聚类也不会总是这么好辨认。

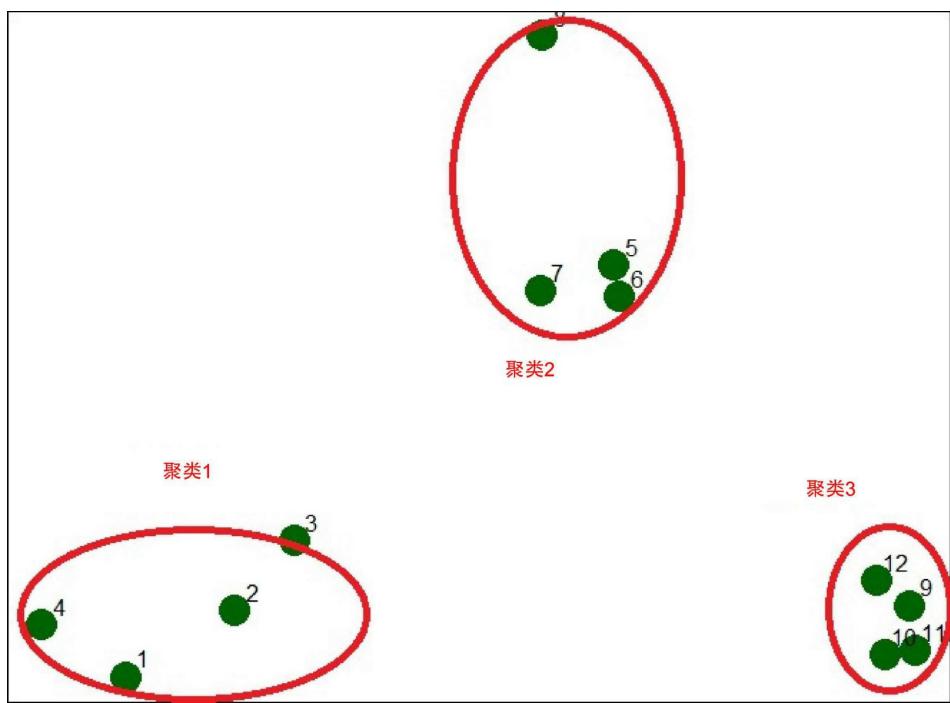


图8-3 仿真数据及的聚类图展现了数据值之间的关系

8.3 分级聚类

分级聚类算法（hierarchical clustering algorithm）比较所有的数据点对，然后将距离最近的合并成一对。用stats包中的hclust()函数可以在R中进行分级聚类。为了执行分级聚类，我们需要定义一个距离测度和将两条观测合并的方法。我们使用欧几里得距离来作为相异性的衡量标准。

算法按照下列步骤进行：

1. 计算每一对数据点/聚类之间的距离。点A和点B的距离通过距离函数来计算。聚类A和聚类B之间的距离首先要计算所有点对（一个点从A聚类中选择，另一个点从B聚类中选择），然后选出其中距离最小/最大/平均值的点。
2. 把两个最接近的点/对加到一个聚类中。重复步骤1，直到只剩下一个较大的聚类。

分级聚类是一种形成聚类的聚合方法（自底向上）。首先找到两条最接近的观测数据或是变量，将它们合并成一个大型观测，然后，使用剩下的观测加上你合并好的这两条，寻找下一对最接近的观测数据。这个过程不断重复，直到所有的观测都被合并到一个大对象中。因为分级聚类的复杂度很高，通常用在数据点数目不太大的情况下。

将分级聚类过程的前几步进行可视化会对你有帮助。分级聚类的可视化表达是一个树形结构图，叫做树状图。基于上面的聚类图表，我们将从鉴别最接近的两个点开始（图中的5和6）。这两个点将通过对它们的x和y值取平均数合并到一起。同时，生成的树状图将有前两个分支，点5和6。接下来，我们要寻找第二组最接近的两个点，发现是点10和11，把它们也合并起来。生成的树状图将有另两个点10和11的分支。一直继续下去，判断下一组最近的两个点（也可以是合并点），不断进行合并。最后生成的树状图如图8-6所示。注意到，最底层的分叉是最早两组最接近的点，5和6，以及10和11。我们可以看到，10和11的合并点接下去与点9和12的合并点合并。最后，在树状图的最顶层，所有点都合并在一起了。树状图的目标是把图中距离接近的点都集合起来。例

如，点5、6、7和8靠的很近。更进一步，点1、2、3和4比较接近，点9、10、11和12比较接近。这符合我们在图8-4上用肉眼看到的结果。

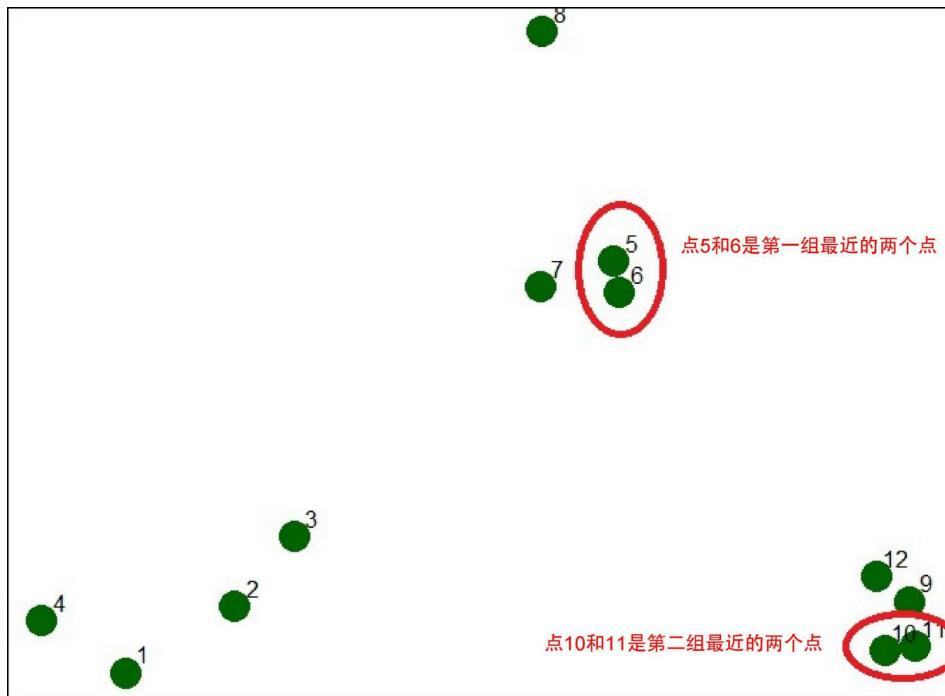


图8-4 分级聚类过程

让我们继续使用8.2节中的仿真数据集。首先，将x和y值加载到数据框中。然后，使用dist()函数计算欧几里得距离。注意，除了曼哈顿距离之外，还有其他几种特殊用途的距离衡量标准。

```
> dataFrame <- data.frame(x=x, y=y) # 12x2 data frame
# Calculate distance between 12 points observed
# (distance between columns)
> distxy <- dist(df) # Default distance method=Euclidean
#distxy <- dist(df, method="minkowski") # Class=dist!!
# Produce cluster object. hclust() requires a dist object
# Returns a hclust object
> hClustering <- hclust(distxy, method="complete")
# Plot dendrogram showing 3 clusters
> plot(hClustering)
> cutree(hClustering, h=1.5) # Will yield fewer clusters
[1] 1 1 1 1 2 2 2 3 3 3 3
> cutree(hClustering, h=0.5) # Will yield more clusters
[1] 1 2 2 1 3 3 3 4 5 5 5 5
```

图8-5展示了dist()函数的结果，计算了数据框中每个点之间距离。例如，点1和点2之间的距离是0.34，点1和点3之间的距离是0.57，等

等。矩形上半部分的值没有包括在内，因为和底部部分的值是相同的。除此之外，矩阵对角线的值也没有展示，因为任何点与自身的距离都是0。

```
> dist(df)
  1      2      3      4      5      6      7      8      9      10     11
2 0.34120511
3 0.57493739 0.24102750
4 0.26381786 0.52578819 0.71861759
5 1.69424700 1.35818182 1.11952883 1.80666768
6 1.65812902 1.31960442 1.08338841 1.78081321 0.08150268
7 1.49823399 1.16620981 0.92568723 1.60131659 0.21110433 0.21666557
8 1.99149025 1.69093111 1.45648906 2.02849490 0.61704200 0.69791931 0.65062566
9 2.13629539 1.83167669 1.67835968 2.35675598 1.18349654 1.11500116 1.28582631 1.76460709
10 2.06419586 1.76999236 1.63109790 2.29239480 1.23847877 1.16550201 1.32063059 1.83517785 0.14090406
11 2.14702468 1.85183204 1.71074417 2.37461984 1.28153948 1.21077373 1.37369662 1.86999431 0.11624471 0.08317570
12 2.05664233 1.74662555 1.58658782 2.27232243 1.07700974 1.00777231 1.17740375 1.66223814 0.10848966 0.19128645 0.20802789
  .
```

图8-5 dist()函数的输出

hclust()函数执行的是实际的分级聚类。这个函数取了dist对象，还有一个可选参数来表明使用的连接方法。默认方法是complete，也可以选择average和single。最长距离法就是在聚类操作之前，比较最远的点（最大的相异性）。与此相反，最短距离法使用聚类之间最小的相异性。或者，类平均法取的是接近的点的平均值（也就是说，它计算了x和y坐标的平均值），将它们合并在一起得到新的点，然后再比较新点之间的距离，得到聚类之间的距离。根据不同的连接方法，你将得到大不相同的聚类图，如图8-6所示。类平均法和最长距离法通常能产生比较均衡的聚类。技巧是使用不同的连接方法产生树状图，再考虑哪些聚类在这个问题领域讲得通。

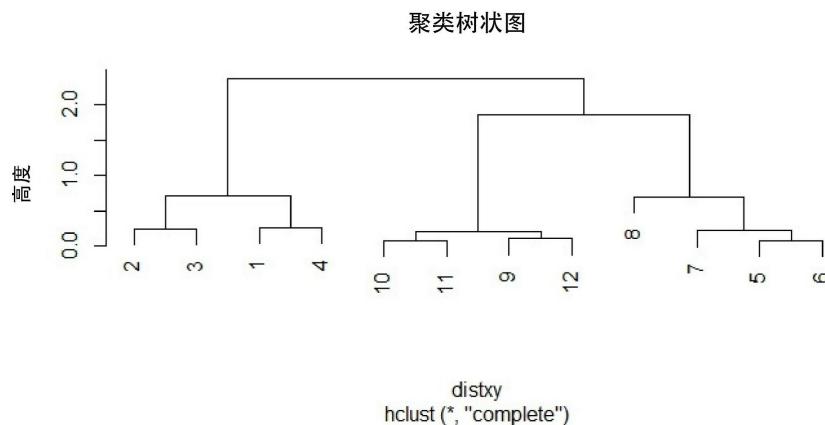


图8-6 仿真数据集的聚类树状图

为了确定点在哪个聚类中，你必须对树状图进行分割。分级聚类完成后，用`cutree()`函数对树进行分割。如果你在高度为1.0到1.5之间分割，将得到3个聚类，如图8-7所示。

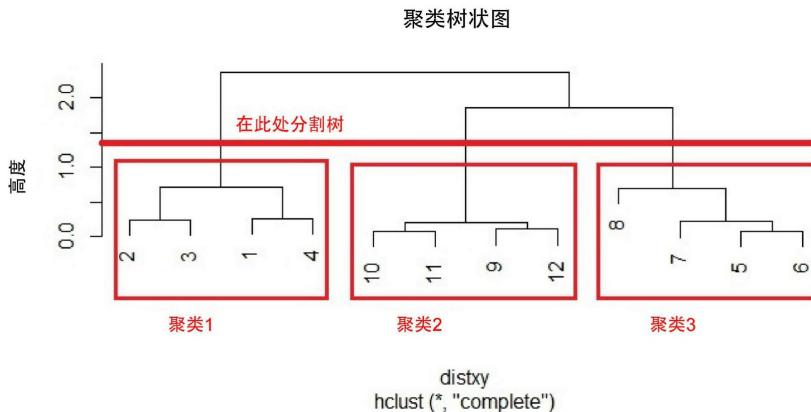


图8-7 在较高层分割树状图可以得到较少的聚类

如果在0.5处对树进行分割，我们将得到更多的聚类。图8-8展示了在较低高度上分割的结果。采用多种分割点的意义在于，分级聚类需要你仔细地考虑每个分割高度和生成的聚类，看它们对正在分析的问题域是否有意义。一些高度产生的聚类能够提供对数据的深刻见解，而其他的聚类可能是无意义的。在这个阶段，与领域专家配合工作进行分析是很关键的。

另一种评估分级聚类的方式是使用热图来考虑经过聚类之后的定量数据。函数`heatmap()`将数据矩阵中的行和列聚集在一起。热图使用分级聚类算法来聚集行和列，然后为每个坐标轴画出树状图。在这个例子中，列聚类不太有趣，因为只有两个变量x和y。但是针对这些变量的观测（行）能够产生有趣的聚类。热图能帮助你看到定量数据中的模式。

```
> set.seed(143)
> dataMatrix <- as.matrix(dataFrame)[sample(1:12),] # 12x2
> heatmap(dataMatrix)
```

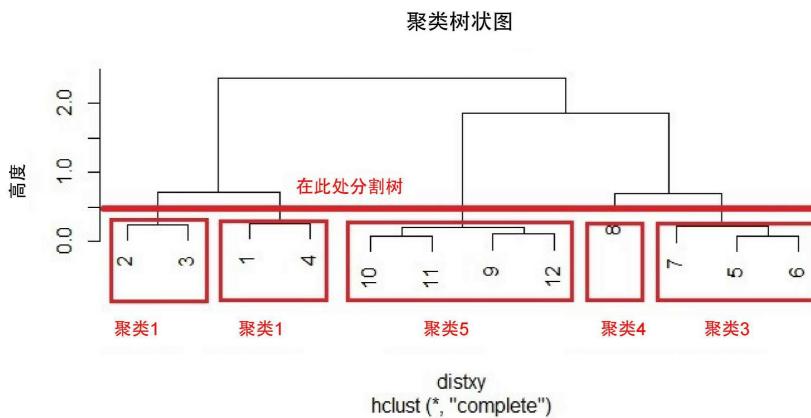


图8-8 在较底层分割树状图可以得到较多的聚类

现在让我们用非仿真数据集——我们熟悉的iris数据集——来举例说明分级聚类。在下面的代码中，我们获得了包含40条观测的随机样本，使用欧几里得距离来计算距离矩阵，使用类平均法计算聚类，然后绘制出树状图，如图8-9所示。注意我们如何基于Species变量来标注得到的聚类。

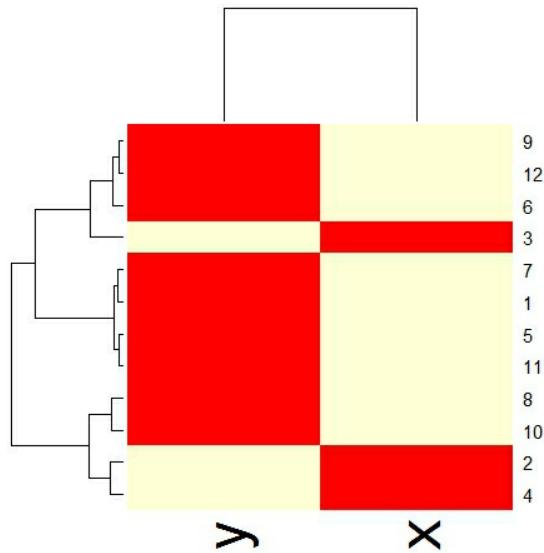


图8-9 变量x和y每个轴上树状图的热图

我们在iris树状图上不断往上看，一些叶子开始合并为分支。这对应为相似的观测。合并出现得越早（在树的越底部），观测组内越相似。从另一方面说，观测合并得越晚（接近树的顶部），观测之间的差异越大。更准确地来说，对于任意两条观测，我们可以在树中寻找这两

条观测第一次合并的分支。这次合并的高度标志着这两条观测的相似程度。因此，在树的最底部合并的观测非常相似，而在接近树的顶端合并的观测则更倾向于相差悬殊。总之，我们可以根据两条观测第一次合并的分叉在树状图上的纵轴位置（不能依据水平轴），对两条观测的相似性下结论。

在不同的垂直高度对树状图进行分割可以产生不同的聚类。在实际使用中，你可以通过肉眼观察树状图，然后基于合并的高度和希望得到的聚类数目选择一个合理的聚类数量。在iris数据集的例子中，我们可以在某一位置分割树，得到四个聚类。

下面的代码会针对iris数据集生成一张集群树状图，如图8-10所示。

```
> data(iris)
> # Get a sample from the iris data set
> # Randomly choose 40 observations from iris
> iris_sample <- iris[sample(1:150, 40),]
> distance_metric <- dist(iris_sample[,-5], method="euclidean") > # dist object
> # Using hclust() from stats package using "average" cluster
> cluster <- hclust(distance_metric, method="average")
> # Plot the cluster dendrogram
> plot(cluster, hang=-1, label=iris_sample$Species, main="Iris Data Set Clusters"
```

让我们总结一下可以从分级聚类中获取的几个关键要点。它能为变量和观测之间的关系提供思路，但是下面这些行为可能导致图像变化：

- 如果修改了一些数据点，删掉了一些数据点，或是发现某些数据点中存在缺失值，可能会得到完全不同的聚类。这是因为距离测量随着其他数据点的变化而变化。
- 如果在数据集中有不同的缺失值，图像会发生变化。
- 如果你选择了不同的距离度量方法，图像会发生变化。
- 如果你选择了不同的连接（合并）策略，图像会发生变化。
- 如果你对一个变量的数值进行标准化，图像也会发生变化。如果你对一个点集进行了标准化，没有处理其他点集，那么你将得到非常不一样的聚类结果。

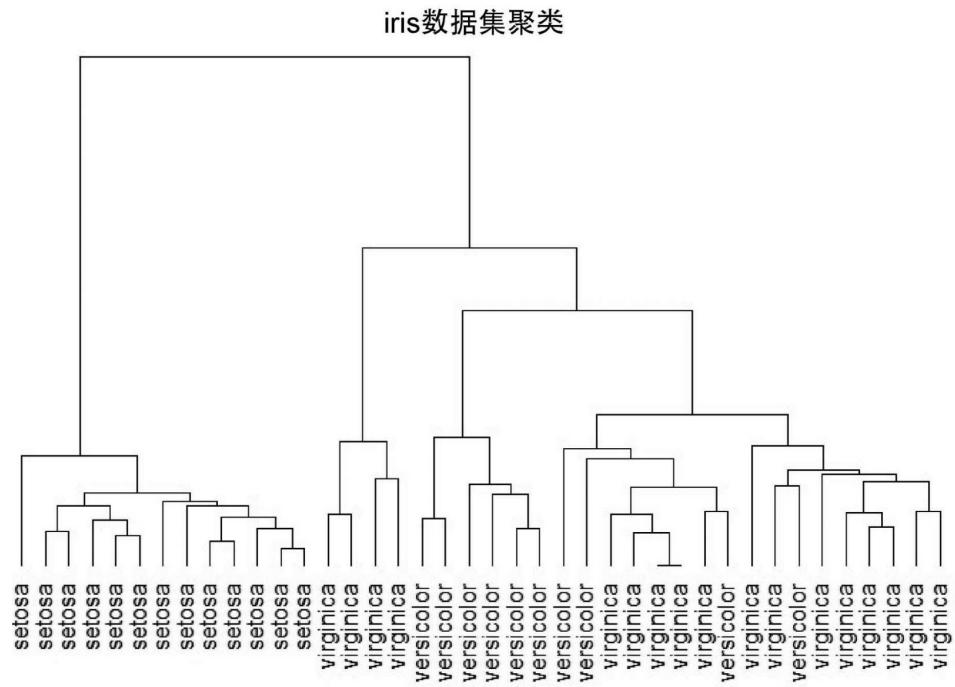


图8-10 iris数据集的聚类树状图

分级聚类是一种确定性算法，如果使用相同的数据点、相同的距离度量以及相同的合并策略，最终就能生成相同的树状图。此外，选择在哪个位置对树进行分割通常都不太明显。分级聚类用在可视化探究方面比验证分析上更有价值。

8.4 K-均值聚类

现在，让我们把注意力转向另一个重要的非监督学习算法：K-均值聚类。K-均值聚类和分级聚类不同，它是一种分割法而不是聚合法。分割法从全部数据点开始，试图将它们拆分到固定数目的聚类中。K-均值应用在定量变量中。聚类的数目提前确定了，然后我们必须猜测这些聚类的中心（质心）可能在什么位置。接下来我们要把数据点分配到最近的质心（centroids）中，然后通过这种集群方法重新计算质心位置。

K-均值聚类的必备条件如下：

- 一个定义好的距离测量标准。K-均值聚类中使用的主要距离测度是欧几里得距离。
- 提前选好聚类的数目。通常，在开始的时候要凭直觉选一个较大的聚类数据，应用到K-均值聚类中，看这些聚类是否合理，然后减少聚类的数目。
- 对聚类质心的初识估计。

在K-均值聚类的最后，你将获得对聚类中心（质心）的最终估计，并将每个数据点都分配到每个聚类中。应该注意K-均值并不是确定的，也就是说，在开始时选取不同的聚类数目，将产生不同的算法迭代次数。为了演示这一过程，让我们重新回到分级聚类中使用过的12个点的仿真数据集中，用它来看看K-均值聚类算法是如何工作的。就像之前一样，假设我们希望得到3个聚类，对每一个聚类质心（图中十字交叉符号的点）提供一个初识估计值（用来展示这一进程是如何工作的）。图8-11展示了K-均值算法的起始点。

现在我们需要把所有的点都分配到欧几里得距离最近的质心中。所以伴随这一过程，我们得到了这样的集群：点（4、8）最接近质心1，点（1、2、3）最接近质心2，点（5、6、7、9、10、11、12）最接近质心3。图8-12展示了生成的聚类。

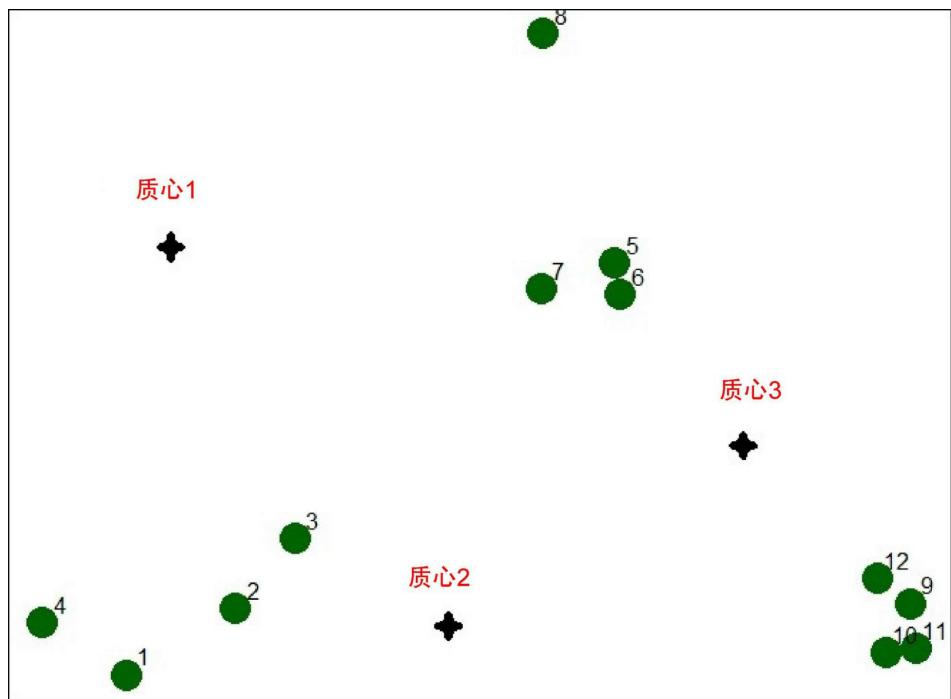


图8-11 带有质心估计的K-均值聚类起始点

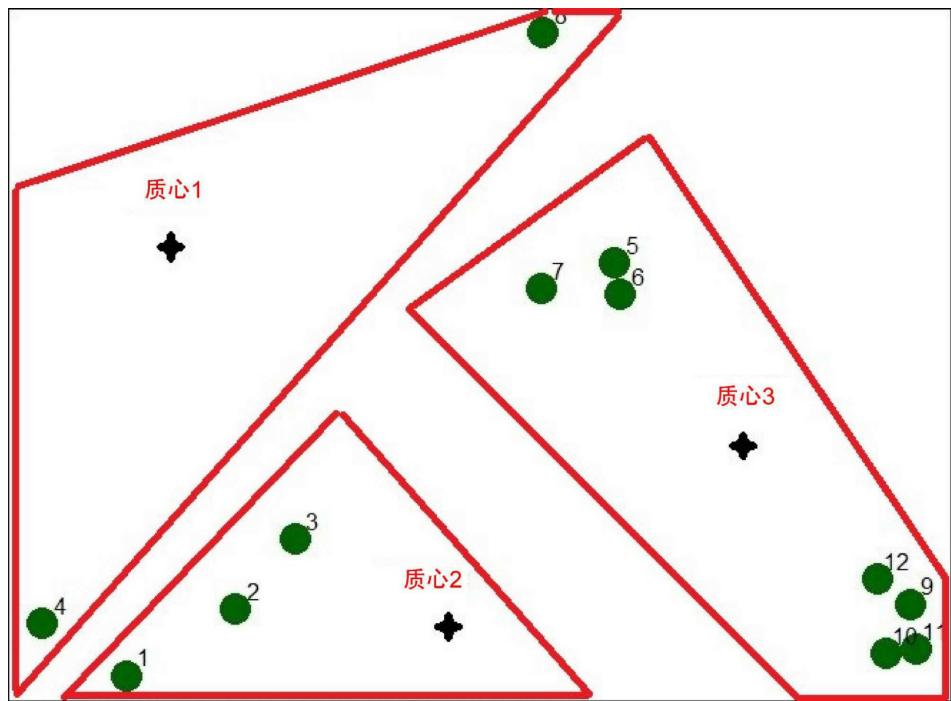


图8-12 K-均值聚类的第一步: 分配集群

K-均值算法的下一步是重新计算质心。我们计算出点 (4、8) 的x平均值和y平均值，得到聚类1的新中心。同理，计算出点 (1、2、3)

的x平均值和y平均值，得到新的质心2。最后，对质心3执行相同的重计算步骤。然后重新计算出每个点到每个新质心的距离，然后把点重新分配到合适的质心。通过不断地重新更新质心来迭代这个过程，你会发现质心自己也会调整到特定的集群中。在应用了K-均值聚类算法之后，如图8-13所示，决定了最终的聚类。

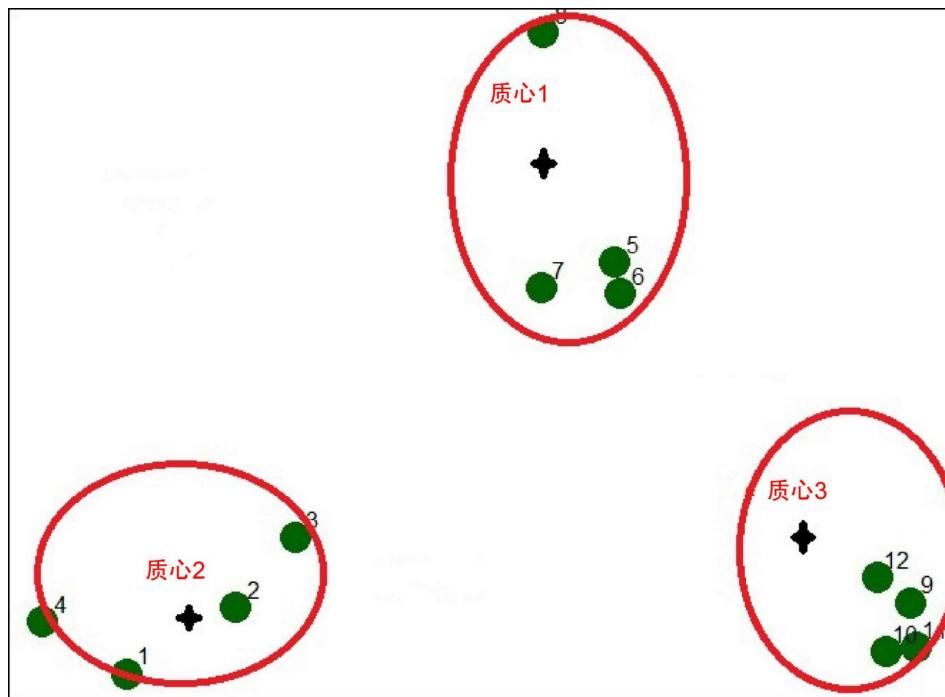


图8-13 通过执行K-均值算法得到的聚类

在R中，可以使用基础stats包内的kmeans()函数来执行K-均值聚类。在这个例子中，我们要给函数传递一个等待聚类的观测数据的数据框，加上需要确定的中心数量，并使用centers参数（随机选择位置）。或者，我们可以在centers参数中传递聚类中心的横纵坐标矩阵。我们也可以用参数iter.max传递需要执行的迭代次数和最大迭代次数，否则当这个算法不收缩时，你可能要花费大量的K-均值处理时间，在处理较大的数据集时尤其如此。iter.max的默认值是10，但是你可能想要用更高的数值进行试验。最后，你可以用参数nstart指定算法的“启动”数目。因为这个算法不是确定的，如果给定完全相同的数据和不同的centers值，你可能发现它收敛到不同的集群集。所以，在nstart的帮助下，你可以让K-均值算法重新计算多次，由此得到多次启动获得的平均集群。

如下是执行K-均值聚类的示例代码。注意，因为在使用centers参数

时，K-均值算法使用了随机数发生器来得出中心点，在每次运行代码之前都需要设置一个种子值，来得到可复验的结果。

```
> df <- data.frame(x,y)
> set.seed(1234)
> # Create a kmeans object with initial number of centroids=3
> kmeans1 <- kmeans(df,centers=3, iter.max=10)
> names(kmeans1)
[1] "cluster"   "centers"    "totss"     "withinss"   "tot.withinss"
[6] "betweenss" "size"      "iter"      "ifault"
> # Show which cluster each data point assigned to
> kmeans1$cluster      # This is an integer vector
[1] 3 3 3 3 1 1 1 1 2 2 2 2
> kmeans1$centers
  x       y
1 1.9906904 2.0078229
2 2.8534966 0.9831222
3 0.8904553 1.0068707
```

我们从names(kmeans1)中可以看到，kmeans()函数返回了kmeans1对象，其中包括用kmeans1\$cluster表示的聚类、用kmeans1\$centers表示的这些聚类的中心以及关于这些聚类估计的其他变量信息。K-均值对象中的cluster部分表明了每个数据点被分配到哪个集群中。在这个例子中，前四个数据点分配到第三个集群，接下去四个数据点分配到第一个集群，剩下的四个数据点分配到第二个集群。

现在绘制出聚类图，我们可以用下面的代码来使用K-均值对象中的cluster和centers部分。生成的图如图8-14所示。图中的十字形展示了每个聚类的中心。一般来说，实际的聚类边界和中心并不会像例子这么清楚。

```
> par(mar=rep(0.2,4))
> # Plot the data points using unique colors for each cluster
> plot(x,y,col=kmeans1$cluster,pch=19,cex=2)
> # Draw crosses showing cluster centers
> points(kmeans1$centers,col=1:4,pch=3,cex=3,lwd=3)
```

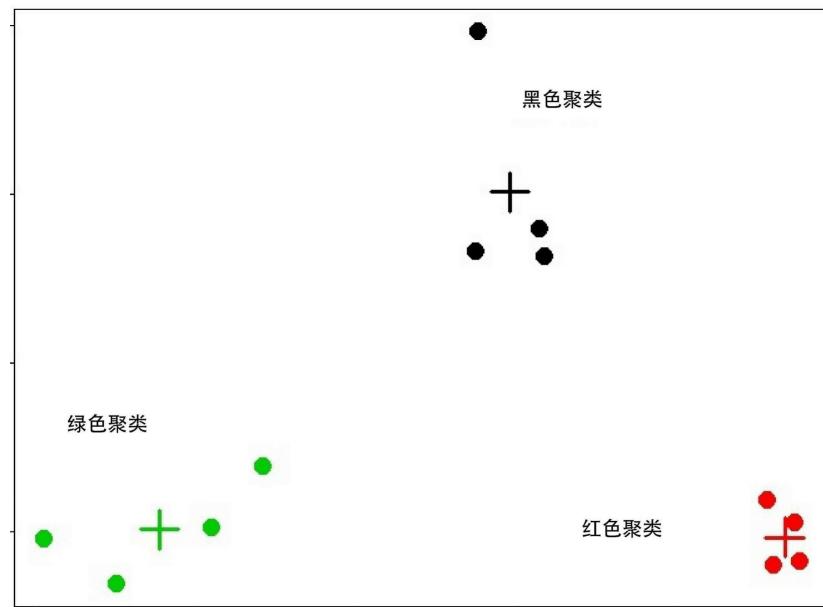


图8-14 绘制kmeans()函数的结果

8.5 主成分分析

主成分分析（PCA，Principal Component Analysis）是计算主成分，用以更好地理解数据的一种处理方法。PCA被当做是一种非监督学习工具，因为它只涉及一组特征变量，没有关联的响应变量。PCA也是探索性数据分析和数据可视化的一种有力工具。本节将探索这一过程，凭借在给定数据集上执行PCA，我们可以获得信息增值。

我们先来讨论为什么需要主成分。假设你有一个数据集，其中包含12个特征变量。探索性数据分析中的一个工具就是比较每个变量下其他变量的散点图。问题在于，对于12个变量，你需要66张图（特征变量组合的数目）。如果你有24个变量，情况甚至会更加难以应对。此外，任何一张图表中的信息只代表了一小部分数据集信息。处理这个问题的一个更好的方式是找到数据的低维替代品，尽可能多地捕捉信息。在12个特征变量的例子中，我们想要找到一个二维的替代品来，包含尽可能大的方差。然后我们可以在这个低维空间中绘制观测的图表。

PCA是能实现这一目标的一种方法。PCA计算出更少的维度来代表数据中最令人关注的部分。它计算主成分的方式可以在数学（线性代数）书中找到，这已经超出了本书的范围。从本质上讲，PCA试图将原始变量转换成一组新的变量，新变量有如下特征：（1）数据集中变量的线性组合；（2）互相不相关；（3）根据能解释的原始变量的方差总量来进行排序。

在R中，我们有多种不同的选择来执行PCA。下面是简短的列表：

- 使用stats包中的prcomp()函数。这个函数是一个数值稳定的程序，是计算PCA的优先选择。它基于奇异值分解（SVD， Singular Value Decomposition）算法。
- 使用princomp()函数，它也在stats包中。这个函数的稳定性稍低，但是有更多的特征。
- 使用基础R包中的svd()函数。PCA基于奇异值分解法，使用它需要更多的数学知识。最好的方法是用到像svd()这样包含了SVD的函数。

- 通过计算协方差矩阵，用它计算特征值和特征向量，然后使用矩阵乘法来估计主成分，“手工”计算PCA。这个方法对完全理解PCA的工作方式很有帮助，但是需要数学方面的背景知识。

执行PCA的后两种方法都超出了本书的范围，因为它们需要更多的数学知识。我们在这里只考虑第一种方法，即使用prcomp()。

我们将使用R中的综合特征计算主成分，并且让它们为我所用。下面是我要遵循的步骤：

- 从包含若干特征变量的未标记数据集开始。
- 然后我们将对数据执行预处理：均值归一化（每个特征的均值都为0），基于你的数据是否有不同的值域（举例来说，一栋房子的平方英尺范围可能是1 500~5 000，但是卧室的数目范围可能是1~4），可以选择特征缩放。特征缩放可以把数值调整到可比较的范围。
- 我们将要计算前几种主成分和载荷向量，包含了后面会用到的信息。使用载荷，我们可以计算出主成分的得分。在新的主成分坐标系中，每条观测数据的位置叫做得分，由原始变量的线性组合和载荷向量的权重来计算。
- 计算出主成分之后，基于主成分之间的关系，画出低维的数据图。
- 我们生成了一张叠图，在一个页面中同时展示主成分得分和载荷向量。
- 使用碎石图描绘每个主成分解释的方差比例，来决定解释数据中可接受的方差总量需要多少个主成分。也就是说，我们可以获得多低的维度，并且依然能保持数据的本质特征。

我们将会看到，PCA在选择如何使用主成分方面涉及很多主观努力。事实上，没有公认的客观方法可以决定多少主成分是足够的、什么时候需要更多主成分。一旦花费了足够多的时间使用PCA，你就能熟悉如何去使用它。

请注意，主成分分析并不能总是降低数据的维数。事实上，如果原始变量已经不相关了，那么这个分析起不到任何作用！在这种情况下，主成分和原始的变量相同。当原始的变量高度相关时，能得到最好的结果，因为很可能大部分的变量测量的都是相同的东西，所以在原始数据

中存在大量冗余。用于PCA的数据集中不应该包括分类变量，只能使用连续变量。

叠图通常和主成分分析一起使用，用于辅助解释分析结果。叠图能在一个平面内同时展示主成分得分和代表每个原始变量对这些成分贡献程度的向量。在主成分分析的背景下，它绘制了前两个成分和原始变量。后者作为方向向量，能指示主成分和原始变量之间关系的方向。主成分分析得到的成分和变量数目一样多。通过确定保留的方差百分比，你可以选择一些希望使用的主成分。好在这个过程能让成分按照代表的方差总数来排序。我们可以用USArrests数据集的扩展案例来演示PCA，该数据集是基础R包的一部分。让我们从熟悉数据集开始。这个数据集中展示了在1973年每100 000位居民中因伤害、谋杀和强奸而被拘留的统计数据，包含了美国50个州的数据。同时也给出了城市居民的人口比例。

```
> data(USArrests)      # 50x4 data frame  
> names(USArrests)  
[1] "Murder" "Assault" "UrbanPop" "Rape"
```

在用新数据集进行分析时，有一个标准，你应该使用summary()函数来获取一些基本的统计结果：

```
> summary(USArrests)  
   Murder          Assault         UrbanPop          Rape  
Min.   : 0.800   Min.   : 45.0   Min.   :32.00   Min.   : 7.30  
1st Qu.: 4.075   1st Qu.:109.0   1st Qu.:54.50   1st Qu.:15.07  
Median  : 7.250   Median :159.0   Median :66.00   Median :20.10  
Mean    : 7.788   Mean   :170.8   Mean   :65.54   Mean   :21.23  
3rd Qu.:11.250   3rd Qu.:249.0   3rd Qu.:77.75   3rd Qu.:26.18  
Max.   :17.400   Max.   :337.0   Max.   :91.00   Max.   :46.00
```

我们可以看到，每个变量的平均值都大不相同。例如，伤害罪的平均数量大约是谋杀罪的8倍。我们也可以使用apply()函数检查变量的方差。注意它们的区别非常大。这意味着在执行PCA之前，我们必须对变量进行缩放。在使用PCA时，这是一个重要的诊断任务，可以将数据的平均值转换为0，标准差转换为1。

```
> apply(USArrests, 2, var)  
   Murder     Assault   UrbanPop       Rape  
18.97  047 6945.16571  209.51878  87.72916
```

现在我们可以使用prcomp()函数执行PCA。这个函数的默认操作就是将变量聚集，使得均值为0，所以无需调整参数center=TRUE。不过，你应该指定scale=TRUE，这样变量的标准差就能调整为1。

```
> pca <- prcomp(USArrests, scale=TRUE, center=TRUE)
```

prcomp()函数提供了许多有用的输出。完整的列表可以这样获得：

```
> names(pca)
[1] "sdev"    "rotation" "center"  "scale"   "x"
```

下面展示的pca\$center部分包含了用于缩放的变量的平均值。应该把它和之前summary()得到的数据进行对比，看看它们的匹配程度。pca\$scale部分包含了变量的标准差。如果你计算标准差的平方，按照定义，将会得到方差。这些计算出的值可以匹配到之前apply()函数得到的值。pca\$sdev部分包含了主成分的标准差。

```
> pca$center
Murder Assault UrbanPop      Rape
  7.788   170.760   65.540   21.232
> pca$scale
Murder Assault UrbanPop      Rape
  4.35   5510.83337661 14.474763  9.366385
> pca$scale^2
Murder Assault UrbanPop      Rape
 18.97    047.694516571 209.51878  87.72916
```

接下来是pca\$rotations部分，它在PCA的应用中非常重要。旋转矩阵包含了主成分载荷，即pca\$rotations的每一列都是对应的主成分载荷向量。在我们的例子中，可以看到四个载荷向量：PC1、PC2、PC3和PC4，这表示有4个主成分。一般来说，如果数据集中观测的数量比变量数量多，那么主成分的数量应该等于变量数量。当你在USArrests数据集和pca\$rotations上使用矩阵乘法的数学概念时，你将会得到坐标系旋转之后数据的坐标。这些坐标叫做主成分得分。我们马上就会看到，如果用pca\$x部分作为替代，就无需再使用矩阵乘法了。

```
> pca$rotation
          PC1        PC2        PC3        PC4
Murder -0.5358995  0.4181809 -0.3412327  0.64922780
Assault -0.5831836  0.1879856 -0.2681484 -0.74340748
UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
Rape    -0.5434321 -0.1673186  0.8177779  0.08902432
```

pca\$x部分包含了旋转后数据的值——中心矩阵（如果需要的话，还有缩放矩阵）乘以旋转矩阵pca\$rotation。pca\$x的维度是50乘4，就是观测的数量乘主成分的数量。pca\$x包含的是降维后的数据集，而不是一开始的高维数据集。接下来，我们看看如何从pca\$x中选择数据。

现在在prcomp对象pca上执行一些探索性数据分析。可以使用biplot()函数生成图8-15。这个叠图能帮助我们绘制主成分之间的相互关系，来生成数据的低维视图。称为“叠图”的原因是，它在一张图表中同时展示了主成分得分和主成分载荷。图中的州名代表了前两个主成分的得分。箭头指示了前两种主成分的载荷向量，向量的轴线位于图的右上部分。

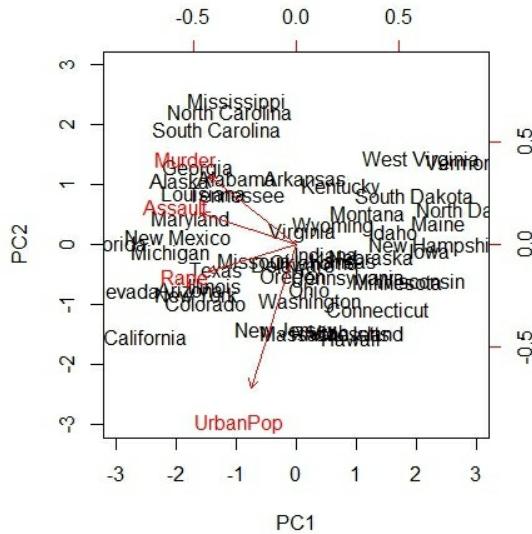


图8-15 前两种主成分的图

例如，单词“Assault”的中心位于(-0.58, 0.19)，-0.58表示第一个主成分上的载荷，0.19表示第二个主成分上的载荷。箭头指示的是方差最大的方向，也就是说，数据区别最大的特征空间的方向。

```
> biplot(pca, scale=0)
```

现在，我们可以把注意力转向PCA中最重要的一个方面，即选择能解释数据中最大比例的方差的主成分。这可以降低数据的维度。我们从计算每个主成分解释的方差总量开始，计算pca\$sdev的平方。然后除以

方差的总和，计算每个主成分能够解释的方差比例。

```
> pca_var=pca$sdev^2  
> pca_var  
[1] 2.4802416 0.9897652 0.3565632 0.1734301  
> pve <- pca_var/sum(pca_var)  
> pve  
[1] 0.62006039 0.24744129 0.08914080 0.04335752
```

pve中展示的结果十分显著。我们可以看到，第一个主成分解释了数据中62.0%的方差，第二个主成分解释了24.7%的方差。这可以用一种特殊的图表——碎石图（scree plot）来解释，如图8-16左侧部分所示。碎石图展示了USArrests数据集中4个主成分分别解释的方差比例。图表的右边部分展示了4个主成分能够解释的累计方差比例。在这些图的帮助下，我们可以确定降维。下面是生成这些图表的代码：

```
> par(mfrow=c(1, 2))  
> plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained", :  
> plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative  
Proportion of Variance Explained", ylim=c(0,1), type='b')
```

从图中可以看到，第1个和第2个主成分共同解释了86.7%的数据方差。通过加入第3个主成分，得到95.6%，这样就已经能很好地代表整个数据集了。既然已经对数据和它的主成分有了更深入的了解，现在让我们来探索如何为了目标而使用PCA。

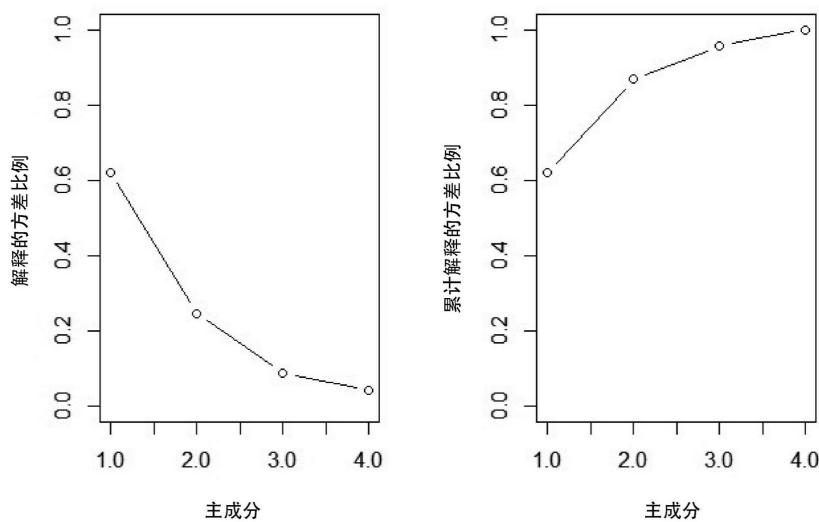


图8-16 碎石图展示了方差的比例和累计解释的比例

每一个新的主成分都可以看做是记录每一条观测的一个新变量。每个州都有Assault值，同样每个州都会有一个PC1值。为了看到每条观测在每个主成分上的值，我们可以使用predict()函数。在这里，d1是一个 50×4 的矩阵。或者，我们可以人工进行计算，将USARests的伸缩后矩阵乘以旋转矩阵。这两个工具是等效的。predict()是一个通用型的R函数，可以用于预测各种模型的结果。在这个例子中，predict()把pca看做一个prcomp对象，然后计算每条观测在每个主成分上的值。

```
> d1 <- predict(pca)
> d2 <- scale(USARests, pca$center, pca$scale) %*% pca$rotation
```

现在我们可以使用前3个主成分来计算新的数据集。在这里，我们已经成功地将四维数据集降到三维数据集。可以使用一张3D图来进行数据可视化，这在之前是无法实现的。我们可以使用scatterplot3d包来绘制图8-17。

```
> d3 <- predict(pca)[,1:3] # 50X3
> d4 <- scale(USARests, pca$center, pca$scale) %*% pca$rotation
[,1:3]
> library(scatterplot3d)
> scatterplot3d(d4[,1], d4[,2], d4[,3], main="3 Principal Components")
```

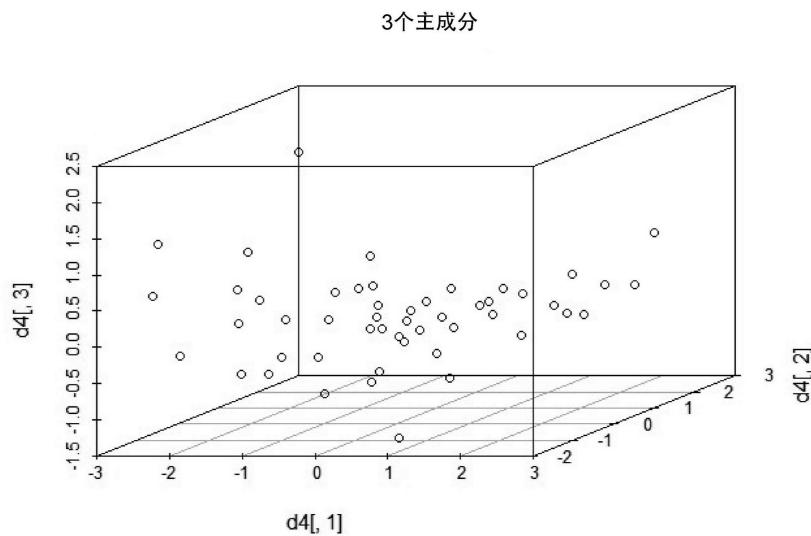


图8-17 使用3个主成分的3D散点图

让我们继续降维过程，用前两个主成分来生成一个二维数据图，前

两个主成分能解释86.7%的方差。生成的2D图如图8-18所示。

```
> d5 <- predict(pca)[,1:2] # 50X2  
> plot(d5[,1], d5[,2], col="blue", main="2 principal components")
```

总而言之，PCA在下面这些应用中用处很大：

- 当你的监督学习特征向量维度很高（例如，在计算机视觉应用中有10 000个特征），可以降低数据的维度来加快用于预测的机器学习算法速度。
- 在这个例子中，只在数据集的特征变量上使用PCA，留出预测变量。将会生成一个较低维的特征集，可以传到神经网络或者逻辑回归这样的学习算法中。
- 通过降低数据的维度，减少数据存储所需的内存/磁盘容量。
- 通过降低维度，可以在2D或是3D图上查看数据，增强可视化效果。

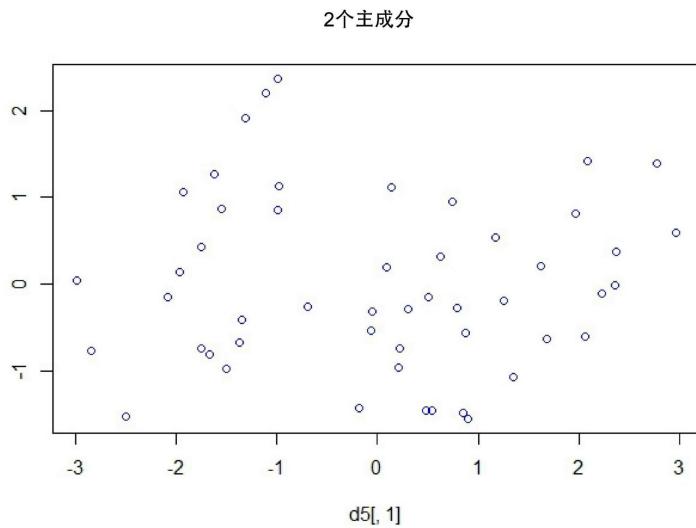


图8-18 使用两个主成分的2D散点图

8.6 小结

在本章中，通过使用一系列不同的工具，我们学习了非监督学习。这些工具包括：

- 分级聚类，一种形成聚类的聚合（自底向上）方法。
- K-均值聚类，一种形成聚类的分割方法。
- 主成分分析，一种为降维而设计的复杂算法。

与在监督学习方法中能获得预测和分类结果不同，在非监督学习中，我们能够获得对数据的直观感受——也就是说，对数据值进行分组。

术语表

预测回头客挑战赛	Acquire Valued Shoppers Challenge
算法交易挑战赛	Algorithmic Trading Challenge
亚马逊云服务	Amazon Web Services
分析	analysis
方差分析	analysis of variance
人工智能	artificial intelligence
人工神经元	artificial neuron
艾萨克·阿西莫夫	Asimov, Isaac
AWS见Amazon Web Services	AWS. See Amazon Web Services
套袋法	bagging
条形图	barplot
BellKor	BellKor
BI见商业智能	BI. See business intelligence

偏差-方差权衡	bias-variance tradeoff
大数据	big data
交叉融合法	blending
提升法	boosting
箱形图	boxplot
商业智能	business intelligence
C++	C++
微积分	calculus
呼叫中心管理	call center management
卡耐基梅隆	Carnegie Mellon
CDC见疾控预防中心	CDC. See Centers for Disease Control and Prevention
疾控预防中心	Centers for Disease Control and Prevention
质心	centroids
客户流失	churn
分类	classification
聚类	clustering
系数矩阵	coefficients matrix

共线性	collinearity
组合数学	combinatorics
逗号分隔值	comma separated value
综合R归档网	Comprehensive R Archive Network
计算机科学	computer science
混淆	confound
混合矩阵	confusion matrix
Cook距离	Cook's Distance
Coursera	Coursera
交叉验证	cross validation
交叉销售	cross-selling
CSV见逗号分隔值 曲度	CSV. See comma separated value curvilinearity
客户细分	customer segmentation
数据框	data frame
数据泄漏	data leakage
数据挖掘师	data miner

数据挖掘	data mining
数据处理	data munging
数据管道	data pipeline
数据采样	data sampling
数据科学流程	data science process of
数据科学韦恩图2.0	Data Science Venn Diagram 2.0
数据科学家	data scientist
数据转换	data transformation. See data munging
数据可视化	data visualization
数据处理	data wrangling. See data munging
需求预测	demand forecasting
树状图	dendrogram
密度图	density plot
维度	dimensionality
dplyr	dplyr
EDA见探索式数据分析	EDA. See Exploratory Data Analysis
edX	edX

集成方法	Ensemble method
由平方偏差导致的误差	error due to squared bias
由方差导致的误差	error due to variance
估计系数	estimated coefficients
欧几里得	Euclidean
Excel	Excel
实验家	experimentalist
探索性数据分析	Exploratory Data Analysis
探索性图表	exploratory plot
解释性图表	expository plot
假阴性	false negative
假阳性	false positive
特征工程	feature engineering
特征变量	feature variable
前馈	feedforward
FN见假阴性	FN. See false negative

FP见假阳性	FP. See false positive
欺诈	fraud
F-统计量	F-statistic
GBDT见梯度提升决策树	GBDT. See Gradient Boosted Decision Trees
GBM见梯度提升机	GBM. See Gradient Boosting Machine
泛化误差	generalization error
GFT见Google流感趋势	GFT. See Google Flu Trends
Google	Google
谷歌关键字	Google Adwords
谷歌分析	Google Analytics
谷歌流感趋势	Google Flu Trends
梯度提升决策树	Gradient Boosted Decision Trees
梯度提升机	Gradient Boosting Machine
Hadoop	Hadoop
热图	heatmap
HPN	Heritage Provider Network
隐藏层	hidden layer

分级聚类算法	hierarchical clustering algorithm
高偏差	high bias
高方差	high variance
直方图	histogram
HPN见Heritage Provider Network	HPN. See Heritage Provider Network
HTML	HTML
超平面	hyperplane
假设	hypothesis
输入数据值	imputing data value
样本内误差	in sample error
内连接	inner join
输入层	input layer
可消除误差	irreducible error
JSON	JSON
Kaggle	Kaggle
可汗学院	Kahn Academy

kernel	kernel
K-最近邻	K-nearest neighbors
KNN见K-nearest neighbors	KNN. See K-nearest neighbors
知识挖掘	knowledge discovery
lambda	lambda
套索	lasso
主导优先级	lead prioritization
最小二乘法	least squares
留一交叉验证	Leave one out cross validation
左外连接	left outer join
线性代数	linear algebra
线性回归	linear regression
Linux	Linux
逻辑回归	logistic regression
LOOCV见留一交叉验证	LOOCV. See Leave one out cross validation
LTV见预测生涯价值 机器学习	LTV. See Predict Lifetime Value machine learning
MacOS	MacOS

曼哈顿距离	Manhattan distance
大规模在线开放课程	massive open online course
数理统计	mathematical statistics
均方误差	mean square error
Microsoft SQL Server 2012 Express	Microsoft SQL Server 2012 Express
误分类率	misclassification rate
缺失值的图表	missing value plot
MIT公开课	MIT OpenCourseWare
Tom Mitchell	Mitchell, Tom
MOOC见大型开放式网络课程	MOOC. See massive open online course
MSE见均方误差	MSE. See mean square error, See mean square error
多元线性回归	multiple linear regression
朴素贝叶斯	Naïve Bayes
Netflix	Netflix
神经网络	neural network
记事本	NotePad

ODBC见开放数据库互联	ODBC. See Open Database Connectivity
开放数据库互联	Open Database Connectivity
Oracle	Oracle
样本外误差	out of sample error
外连接	outer join
异常值	outlier
输出层	output layer
过拟合	overfit
过拟合	overfitting
偏微分方程	partial differential equations
性能评估	performance evaluation
PMI见采购经理人指数	PMI. See Purchasing Managers Index
多项式回归	polynomial regression
POWERDOT	POWERDOT
预测生涯价值	Predict Lifetime Value
预测	prediction
预测变量	predictor variable

素数辐射法	Prime Radiant
主成分分析	Principal Component Analysis
概率论	probability theory
采购经理人指数	Purchasing Managers Index
勾股定理	Pythagorean Theorem
Python	Python
分位数-分位数图	QQ-plot
quantile-quantile plot见QQ图	quantile-quantile plot. See QQ-plot
R	R
Reasons for using	Reasons for using
R包	R packages
R2	R2
随机森林	random forest
随机抽样	random sampling
randomForest	randomForest
R博客	r-bloggers

RBM见受限的波尔兹曼机	RBM. See Restricted Boltzmann Machines
重复购买	reactivation
不可消除误差	reducible error
正则化	regularization
残差	residuals
响应变量	response variable
受限的玻尔兹曼机	Restricted Boltzmann Machines
再代入误差	resubstitution error
Revolutions Analytics公司	Revolutions Analytics
岭回归	ridge regression
右外连接	right outer join
RMSE见root mean square error	RMSE. See root mean square error
均方根误差	root mean square error
决定系数	R-squared
RStudio	RStudio
SaaS见软件即服务	SaaS. See Software as a Service
销售预测	sales forecasting

Samuel, Arthur	Samuel, Arthur
散点图	scatterplot
科学方法	scientific method
碎石图	scree plot
收缩	shrinkage
显著编码说明	significance codes legend
显著星号	significance stars
奇异值分解	Singular Value Decomposition
软件即服务	Software as a Service
SOP见标准作业程序	SOP. See Standard Operating Procedure
Spark	Spark
SQL	SQL
标准误差	standard error
标准作业程序	Standard Operating Procedure
统计F检验法	statistical F-test
强分类器	strong learner

监督学习	supervised learning
支持向量机	Support Vector Machine
SVM见支持向量机	SVM. See Support Vector Machine
t统计量	t statistic
测试集	test set
测试	testing
理论家	theorist
TN见真阴性	TN. See true negative
TP见真阳性	TP. See true positive
训练误差率	training error rate
训练集	training set
真阴性	true negative
真阳性	true positive
t统计量	t-statistics
Tukey, John	Tukey, John
Twitter	Twitter
加州大学欧文分校的机器学习资源库	UC Irvine Machine Learning Repository

欠拟合	underfit
欠拟合	underfitting
UNIX	UNIX
未标记	unlabeled
非监督学习	unsupervised learning
提升销售	up-selling
变量的假设几率	variable p-value
弱分类器	weak learner
抓取网页	web scraping
Wickham, Hadley	Wickham, Hadley
Windows	Windows
YouTube	YouTube

欢迎来到异步社区！

异步社区的来历

异步社区(www.epubit.com.cn)是人民邮电出版社旗下IT专业图书旗舰社区，于2015年8月上线运营。

异步社区依托于人民邮电出版社20余年的IT专业优质出版资源和编辑策划团队，打造传统出版与电子出版和自出版结合、纸质书与电子书结合、传统印刷与POD按需印刷结合的出版平台，提供最新技术资讯，为作者和读者打造交流互动的平台。

 同步社区
人民邮电出版社

软技能 

书架 购 (0) 通知 

首页 图书 电子书 文章 写作

新年新气象

社区UI全新改版，崭新面貌迎接2017！为答谢社区用户，
即日起到1月26号 全场电子书8折优惠！



前端开发 

数据科学 

编程语言 

移动开发 

游戏开发 

机器学习&深度学习 [更多>>](#)

 Python 机器学习
预测分析核心算法

 贝叶斯方法
概率编程与贝叶斯推断

 机器学习
项目开发实战

 贝叶斯思维：
统计建模的Python学习法

[免费电子书
Free eBook](#)  立即获取

[我要写书
Write for Us](#)  立即查看

近期活动

社区里都有什么？

购买图书

我们出版的图书涵盖主流IT技术，在编程语言、Web技术、数据科学等领域有众多经典畅销图书。社区现已上线图书1000余种，电子书400多种，部分新书实现纸书、电子书同步出版。我们还会定期发布新书书讯。

下载资源

社区内提供随书附赠的资源，如书中的案例或程序源代码。

另外，社区还提供了大量的免费电子书，只要注册成为社区用户就可以免费下载。

与作译者互动

很多图书的作译者已经入驻社区，您可以关注他们，咨询技术问题；可以阅读不断更新的技术文章，听作译者和编辑畅聊好书背后有趣的故事；还可以参与社区的作者访谈栏目，向您关注的作者提出采访题目。

灵活优惠的购书

您可以方便地下单购买纸质图书或电子图书，纸质图书直接从人民邮电出版社书库发货，电子书提供多种阅读格式。

对于重磅新书，社区提供预售和新书首发服务，用户可以第一时间买到心仪的新书。

用户帐户中的积分可以用于购书优惠。100积分=1元，购买图书时，在里填入可使用的积分数值，即可扣减相应金额。

特别优惠

购买本电子书的读者专享异步社区优惠券。 使用方法：注册成为社区用户，在下单购书时输入“**57AWG**”，然后点击“使用优惠码”，即可享受电子书8折优惠（本优惠券只可使用一次）。

纸电图书组合购买

社区独家提供纸质图书和电子书组合购买方式，价格优惠，一次购买，多种阅读选择。

Wireshark网络分析的艺术

作者：林沛满
责编：傅道坤
分类：计算机科学 > 安全与加密 > 网络安全

Wireshark是当前最流行的网络包分析工具。它上手简单，无需培训就可入门。很多棘手的网络问题遇到Wireshark都能迎刃而解。

本书挑选的网络包来自真实场景，经典且接地气。讲解时采用了生活化的 [更多>>](#)

[下载PDF样章](#) [配套文件下载](#) [分享：](#)

◎ 5.6K 浏览 57 想读 7 推荐

纸质 ¥45.00-¥31.50 (7 折) 电子 ¥25.00 电子 + 纸质 ¥45.00 购买

(纸质) + (纸质) 总价：75.60 一起购买

[目录](#) [评论 9](#) [勘误 1](#) [出版信息](#)

作者简介 专业书评 内容提要

本书作译者

LinPeiman 上海 1.0K 经验值

[发私信](#) [送积分](#) [关注](#)

《Wireshark网络分析就这么简单》即《Wireshark网络分析的艺术》作者

兑换样书

[立即兑换](#) [如何赚取积分](#)

电子书版本

PDF Epub Mobi

精彩推荐

Nmap渗透测试指南
作者：商广明

社区里还可以做什么？

提交勘误

您可以在图书页面下方提交勘误，每条勘误被确认后可以获得100积分。热心勘误的读者还有机会参与书稿的审校和翻译工作。

写作

社区提供基于Markdown的写作环境，喜欢写作的您可以在此一试身手，在社区里分享您的技术心得和读书体会，更可以体验自出版的乐趣，轻松实现出版的梦想。

如果成为社区认证作译者，还可以享受异步社区提供的作者专享特色服务。

会议活动早知道

您可以掌握IT圈的技术会议资讯，更有机会免费获赠大会门票。

[加入异步](#)

扫描任意二维码都能找到我们：



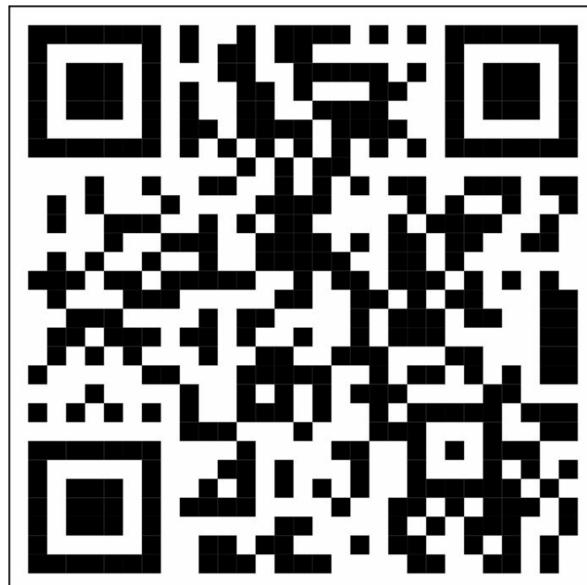
异步社区



微信订阅号



微信服务号



官方微博



QQ群: 436746675

社区网址: www.epubit.com.cn

官方微信: 异步社区

官方微博: @人邮异步社区, @人民邮电出版社-信息技术分社

投稿&咨询: contact@epubit.com.cn

Table of Contents

[版权信息](#)

[版权声明](#)

[内容提要](#)

[前言](#)

[第1章 机器学习综述](#)

[1.1 机器学习的分类](#)

[1.2 机器学习的实际案例](#)

[1.2.1 预测回头客挑战赛](#)

[1.2.2 Netflix公司](#)

[1.2.3 算法交易挑战赛](#)

[1.2.4 Heritage健康奖](#)

[1.3 机器学习的过程](#)

[1.4 机器学习背后的数学](#)

[1.5 成为一名数据科学家](#)

[1.6 统计算的R工程](#)

[1.7 RStudio](#)

[1.8 使用R包](#)

[1.9 数据集](#)

[1.10 在生产中使用R](#)

[1.11 小结](#)

[第2章 连接数据](#)

[2.1 管理你的工作目录](#)

[2.2 数据文件的种类](#)

[2.3 数据的来源](#)

[2.4 从网络中下载数据集](#)

[2.5 读取CSV文件](#)

[2.6 读取Excel文件](#)

[2.7 使用文件连接](#)

[2.8 读取JSON文件](#)

[2.9 从网站中抓取数据](#)

[2.10 SQL数据库](#)

[2.11 R中的SQL等价表述](#)

[2.12 读取Twitter数据](#)

[2.13 从谷歌分析中读取数据](#)

[2.14 写数据](#)

[2.15 小结](#)

[第3章 数据处理](#)

[3.1 特征工程](#)

[3.2 数据管道](#)

[3.3 数据采样](#)

[3.4 修正变量名](#)

[3.5 创建新变量](#)

[3.6 数值离散化](#)

[3.7 日期处理](#)

[3.8 将类变量二值化](#)

[3.9 合并数据集](#)

[3.10 排列数据集](#)

[3.11 重塑数据集](#)

[3.12 使用dplyr进行数据操作](#)

[3.13 处理缺失数据](#)

[3.14 特征缩放](#)

[3.15 降维](#)

[3.16 小结](#)

[第4章 探索性数据分析](#)

[4.1 数据统计](#)

[4.2 探索性可视化](#)

[4.3 直方图](#)

[4.4 箱形图](#)

[4.5 条形图](#)

[4.6 密度图](#)

[4.7 散点图](#)

[4.8 QQ图](#)

[4.9 热图](#)

[4.10 缺失值的图表](#)

[4.11 解释性图表](#)

[4.12 小结](#)

[第5章 回归](#)

[5.1 一元线性回归](#)

[5.2 多元线性回归](#)

[5.3 多项式回归](#)

[5.4 小结](#)

[第6章 分类](#)

[6.1 一个简单的例子](#)

[6.2 逻辑回归](#)

[6.3 分类树](#)

[6.4 朴素贝叶斯](#)

[6.5 K-最近邻](#)

[6.6 支持向量机](#)

[6.7 神经网络](#)

[6.8 集成](#)

[6.9 随机森林](#)

[6.10 梯度提升机](#)

[6.11 小结](#)

[第7章 评估模型性能](#)

[7.1 过拟合](#)

[7.2 偏差和方差](#)

[7.3 干扰因子](#)

[7.4 数据泄漏](#)

[7.5 测定回归性能](#)

[7.6 测定分类性能](#)

[7.7 交叉验证](#)

[7.8 其他机器学习诊断法](#)

[7.8.1 获取更多的训练观测数据](#)

[7.8.2 特征降维](#)

[7.8.3 添加新特征](#)

[7.8.4 添加多项式特征](#)

[7.8.5 对正则化参数进行微调](#)

[7.9 小结](#)

[第8章 非监督学习](#)

[8.1 聚类](#)

[8.2 模拟聚类](#)

[8.3 分级聚类](#)

[8.4 K-均值聚类](#)

[8.5 主成分分析](#)

[8.6 小结](#)

术语表

欢迎来到异步社区！