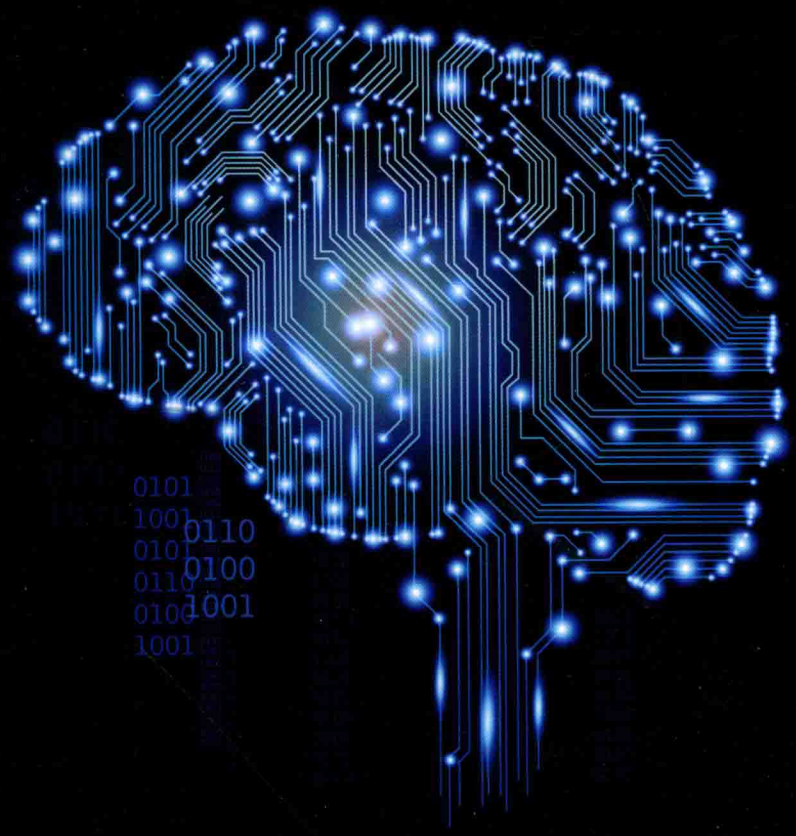


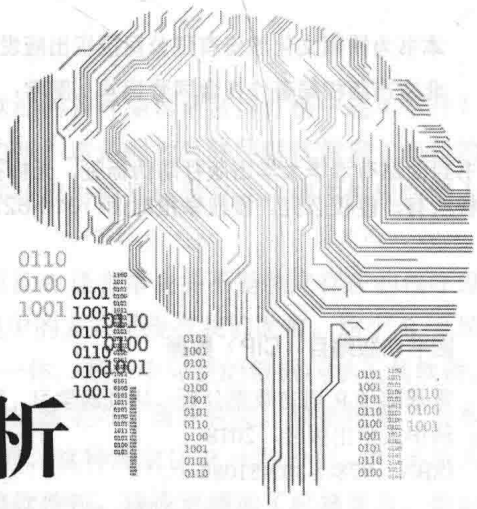
Data Analysis with R for Beginners

零基础学R语言数据分析

从机器学习、数据挖掘、文本挖掘到大数据分析

李仁钟　李秋缘　编著

清華大學出版社



零基础学
R语言数据分析

从机器学习、数据挖掘、文本挖掘到大数据分析

李仁钟　李秋缘　编著

清华大学出版社
北京

内 容 简 介

本书共分 14 章，内容主要有 R 语言简介、数据读取与写入的方法，条件判断、循环等流程控制以及自定义函数，高级绘图、低级绘图、交互式绘图的说明，决策树、支持向量机、人工神经网络的介绍，基本统计、机器学习、数据挖掘、文本挖掘、大数据分析的应用，层次聚类法、K 平均聚类算法、模糊 C 平均聚类算法、聚类指标、基因算法及人工蜂群算法的应用。

本书适合没有程序设计经验、想要接触 R 语言的人以及对统计、机器学习、数据挖掘、文本挖掘、大数据分析有兴趣的人阅读。

本书为博硕文化股份有限公司授权出版发行的中文简体字版本

北京市版权局著作权合同登记号　图字：01-2018-1990

图书在版编目（CIP）数据

零基础学 R 语言数据分析：从机器学习、数据挖掘、文本挖掘到大数据分析 / 李仁钟，李秋缘编著.—北京：清华大学出版社，2018
ISBN　978-7-302-51080-2

I. ①零… II. ①李… ②李… III. ①程序语言－程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字（2018）第 195645 号

责任编辑：夏毓彦
封面设计：王　翔
责任校对：闫秀华
责任印制：李红英

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市铭诚印务有限公司
经　　销：全国新华书店
开　　本：190mm×260mm　　**印　张：**17.75　　**字　数：**454 千字
版　　次：2018 年 10 月第 1 版　　**印　次：**2018 年 10 月第 1 次印刷
定　　价：59.00 元

产品编号：078581-01

序

有人可能会问：近几年来，为什么 R 语言在较受欢迎的各种编程语言排行榜中逐年攀升？源于统计领域、广泛使用的 R 程序设计语言具有什么魔力吗？这是因为互联网时代带来了海量的数据，而 R 是较受欢迎的数据科学语言。如果你的工作和研究与数据科学有关，或者你现在想入门数据科学，那么推荐你学习 R 语言。

R 语言与生俱来就拥有数据统计和分析的 DNA，而且 R 语言本身并不是独立存在的程序设计语言。更准确地说，R 语言以集成在一个 R 系统或环境中的方式呈现在我们面前，这个 R 系统集数据计算、数据处理、统计分析和图形绘制等软件包于一体，是一个完整的数据科学工具软件。

如今，以互联网大数据分析为基础的人工智能，如机器学习、商业智能、数据挖掘、文本挖掘、数据可视化等领域都渴求强大、高效的数据科学工具，这种渴求让 R 大放异彩。R 系统本身就是一个开放的系统，除了传统的数据统计分析/绘图等软件包，现在更增加了机器学习、数据和文本挖掘、大数据分析等相关的诸多程序包，让 R 语言在这些领域成为“光彩夺目”的明星。

如果你对上述热门的领域之一感兴趣，并且想将 R 引入你的工作或研究中，那么本书就是一本快速参考指南。本书也可以作为完全不懂 R 软件及数据分析的读者自学 R 语言的第一本读物。本书各章提供了丰富的范例程序，因而也可以作为大专院校 R 语言的上机实践课教材。

资深架构师　赵军

2018 年 6 月

前　言

随着 R 软件的流行及普及化，许多学者和专家转而使用 R 作为研究与开发的工具。R 软件有 Windows、UNIX、Linux 及 Apple MacOS 等不同操作系统的免费版本，更有一万种以上免费程序包可供使用，所以学习 R 软件是睿智的选择。

本书内容共有 14 章，前 4 章先介绍 R 软件的基本操作和应用，第 5 章对本书所使用的程序包做完整的介绍，包含 R 软件在机器学习（Machine Learning）、数据挖掘（Data Mining）、文本挖掘（Text Mining）及大数据（Big Data）分析的相关程序包，第 6~9 章介绍各类学习算法，第 10~12 章介绍关联规则、网络社群分析及文本挖掘、图形化数据分析工具，最后两章介绍 Hadoop 和 Spark 大数据分析。

本书是作者多年来从事教学的心血结晶，适合作为大专院校信息类相关科系的教材，同时书中范例的程序代码丰富，也可作为练习的补充教材。本书的撰写以完全不懂 R 软件及数据分析的读者为对象，对于有意愿自学的读者而言，本书也是一本不错的入门参考书。

本书配套范例程序可从下面的网址（注意区分数字和字母大小写）下载或扫描右边的二维码获取：

https://pan.baidu.com/s/17b-xnYfhICguW4wSz8pWXA

如果下载有问题，请联系 booksaga@126.com，邮件主题为“零基础学 R 语言数据分析：从机器学习、数据挖掘、文本挖掘到大数据分析”。

本书的撰写虽已力求完美，但难免会有疏漏之处，欢迎各位读者指教。

李仁钟、李秋缘

2018 年 6 月

目 录

第 1 章　R 简介

R 是统计软件，也是一种程序设计语言。R 当初是由 Ross Ihaka 与 Robert Gentleman 开发的，类似于 AT & T 贝尔实验室 Rick Becker、John Chambers 与 Allan Wilks 等人所开发的 S 语言。R 有 Windows、UNIX、Linux 及 Apple MacOS 等不同操作系统的版本。R 目前开发的核心团队是由世界各地不同机构所组成的，其网站位于 https://www.r-project.org，在此网站上可参阅许多有关 R 的文件、书籍及信息。R 软件的应用领域包含统计分析、数据挖掘、机器学习、推荐系统、文本挖掘及大数据分析等。

本章重点内容：

- 开始使用 R 软件
- R 对象

1.1　开始使用 R 软件

R 软件的最新版本可到网站 http://www.r-project.org 中下载，单击网页左边下载区 Download 下的 CRAN（Comprehensive R Archive Network），再从 CRAN 中选择 CRAN Mirrors 的镜像（Mirror）网站，从中下载适合用户操作系统的最新版本。用户安装 R 软件后，也可到 http://www.rstudio.com/ 下载 RStudio。RStudio 是一个为 R 设计的集成操作软件。R 及 RStudio 在 Windows 操作系统中的安装步骤可参考附录 A 和附录 B。

R 网站中提供了功能非常强大的工具集，用户可以从 CRAN 上安装相关程序包（Package），R 提供了一万个以上免费的程序包。当用户的计算机连接到网络后，若使用 Windows 版本，则可以很容易地通过“程序包”菜单来安装程序包。用户可从该菜单中选择“加载程序包”选项来选择可用的程序包。当用户选择想要的程序包后，R 软件将下载所选择的程序包并自动进行安装。在本书中除了第 13、14 章外，主要的范例和操作都在 Windows 操作系统下进行，如果用户在 UNIX、Linux 或者 Apple MacOS 上执行 R 软件，可能需要进行微调。用户也可以自行安装程序包，例如安装 C5.0 决策树程序包“C50”（注意英文字母大小写的意义是不同的），只需要在 R 提示符“>”后输入以下指令即可（注意：当提示符为“+”时，表示程序正在执行中，或者正在等待未完成的指令）。

```
> install.packages("C50")
```

用户可使用以下指令来使用 C50 程序包中提供的函数：

```
> library(C50)
```

若是要删除已安装的程序包，例如 C50 程序包，用户可以使用下面的指令：

```
> remove.packages("C50")
```

有些程序包无法在 CRAN “程序包”下的“加载程序包”选项中找到，此时用户需先到适当网站下载程序包的 ZIP 文件，再使用“程序包”下的“Install Package(s) from local files…”选项来安装。例如，用户要使用 ANN 程序包，需先到 http://cran.r-project.org/src/contrib/Archive/ANN/ 下载 ANN_0.1.4.zip，再选择“Install Package(s) from local files…”来安装，安装后可使用以下指令来调用 ANN 程序包中提供的函数：

```
> library(ANN)
```

如果用户想要知道计算机中已经安装了哪些程序包，可以输入：

```
> installed.packages()
```

其输出结果包含程序包版本信息等较复杂的信息。为了方便用户使用，可以使用 library() 来查看已安装程序包的简易信息：

```
> library()
```

此外，用户可以使用下面的指令来更新所有已安装的程序包：

```
> update.packages()
```

R 软件是一种语法非常简单的表达式语言（Expression Language）。R 语言通过对象（Object）来运行，这些对象使用它们的名称（Name）和值（Content，或内容）来描述其特性。对象名称（变量）的第一个字母必须为英文字母或句点“.”，若以句点当对象的第一个字母，则其后接的第一个字符不能为数字，例如 .2iswrong 是不能作为对象名称的。对象不需要事先声明，但其名字的英文字母大小写代表不同的对象，因此 X 和 x 是不同的对象。R 语言保留了一些用于指令名称的保留字，如 c 与 NA 等。R 语言可使用赋值（Assignment）表达式“<-”来给对象赋值（也可以使用“=”号），例如：

```
> x <- 10
> x
[1] 10

> X <- x^2
> X
[1] 100
> z <- sqrt(X)
> z
[1] 10
```

其次，也可以通过对象的数据种类（属性，Attribute）来描述对象的特性，也就是说，作用于一个对象的函数取决于其对象的属性。所有对象都有两个内在属性：数据类型（Mode）和长度（Length）。对象中的元素（Element）共有 4 种基本数据类型：数值（Numeric）、字符（Character）、

复数（Complex）和逻辑（Logical），虽然也存在其他数据类型，但是并不能用来表示数据，例如函数 （Function）、表达式（Expression）；长度（Length）属性用于表示对象中元素的数量。对象的数据类型和长度可以分别通过函数 mode() 和 length() 来获取。

```
> x <- 10
> x [1] 10

> mode(x)
[1] "numeric"

> length(x)
[1] 1
```

如果要在同一行内执行多条指令，那么可以用分号“;”隔开R指令，例如：

```
> x <- 10; y <- x^2; z <- sqrt(y)
> z
[1] 10
```

注释可以放在程序中的任何地方，从“#”号开始到句子结束之间的语句就是注释，例如：

```
> x <- 10                #整数型数值
> x
[1] 10
> mode(x)
[1] "numeric"
> length(x)
[1] 1
> y <- 10.9              #实数型数值
> y
[1] 10.9
> mode(y)
[1] "numeric"
> length(y)
[1] 1
> z <- T                 #逻辑
> z
[1] TRUE
> mode(z)
[1] "logical"
> length(z)
[1] 1
> a <- "Hello"           #文本
> a
[1] "Hello"
> mode(a)
[1] "character"
> length(a)
[1] 1
> z <- 4+2i              #复数
> z
```

```
[1] 4+2i
> mode(z)
[1] "complex"
> length(z)
[1] 1
```

1.2 R 对象

R 是以面向对象为主的程序设计语言，其常用对象可以是向量（Vector）、数组（Array）、矩阵（Matrix）、因子（Factor）、数据框（Data Frame）及列表（List）等。

1.2.1 向量

向量是由包含相同数据类型的元素组成的，R 程序中最简单的结构就是由一串有序数值构成的数值（Numeric）向量。假如用户要创建一个含有 6 个数值的向量 V，且其值分别为 10、5、3.1、6.4、9.2 和 21.7，则 R 程序中的指令为 c() 函数。

```
> V <- c(10, 5, 3.1, 6.4, 9.2, 21.7)
> V
[1] 10.0 5.0 3.1 6.4 9.2 21.7
> length(V)
[1] 6
> mode(V)
[1] "numeric"
```

也可以使用 assign() 函数来实现相同的功能：

```
> assign("V", c(10, 5, 3.1, 6.4, 9.2, 21.7))
> V
[1] 10.0 5.0 3.1 6.4 9.2 21.7
> length(V)
[1] 6
> mode(V)
[1] "numeric"
```

在某些情况下，向量的元素可能会遗失。当向量中的元素为缺失值（Missing Value）时，其相关位置可给予一个特定的值 NA（需为大写）。

```
> V <- c(10, 5, NA, 6.4, 9.2, 21.7)
> V
[1] 10.0 5.0 NA 6.4 9.2 21.7
```

用户可以使用中括号“[]”来存取向量中的特定元素。值得注意的是，R 程序向量对象默认的第一个元素的序号（Index，也称为下标或者索引）是 1 而不是 0。

```
> V[2]
[1] 5
```

R 还提供了 Inf、–Inf 及 NaN（Not a Number），而 NULL 是指对象的长度是 0。

```
>  V <- c(1,-2,0)
>  V/0
[1]  Inf -Inf  NaN

> V <- NULL
> length(V)
[1] 0
```

用户也可使用冒号“:”创建向量。

```
> V2=1:10
> V2
[1]  1  2  3  4  5  6  7  8  9 10

> V2[1]
[1] 1

> V2[2:4]
[1] 2 3 4
```

1.2.2 数组

数组可以看作是多维的向量。例如，一个 3 维的数组 X 可以用 X[i,j,k]来指向特定元素。假设数组 X 的维度向量是 c(3,4,2)，则 X 中有 3 × 4 × 2=24 个元素，依次为 X[1,1,1],X[2,1,1], ... ,X[2,4,2],X[3,4,2]。

假设 X 是一个包含 24 个元素的向量：

```
> X <- 1:24
```

用户可以使用 dim()函数指定其数组维数（Dimension），让 X 变成一个 3×4×2 的 3 维数组，而 R 程序会按照行的方式排列：

```
> dim(X) <- c(3,4,2)
> X
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

用户可以让 X 变成一个 4×6 的 2 维数组：

```
> dim(X) <- c(4,6)
> X

     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    5    9   13   17   21
[2,]    2    6   10   14   18   22
[3,]    3    7   11   15   19   23
[4,]    4    8   12   16   20   24
```

要创建一个数组，也可以直接调用 array() 函数来创建，此函数第一个参数指定数据向量，第二个参数指定数组维数。假设要创建一个 3×4×2 的 3 维数组：

```
> X <- array(1:24, dim = c(3,4,2))
> X
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

假设要创建一个 4×6 的 2 维数组：

```
> X <- array(1:24, dim = c(4,6))
> X
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    5    9   13   17   21
[2,]    2    6   10   14   18   22
[3,]    3    7   11   15   19   23
[4,]    4    8   12   16   20   24
```

值得注意的是，以下指令会创建一个所有元素都是 0 的 3 维数组：

```
> X <- array(0, dim = c(3,4,2))
> X
, , 1

     [,1] [,2] [,3] [,4]
[1,]    0    0    0    0
[2,]    0    0    0    0
[3,]    0    0    0    0

, , 2
```

```
     [,1] [,2] [,3] [,4]
[1,]    0    0    0    0
[2,]    0    0    0    0
[3,]    0    0    0    0
```

也可以使用 rbind() 和 cbind() 函数来创建数组。rbind() 表示按向量行（Row）合并成一个数组，而 cbind() 是使用列的方式合并：

```
> X1 <- c(1,2,3,4)
> X2 <- c(5,6,7,8)
> X  <- rbind(X1,X2)
> X
   [,1] [,2] [,3] [,4]
X1    1    2    3    4
X2    5    6    7    8
> X <- cbind(X1,X2)
> X
     X1 X2
[1,]  1  5
[2,]  2  6
[3,]  3  7
[4,]  4  8
```

1.2.3 矩阵

矩阵（Matrix）就是一个 2 维数组，要创建一个矩阵，可以使用函数 matrix() ：

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE,
dimnames = NULL)
```

其中 ：

byrow 表示矩阵数据是按行还是按列（byrow = FALSE）的顺序排列。

nrow表示矩阵的行数。

ncol 表示矩阵的列数。

dimnames 表示可以帮行列命名。

```
> X <- matrix(1:24, nrow=4, ncol=6, byrow=TRUE)
> X
    [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    2    3    4    5    6
[2,]    7    8    9   10   11   12
[3,]   13   14   15   16   17   18
[4,]   19   20   21   22   23   24

> X <- matrix(1:24, nrow=4, ncol=6, byrow=FALSE)
> X

    [,1] [,2] [,3] [,4] [,5] [,6]
```

```
[1,]    1    5    9   13   17   21
[2,]    2    6   10   14   18   22
[3,]    3    7   11   15   19   23
[4,]    4    8   12   16   20   24
```

也可以使用 rbind() 和 cbind() 函数来创建矩阵。t()是矩阵的转置（Transposition）函数，nrow()和 ncol() 函数分别返回矩阵的行数和列数。

```
> X1 <- c(1,2,3)
> X2 <- c(4,5,6)
> X3 <- c(7,8,9)
> X <- cbind(X1,X2,X3)
> X
     X1 X2 X3
[1,]  1  4  7
[2,]  2  5  8
[3,]  3  6  9
> Y=t(X)
> Y
   [,1] [,2] [,3]
X1    1    2    3
X2    4    5    6
X3    7    8    9

> m <- nrow(Y)
> m
[1] 3
> n <- ncol(Y)
> n
[1] 3
```

若要显示矩阵 X 第一行元素，则可使用：

```
> X[,1]
[1] 1 2 3
```

若要显示矩阵 X 第二列元素，则可使用：

```
> X[2,]
X1 X2 X3
 2  5  8
```

若要显示矩阵 X 第一列和第三列元素，则可使用：

```
> X[c(1,3),]
     X1 X2 X3
[1,]  1  4  7
[2,]  3  6  9
```

若要删除矩阵 X 第一行元素，则可使用：

```
> X[,-1]
     X2 X3
[1,]  4  7
[2,]  5  8
```