資料科學與 R 語言

曾意儒 *Yi-Ju Tseng*

2017-02-07

Contents

長庚大學資訊管理學系 大數據分析方法教學使用書籍，內容包括使用R 語言做資料擷取、資料清洗與處理、探索式資料分析、資料視覺化與資料探勘等，並介紹 R 與 Hadoop EcoSystems 介接方法。

目前完成 1~5 與第 8 章，PDF 版本與 epub 版本格式微調中。

Chapter 1

R 語言 101

本章節介紹學習 R 語言的基本知識，包括基本指令操作、運算子介紹等。

1.1 什麼是 R 語言

R 語言是一種自由軟體程式語言，主要用於資料分析與統計運算，2000 年時終於發表 R 1.0.0，有關 R 語言的發展歷史可參考維基百科。基本的 R 軟體已經內建多種統計及分析功能，其餘功能可以透過安裝**套件 (Packages)** 加載，眾多的套件使 R 的使用者可以【站在巨人的肩膀上 (Standing on the shoulders of giants (Hal R. Varian, Google))】做資料分析，截至 2017 年 1 月為止，R 軟體可另外安裝的套件數目共有 10,000 個以上 (R Studio 報導)。常用的套件清單可參考各項網路資訊，如R Studio 的整理 : Quick list of useful R packages

安裝套件 Package 的方法如下：

```
install.packages(" 套件名稱")
```

值得注意的是，套件名稱需要加上雙引號，舉例來說，若要安裝 `ggplot2` 套件，則要在 R 的 Console 視窗內輸入：

```
install.packages("ggplot2")
```

若要載入**已安裝**的套件，則輸入 `library(套件名稱)`，範例：

```
library(ggplot2)
```

載入已安裝的套件時，**不用**在套件名稱前後加雙引號。

1.2 函數使用

在 R 中有許多內建函數，安裝套件後各套件也會提供各式各樣寫好的函數，函數使用方式為函數名稱 `(參數 1, 參數 2,....)`，以計算平均數為例，可使用 `mean()` 函數，範例如下:

```
mean(c(1,2,3,4,5,6)) ## 計算 1~6 的平均數
```

```
## [1] 3.5
```

若想知道各函數所需參數，可使用`?` 函數名稱觀看函數作者所撰寫的說明文件

```
?mean
```

除非有指定參數名稱，函數的參數設定有順序性，如序列產生函數 `seq()`，參數順序為 `from, to, by`，代表序列起點、序列終點，以及相隔單位。

```
seq(from=1,to=9,by=2)#1~9，每隔 2 產生一數字
```

```
## [1] 1 3 5 7 9
```

```
seq(1,9,2)# 按照順序輸入參數，可省去參數名稱
```

```
## [1] 1 3 5 7 9
```

```
seq(by=2,to=9,from=1)# 若不想照順序輸入參數，需要指定參數名稱
```

```
## [1] 1 3 5 7 9
```

1.3 變數設定

在開始深入學習 R 語言之前，首要任務是學習最基本的 R 程式碼：**變數設定**，在 R 語言中，主要使用 <-設定變數，設定方法為：**變數名稱 <-變數內容 (值)**，雖然**變數名稱**可依箭頭方向放置於左側 <-或右側->，但為方便閱讀，**變數名稱**多放置於左側。

```
a<-1
2->b
a
```

```
## [1] 1
```

```
b
```

```
## [1] 2
```

R 語言也接受使用 = 設定變數，此時**變數名稱**必須在左側，如：**變數名稱 = 變數內容**

```
c=1
c
```

```
## [1] 1
```

除了**變數設定**外，str() 函數也為常用基本函數，str() 用在檢查與總覽各類變數型態。

```
d<-3
str(d)
```

```
##  num 3
```

變數的命名有以下規則:

- 不可使用保留字，如 break, else, FALSE, for, function, if, Inf, NA, NaN, next, repeat, return, TRUE, while 等
- 開頭只能是英文字，或 .
- 大小寫敏感

1.4 執行視窗

R 是可直譯的語言，也就是說，可以在執行視窗 (Console) 直接打程式碼，在視窗出現 > 時，表示可輸入指令，若視窗出現 + 時，表示前面的程式碼還沒打完，必須鍵入完整的程式碼讓 R 執行。

1.5 資料型態

在 R 語言中，常用的資料型態包括**數值 (numeric)**、**字串 (character)**、**布林變數 (logic)** 以及**日期 (Date)** 等。

1.5.1 數值 numeric

數值包括整數（沒有小數點）與符點數（有小數點）的數值

```
num1<-100
num2<-1000.001
```

值得注意的是，若數值長度超過 2^53，必須導入 bit64 package (?)，將數值長度上限提高為 2^63，才能表示完整數值

```
print(2^53, digits=20)
```

```
## [1] 9007199254740992
```

```
print(2^53+1, digits=20) # +1 後，數值仍與 2^53 相同
```

```
## [1] 9007199254740992
```

```
library(bit64) # 導入 bit64 package
print(as.integer64(2)^53, digits=20)
```

```
## integer64
## [1] 9007199254740992
```

```
print(as.integer64(2)^53+1, digits=20)# 導入 bit64 後，可得正確答案
```

```
## integer64
## [1] 9007199254740993
```

1.5.2 字串 character

用雙引號" 框起的文字會被儲存為字串格式，若在數字前後加上雙引號，數字也會被儲存為文字形式，無法進行數值的加減乘除等運算。

```
char1<-"abcTest"
char2<-"100"
char3<-"200"
#char2+char3 # 會輸出 Error message: non-numeric argument to binary operator
```

1.5.3 布林變數 logic

用於邏輯判斷，可使用大寫 **TRUE** 或 **T** 代表**真**，大寫 **FALSE** 或 **F** 代表假。

```
boolT<-TRUE
boolT1<-T
boolF<-FALSE
boolF1<-F
```

1.5.4 日期 (Date)

用於表示日期，於資料分析中常用，使用 Sys.Date() 指定可得系統日期。

```
dateBook<-Sys.Date()
dateBook
```

```
## [1] "2017-02-07"
```

日期與字串的相關轉換操作可考慮使用簡單易懂的 `lubridate(?)` package，如果想要將年/月/日格式的文字轉換為日期物件，可使用 `ymd()` 函數 (y 表年 year，m 表月 month，d 表日 day)，如果想要將月/日/年格式的文字轉換為日期物件，則使用 `mdy()` 函數，以此類推。

```
library(lubridate)
ymd('2012/3/3')
```

```
## [1] "2012-03-03"
```

```
mdy('3/3/2012')
```

```
## [1] "2012-03-03"
```

其他使用方式可參考 The Yhat Blog。

1.6 基本運算子

1.6.1 數學基本運算

在 R 中，數學運算與其他程式語言相同

- 加 +
- 減 -
- 乘 *
- 除 /
- 餘數 %%
- 次方 ^

```
num1<-1
num2<-100
num1+num2
```

```
## [1] 101
```

```
num1-num2
```

```
## [1] -99
```

```
num1*num2
```

```
## [1] 100
```

```
num1/num2
```

```
## [1] 0.01
```

```
100%%3 ##100 除以 3 後所得餘數
```

```
## [1] 1
```

```
2^3 ##2 的 3 次方
```

```
## [1] 8
```

1.6.2 進階數學函數

- 四捨五入 `round()`
- 無條件捨去 `floor()`
- 無條件進位 `ceiling()`

```
num1<-1.568
num2<-2.121
round(num1,digits = 2) # 四捨五入至小數點第二位
```

```
## [1] 1.57
```

```
round(num2,digits = 1) # 四捨五入至小數點第一位
```

```
## [1] 2.1
```

```
floor(num1) ##1.568
```

```
## [1] 1
```

```
ceiling(num2) ##2.121
```

```
## [1] 3
```

1.6.3 邏輯運算

常用之邏輯判斷也可在 R 中直接使用

- 大於 >
- 小於 <
- 等於 ==，為了不與變數設定混淆，判斷兩變數是否相等，要用**雙等號**
- 大於等於 >=
- 小於等於 <=

```
num1<-1
num2<-100
num1>num2
```

```
## [1] FALSE
```

```
num1<num2
```

```
## [1] TRUE
```

文字字串也可比較大小

```
char1<-"abcTest"
char2<-"defTest"
char1>char2
```

```
## [1] FALSE
```

邏輯混合判斷，和 JAVA 等語言不同的是，在 R 中使用**單符號**即可表示且 & 和或 |

- 且 &
- 或 |

```
TRUE & TRUE
```

```
## [1] TRUE
```

```
TRUE & FALSE
```

```
## [1] FALSE
```

```
TRUE | TRUE
```

```
## [1] TRUE
```

```
TRUE | FALSE
```

```
## [1] TRUE
```

反向布林變數!

```
!TRUE
```

```
## [1] FALSE
```

```
!FALSE
```

```
## [1] TRUE
```

1.7 錯誤訊息

- Message : 有可能的錯誤通知，程式會繼續執行
- Warning : 有錯誤，但是不會影響太多，程式會繼續執行
- Error : 有錯，而且無法繼續執行程式
- Condition : 可能會發生的情況

```
log(-1)
```

```
## Warning in log(-1): 產生了 NaNs
```

```
## [1] NaN
```

```
mena(NA)
```

```
## Error in eval(expr, envir, enclos): 沒有這個函數 "mena"
```

錯誤訊息範例 1:

```
# Error: could not find function "fetch_NBAPlayerStatistics"
# 找不到"fetch_NBAPlayerStatistics" function
```

可能原因 : 沒安裝或沒讀入 SportsAnalytics package

錯誤訊息範例 2:

```
# Error in library(knitr): there is no package called 'knitr'
# 找不到"knitr" package
```

可能原因 : 沒安裝 knitr package

1.8 Help

R 語言與套件均有完整的文件與範例可以參考，在 R 的執行視窗中，輸入? 函數名稱或? 套件名稱即可看到函數或套件的使用說明

```
?ggplot2
?ymd
```

除此之外，Stack Overflow中也有許多問答，可直接在網站中搜尋關鍵字與錯誤訊息。

Chapter 2

R 資料結構

2.1 向量 vector

向量為一維資料的表現和儲存方式，用 `c()` 函數可定義向量，如：

```
vec<-c('a','b','c','d','e')
```

a~e 為 vec 向量中的**元素 (element)**，各元素向中的順序固定，a 為 vec 向量中的第 **1** 個元素，b 則為第 **2** 個元素，以此類推，若要將 vec 向量的第 **4** 個元素取出，可使用

```
vec[4] ## 第 4 個元素
```

```
## [1] "d"
```

也可同時取出多個元素

```
vec[c(2,3)] ## 第 2 與第 3 個元素
```

```
## [1] "b" "c"
```

此外，在同一向量中，所有元素之**資料型態必須相同**，如上述 vec 向量，元素均為文字型態。

和變數指定類似，向量中的元素也可以使用 <-重新指定

```
vec[3]
```

```
## [1] "c"
```

```
vec[3]<-'z' ## 第三個元素值設定為 "z"
vec[3]
```

```
## [1] "z"
```

2.1.1 快速產生向量函數

若要產生連續向量，如 1~20，可使用: 來串連首字與最後一字

```
1:20 ## c(1,2,...,19,20)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

或是使用 seq() 函數

```
seq(from=1,to=20,by=1) ##1~20，中間相隔 1
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```
seq(from=1,to=50,by=2) ##1~50，中間相隔 2
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
```

2.1.2 向量運算

向量也可直接做加減乘除運算，如

```
numvec<-1:10 ## c(1,2,3,4,5,6,7,8,9,10)
numvec+3 ## 所有元素 +3
```

```
##  [1]  4  5  6  7  8  9 10 11 12 13
```

```
numvec*2 ## 所有元素 *2
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

向量和向量也可做運算，如

```
numvec1<-1:3 ## c(1,2,3)
numvec2<-4:6 ## c(4,5,6)
numvec1+numvec2
```

```
## [1] 5 7 9
```

```
numvec1*numvec2
```

```
## [1]  4 10 18
```

2.2 因子 factor

因子是由向量轉換而成，多用於表示**類別**數據，如大學中有大學生、碩士班學生與博士班學生三種類別的學生，使用方法為 factor(資料向量,levels= 類別次序)，levels 參數可設定各類別的次序

```
factor(c(" 大學生"," 碩士班學生"," 博士班學生"),
       levels = c(" 大學生"," 碩士班學生"," 博士班學生"))
```

```
## [1] 大學生     碩士班學生 博士班學生
## Levels: 大學生 碩士班學生 博士班學生
```

因子變量一但決定其類別的種類與數目時，通常不會再作更動，也就是任何新增的元素都要是大學生、碩士班學生與博士班學生其中一種。

2.3 列表 list

由於向量和因子都只能儲存一種元素，使用上彈性較不足，在 R 語言中，有一彈性很大的資料型態**列表 list**，在列表中，元素可分屬不同資料類別，除了可包括**數值**與**文字**外，也可以包括資料集，如**向量**和**因子**等，更進階的使用，還可以包括矩陣與資料框。如要建立列表，可使用 list() 函數

```
listSample<-list(Students=c("Tom","Kobe","Emma","Amy"),Year=2017,
                 Score=c(60,50,80,40),School="CGU")
listSample
```

```
## $Students
## [1] "Tom"  "Kobe" "Emma" "Amy"
##
## $Year
## [1] 2017
##
## $Score
## [1] 60 50 80 40
##
## $School
## [1] "CGU"
```

2.3.1 列表資料擷取

列表可用 $ 符號做資料擷取

```
listSample$Students ## 取得中表中的 Students 變量
```

```
## [1] "Tom"  "Kobe" "Emma" "Amy"
```

也可和向量一樣，使用索引值來擷取資料，和向量不同的是，若要取得**值**，要使用雙中括號 [[]]

```
listSample[[1]] ## 取得中表中第一個變量的值
```

```
## [1] "Tom"  "Kobe" "Emma" "Amy"
```

如果只使用單中括號，回傳的資料型態會是列表 list，並非列表中的值

```
listSample[1] ## 取得中表中第一個變量 ( 列表型態 )
```

```
## $Students
## [1] "Tom"  "Kobe" "Emma" "Amy"
```

2.3.2 列表資料編輯設定

列表資料也可和向量資料一樣，重新編輯設定

```
listSample[[1]]
```

```
## [1] "Tom"  "Kobe" "Emma" "Amy"
```

```
listSample[[1]]<-c(" 小明"," 大雄"," 胖虎"," 小新"," 大白") ## 將 Students 變量重新設定
listSample[[1]]
```

```
## [1] " 小 明 " " 大 雄 " " 胖 虎 " " 小 新 " " 大 白 "
```

除了編輯以外，列表資料也能用 $ 符號與 <-變數設定符號新增

```
listSample$Gender<-c("M","F","M","F","M") ## 新增 Gender 變量，並設定向量值
```

2.4 矩陣 matrix

```
a <- matrix(c(1:6), nrow=3, ncol=2) ## 建立 3x2 的矩陣，分別填入 1~6 的值
a
```

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

2.5 資料框 data.frame

資料框是非常常見的二維資料格式，由一系列的欄位（Column）和列（Row）所組成，常見的 Excel 試算表也是類似的資料表現形式，可使用 `data.frame()` 來創建新的資料框

```
StuDF <- data.frame(StuID=c(1,2,3,4,5), ## 欄位名稱 = 欄位值
                    name=c(" 小明"," 大雄"," 胖虎"," 小新"," 大白"),
                    score=c(80,60,90,70,50))
StuDF
```

```
##   StuID name score
## 1     1 小明    80
## 2     2 大雄    60
## 3     3 胖虎    90
## 4     4 小新    70
## 5     5 大白    50
```

如範例所示，每個欄位都有名稱（StuID, name, score），若沒有設定欄位名稱，R 會自動指派 $V1_{Vn}$ 作為欄位名稱。在 R 中，每個欄位的資料型態必須相同，如 StuID 作為列名。如需檢查欄位名稱與列名，可使用 `colnames()` 和 `rownames()`

```
colnames(StuDF) ## 欄位名稱
```

```
## [1] "StuID" "name"  "score"
```

```
rownames(StuDF) ## 列名
```

```
## [1] "1" "2" "3" "4" "5"
```

如需檢查個欄位之資料型別，可使用 `str()` 函數

```
str(StuDF)
```

```
## 'data.frame':    5 obs. of  3 variables:
##  $ StuID: num  1 2 3 4 5
##  $ name : Factor w/ 5 levels "大白","大雄",..: 3 2 5 4 1
##  $ score: num  80 60 90 70 50
```

2.6 資料表 data.table

data.table 是 data.frame 資料框型別的延伸，如要使用必須安裝 data.table (**?**) package，使用 `data.table` 讀取大型資料的速度比使用資料框快上數倍，進階處理語言也相當好用，在探索式資料分析章節 Chapter 6會詳細介紹。其他詳細教學可見 A data.table R tutorial by DataCamp，DataCamp 也提供互動式教學課程，可自行參閱。

2.7 資料屬性查詢函數

資料屬性可透過下列函數查詢:

- 名稱 `names()`
- 各維度名稱 `dimnames()`

- 長度 length()
- 各維度長度 dim()
- 資料型態 class()
- 各類資料計數 table()
- 總覽資料 str()

透過 names() 函數，可取得各種資料之名稱

```
head(islands) ##R 內建的資料
```

```
##       Africa   Antarctica         Asia    Australia Axel Heiberg       Baffin
##        11506         5500        16988         2968           16          184
```

```
head(names(islands)) ## 顯示上述資料之資料名稱
```

```
## [1] "Africa"       "Antarctica"   "Asia"         "Australia"    "Axel Heiberg"
## [6] "Baffin"
```

若為資料框，則會顯示行（欄位）名稱

```
head(USArrests) ##R 內建的資料
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

```
head(names(USArrests)) ## 顯示上述資料之資料名稱
```

```
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

透過 dimnames() 函數可顯示資料框列與行的名稱，先顯示列，再顯示行

```
dimnames(USArrests)
```

```
## [[1]]
##  [1] "Alabama"        "Alaska"         "Arizona"        "Arkansas"
##  [5] "California"     "Colorado"       "Connecticut"    "Delaware"
##  [9] "Florida"        "Georgia"        "Hawaii"         "Idaho"
## [13] "Illinois"       "Indiana"        "Iowa"           "Kansas"
## [17] "Kentucky"       "Louisiana"      "Maine"          "Maryland"
## [21] "Massachusetts"  "Michigan"       "Minnesota"      "Mississippi"
## [25] "Missouri"       "Montana"        "Nebraska"       "Nevada"
## [29] "New Hampshire"  "New Jersey"     "New Mexico"     "New York"
## [33] "North Carolina" "North Dakota"   "Ohio"           "Oklahoma"
## [37] "Oregon"         "Pennsylvania"   "Rhode Island"   "South Carolina"
## [41] "South Dakota"   "Tennessee"      "Texas"          "Utah"
## [45] "Vermont"        "Virginia"       "Washington"     "West Virginia"
## [49] "Wisconsin"      "Wyoming"
##
## [[2]]
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

透過 length() 函數可顯示資料長度，包括向量與資料框，若資料行態為資料框，則會顯示行（欄位）數

```
length(islands)
```

```
## [1] 48
```

```
length(USArrests)
```

```
## [1] 4
```

透過 dim() 函數可顯示資料框列與行的長度，與 dimnames() 相同，先顯示列，後顯示行

```
dim(USArrests)
```

```
## [1] 50  4
```

使用 class() 函數可知道變數類別

```
class(1)
```

```
## [1] "numeric"
```

```
class("Test")
```

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

```
class(Sys.Date())
```

```
## [1] "Date"
```

使用 table() 函數可知道向量中每個值出現幾次

```
iris$Species ## 原始值
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

```
table(iris$Species) ## 統計結果
```

```
##
##     setosa versicolor  virginica
##         50         50         50
```

使用 str() 函數可總覽變數資訊

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
str(listSample)
```

```
## List of 5
##  $ Students: chr [1:5] "小明" "大雄" "胖虎" "小新" ...
##  $ Year    : num 2017
##  $ Score   : num [1:4] 60 50 80 40
##  $ School  : chr "CGU"
##  $ Gender  : chr [1:5] "M" "F" "M" "F" ...
```

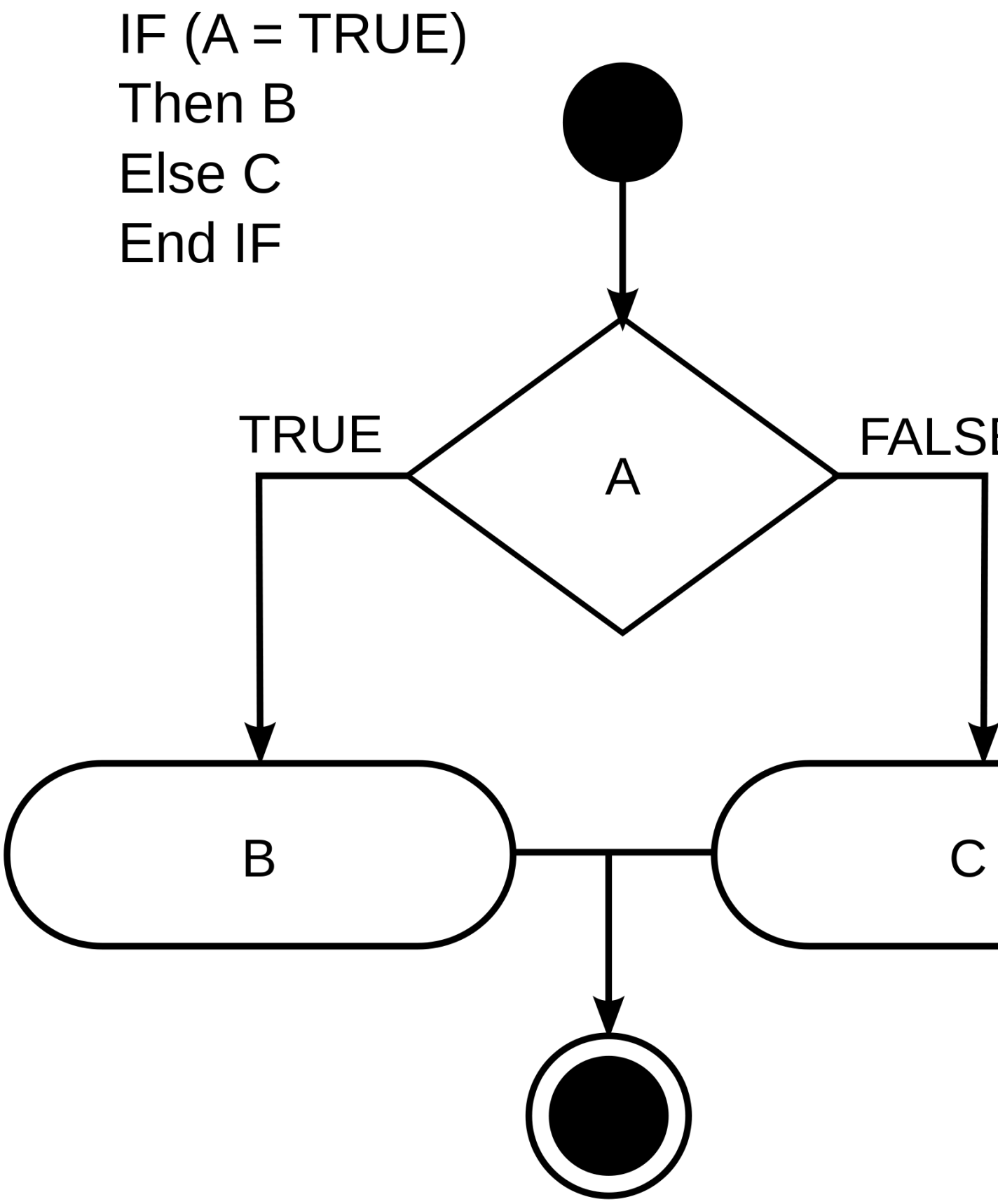
Chapter 3

控制流程

3.1 條件判斷

3.1.1 if-else 敘述

if-else 敘述使用在邏輯判斷，若需要依條件改變需要執行的程式碼，就會使用 **if-else**，若 **if** 後所接邏輯判斷為**真 (TRUE)**，就會執行 if 下方之程式碼，若為**偽 (FALSE)**，則執行 **else** 下方之程式碼，若程式中沒有 **else** 片段，則不執行任何程式碼。


IF (A = TRUE)
Then B
Else C
End IF
TRUE
A
FALSE
B
C

`if` 與 `else` 下方的程式碼必須要使用 `{}` 將程式碼包起來，若程式碼只有一行，可省略 `{}`，但為閱讀方便，建議不要省略 `{}`。

舉例來說，若考試分數**大於等於 60 分**，則印出**及格**字樣，小於 60 分則印出**不及格**字樣，程式範例如下:

```
score<-59
if(score>=60){
  print(" 及格")
}else{
  print(" 不及格")
}
```

```
## [1] "不 及 格"
```

```
score<-80
if(score>=60){
  print(" 及格")
}else{
  print(" 不及格")
}
```

```
## [1] "及 格"
```

3.1.2 if-else if-else

很多時候必須要使用多重邏輯判斷，若考試分數大於等於 90 分，印出**優良**，介於 60 到 90 分間，印出**及格**，小於 60 分則印出**不及格**，此時就會用到多重邏輯，使用多重邏輯時，會在 `if` 和 `else` 間新增邏輯區段 **else if**，程式範例如下:

```
score<-95
if(score>=90){
  print(" 優秀")
}else if(score>=60){
  print(" 及格")
}else{
  print(" 不及格")
}
```

```
## [1] "優 秀"
```

`if-else if-else` 敘述是有順序的，若在 `if` 敘述判斷為真，就算後方 `else if` 判斷也為真，也只會執行 `if` 區段的程式碼，如上述範例，95 分大於等於 90 分 (if 邏輯)，也大於等於 60 分 (else if 邏輯)，但最後只印出**優秀**字樣。

3.1.3 巢狀 if

巢狀 if 是指在 `if` 區段程式碼內包含其他 `if-else` 判斷，舉例來說，若國文分數與英文分數皆大於等於 60 分，印出**全部及格**，國文分數大於等於 60 分，英文小於 60 分，則印**國文及格，英文再加油**，以此類推，程式範例如下:

```
CHscore<-95 ## 國文成績
ENscore<-55 ## 英文成績
if(CHscore>=60){
  if(ENscore>=60){
    print(" 全部及格")
  }else{
    print(" 國文及格，英文再加油")
  }
}else{
  if(ENscore>=60){
```

```
    print(" 英文及格，國文再加油")
  }else{
    print(" 全部不及格")
  }
}
```

```
## [1] "國 文 及 格 ， 英 文 再 加 油"
```

3.1.4 ifelse()

ifelse() 函數可用最短的方式取代 if-else 敘述，使用方法為 ifelse(邏輯判斷, 判斷為真要執行的程式碼, 判斷為偽要執行的程式碼)，依上述範例，重寫程式碼如下:

```
score<-80
ifelse(score>=60," 及格"," 不及格")
```

```
## [1] "及 格"
```

值得注意的是，ifelse() 可判斷向量，也就是可一次**判斷多個元素**

```
scoreVector<-c(30,90,50,60,80)
ifelse(scoreVector>=60," 及格", " 不及格")
```

```
## [1] "不 及 格" "及 格"   "不 及 格" "及 格"   "及 格"
```

3.2 迴圈

3.2.1 for

R 語言的 for 迴圈寫法和其他語言不同，首先必須建立需要逐一執行的參數向量或序列，再使用 for 迴圈逐一執行，程式寫法為 for (單一變數 in 參數向量){ 程式碼 }，範例如下:

```
for (n in 1:10){ #n 為單一變數，1:10 為需要逐一執行的參數向量
  print(n)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

for 迴圈也可和 if-else 函數合併使用，如:

```
for (n in 1:10){
  if(n%%2==0){ # 偶數直接輸出數字
    print(n)
  }else{
    print(" 奇數") # 奇數則輸出 " 奇數 "
```

```
  }
}
```

```
## [1] "奇 數"
## [1] 2
## [1] "奇 數"
## [1] 4
## [1] "奇 數"
## [1] 6
## [1] "奇 數"
## [1] 8
## [1] "奇 數"
## [1] 10
```

3.2.2 while

`while` 函數則是在每次執行迴圈時檢查 while 邏輯判斷是否為真，若邏輯判斷為真，就會執行區段程式碼，若邏輯判斷為偽，則會結束迴圈執行。

```
x<-0
while(x<=5){
  print(x)
  x<-x+1
}
```

```
## [1] 0
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

3.2.3 break

若遇特殊情形想**結束**迴圈執行，可使用 `break` 指令

```
for(n in 1:10){
  if(n==5){
    break ## 一執行到 5，跳出迴圈，不再執行之後的迴圈
  }
  print(n)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
```

3.2.4 next

若遇特殊情形想**跳過**迴圈執行，可使用 `next` 指令

```
for(n in 1:10){
  if(n==5){
    next ## 跳過 5，直接執行下一個迴圈
  }
  print(n)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

Chapter 4

資料讀取與匯出

資料 (Data) 在維基百科的定義是 `values of qualitative or quantitative variables, belonging to a set of items.`，一般來說，在資料分析前會經過多個步驟，包括**資料匯入** Chapter 4、**資料清洗處理** Chapter 5並轉換為 Tidy data、**資料分析** Chapter 6、**資料呈現與視覺化** Chapter 7。

資料有多種可能來源，包括:

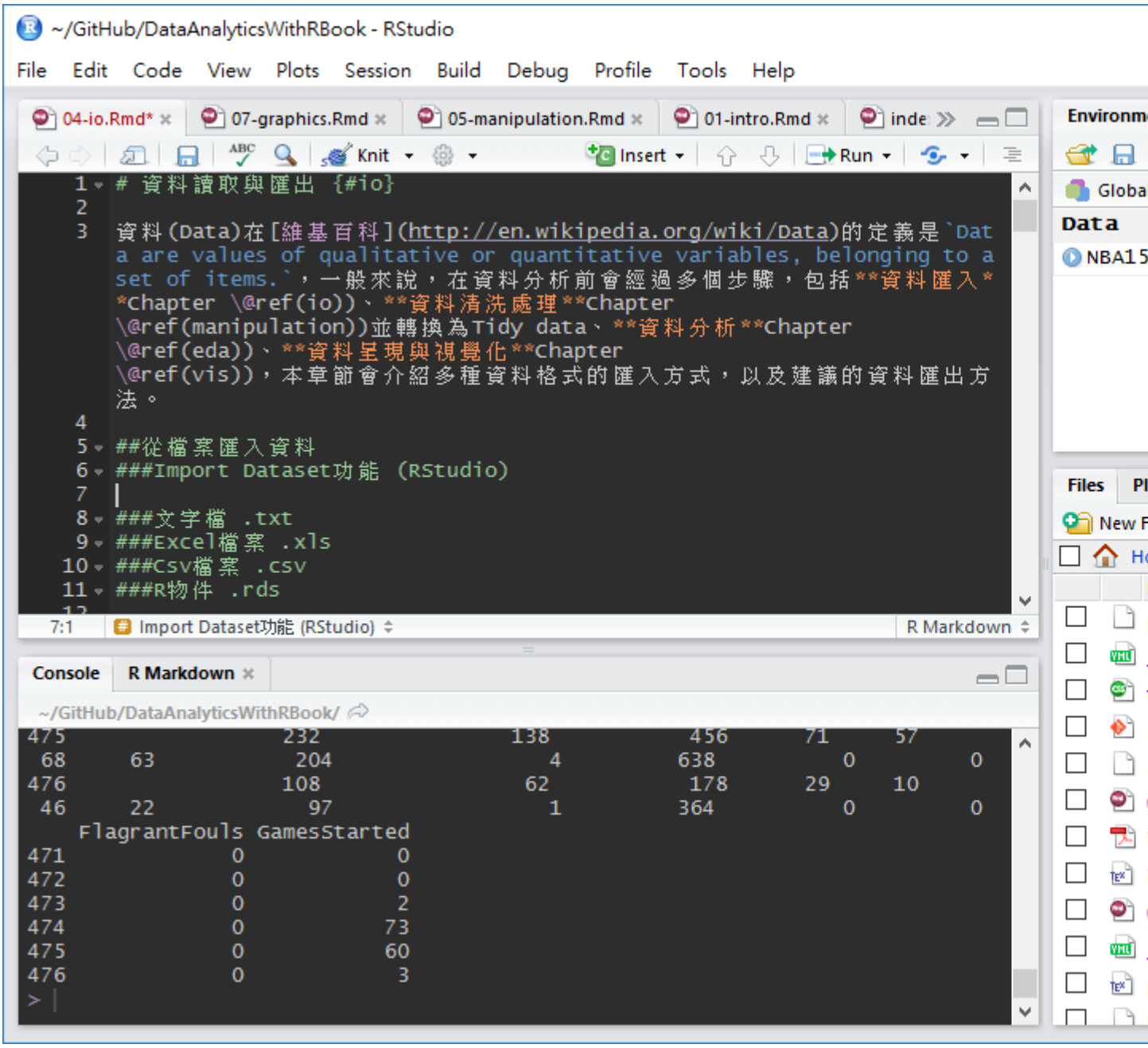
- 硬碟
- 網路下載
- Open Data (API)
- 網頁裡 (爬蟲！)
- 任何地方

以下介紹由檔案、網路等來源匯入多種資料格式的匯入方式，以及建議的資料匯出方法。

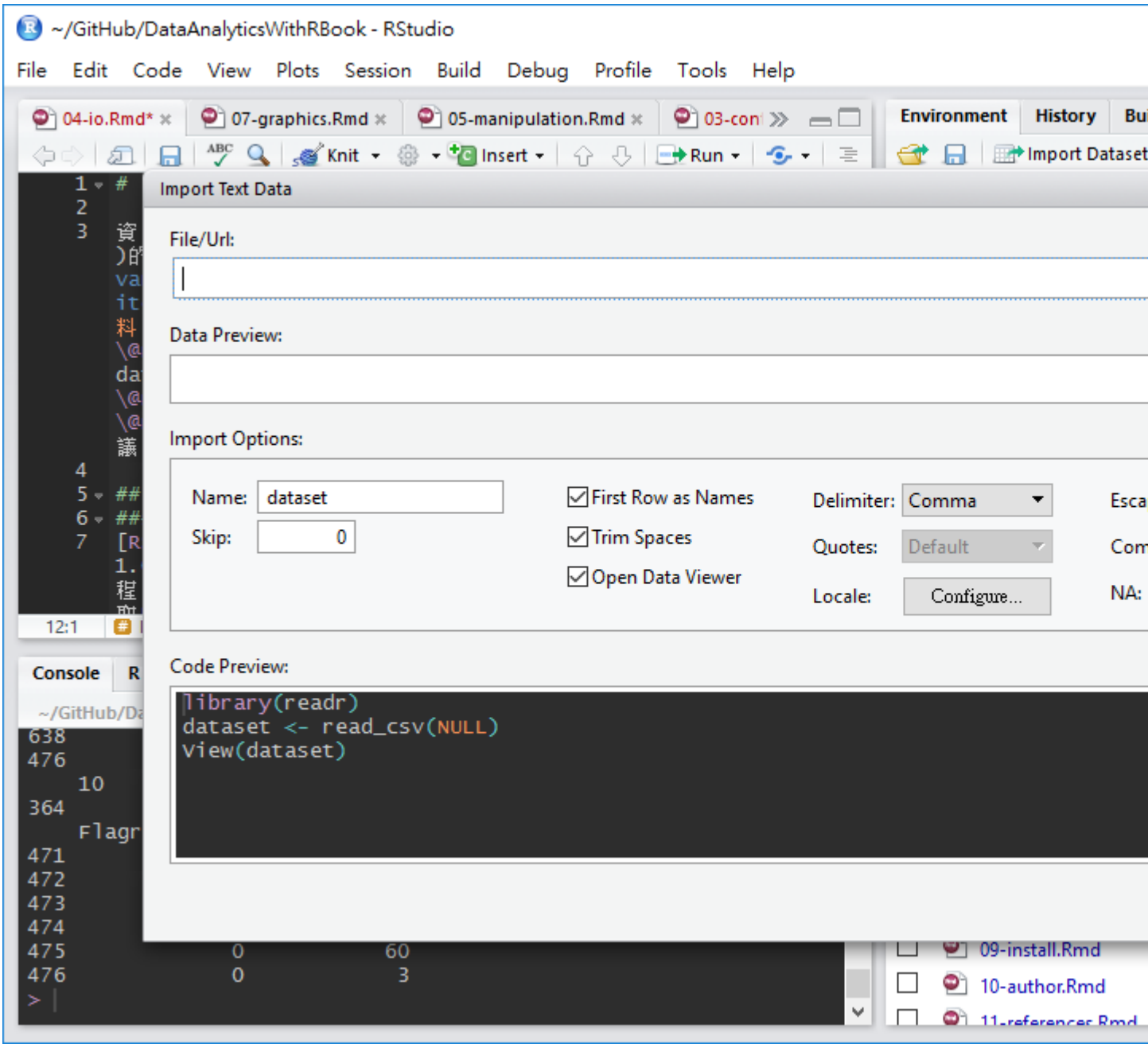
4.1 從檔案匯入基本資料格式

4.1.1 Import Dataset 功能 (RStudio)

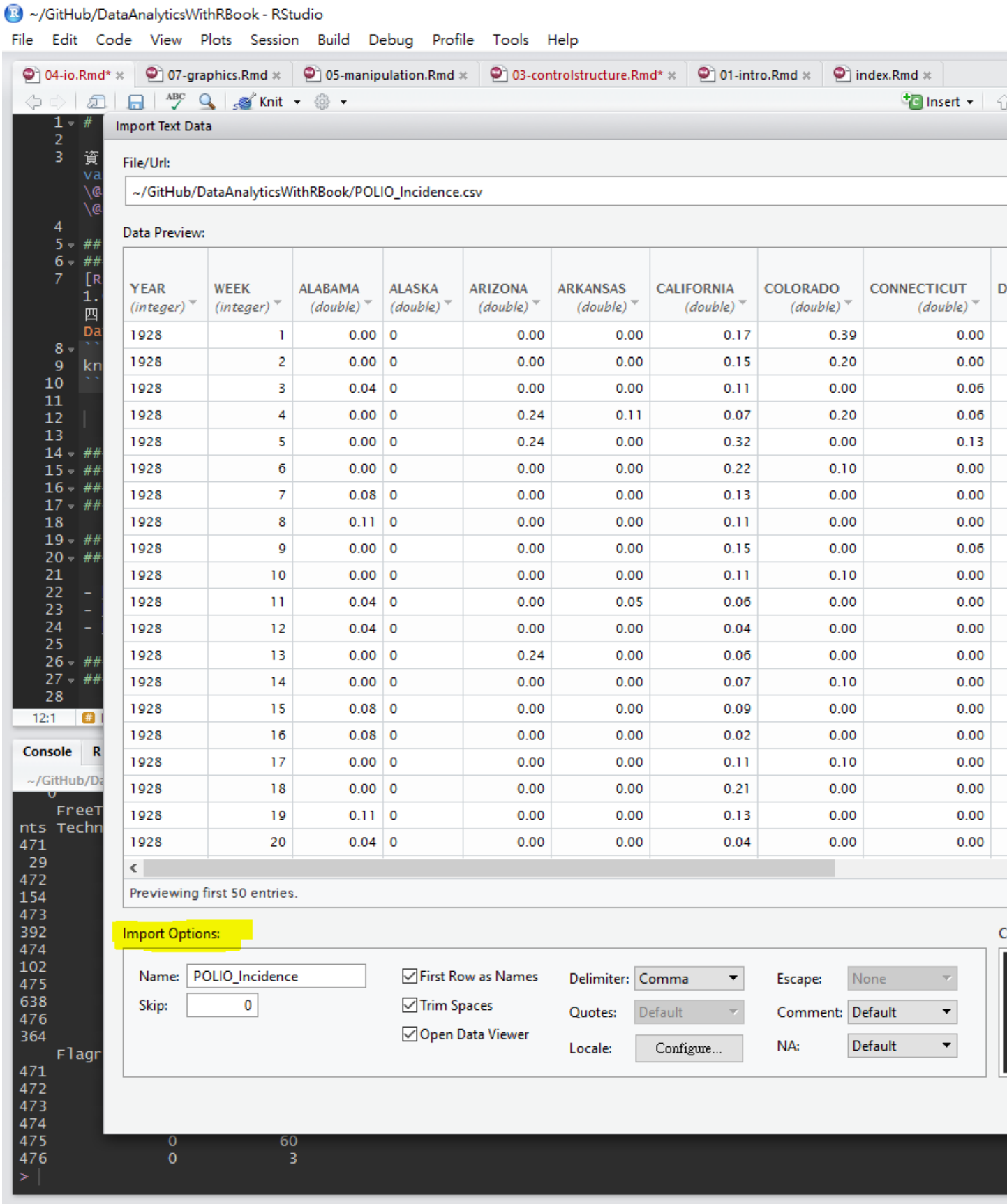
RStudio 1.0 版後即提供很好的資料匯入介面，使用者可以不用撰寫任何程式碼，就能完成`.csv`、`Excel` 以及 `SAS` 等檔案匯入。首先選取 RStudio 四分割視窗右上角的 Environment 標籤，選擇 **Import Dataset**，就會出現檔案格式的選項



以 csv 檔案為例，在選單中選取 `From CSV`，選取後會跳出資料匯入輔助視窗，點選 `Browse` 按鈕開啟檔案選取器，並點選欲匯入之文字檔案



檔案選取後，資料匯入輔助視窗有預覽功能，供使用者檢查資料匯入方法是否正確，若需調整各項參數，可利用下方 `Import Options` 的選項微調，最常用的調整功能是 `Delimiter` 分隔符號與 `First Row as Names` 首列是否為欄位名稱。


~/GitHub/DataAnalyticsWithRBook - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
04-io.Rmd*
07-graphics.Rmd
05-manipulation.Rmd
03-controlstructure.Rmd*
01-intro.Rmd
index.Rmd
Knit
Insert
Import Text Data
File/Url:
~/GitHub/DataAnalyticsWithRBook/POLIO_Incidence.csv
Data Preview:
YEAR (integer) WEEK (integer) ALABAMA (double) ALASKA (double) ARIZONA (double) ARKANSAS (double) CALIFORNIA (double) COLORADO (double) CONNECTICUT (double)
1928 1 0.00 0 0.00 0.00 0.17 0.39 0.00
1928 2 0.00 0 0.00 0.00 0.15 0.20 0.00
1928 3 0.04 0 0.00 0.00 0.11 0.00 0.06
1928 4 0.00 0 0.24 0.11 0.07 0.20 0.06
1928 5 0.00 0 0.24 0.00 0.32 0.00 0.13
1928 6 0.00 0 0.00 0.00 0.22 0.10 0.00
1928 7 0.08 0 0.00 0.00 0.13 0.00 0.00
1928 8 0.11 0 0.00 0.00 0.11 0.00 0.00
1928 9 0.00 0 0.00 0.00 0.15 0.00 0.06
1928 10 0.00 0 0.00 0.00 0.11 0.10 0.00
1928 11 0.04 0 0.00 0.05 0.06 0.00 0.00
1928 12 0.04 0 0.00 0.00 0.04 0.00 0.00
1928 13 0.00 0 0.24 0.00 0.06 0.00 0.00
1928 14 0.00 0 0.00 0.00 0.07 0.10 0.00
1928 15 0.08 0 0.00 0.00 0.09 0.00 0.00
1928 16 0.08 0 0.00 0.00 0.02 0.00 0.00
1928 17 0.00 0 0.00 0.00 0.11 0.10 0.00
1928 18 0.00 0 0.00 0.00 0.21 0.00 0.00
1928 19 0.11 0 0.00 0.00 0.13 0.00 0.00
1928 20 0.04 0 0.00 0.00 0.04 0.00 0.00
Previewing first 50 entries.
Import Options:
Name: POLIO_Incidence
Skip: 0
First Row as Names
Trim Spaces
Open Data Viewer
Delimiter: Comma
Quotes: Default
Locale: Configure...
Escape: None
Comment: Default
NA: Default
Console

如果要匯入的檔案為 **tab 分隔文字檔**，一樣可以選擇.csv 選項，再修改 Delimiter 參數為 Tab 即可。

資料匯入輔助視窗右下方 Code Preview: 子視窗中會自動產生資料匯入程式碼，如果未來想再使用視窗匯入，希望透過程式碼匯入，可以將此段程式碼複製貼上到 R 程式碼檔案 (.R)，供後續分析使用。

4.1.2 分隔文字檔.txt

readr (?) package 提供完整的文字檔讀取功能，各讀取函數的第一個參數通常為**檔案路徑與名稱**，read_delim() 函數可用來讀取所有用分隔符號分隔的文字檔案，以 tab 分隔為例，只需將 delim 參數設定為\t，即可用 tab 將各欄位分開讀取。此外，col_names 參數也常被使用，TRUE 代表資料內有包含欄位名稱 (通常在首列)，預設為 TRUE，如果設定為 FALSE，欄位名稱則會一順序被設定為 X1, X2, X3 ...。

參數整理如下 (可用?read_delim 指令閱讀官方說明)：

- file, 檔名
- delim, 分隔符號
- quote, 把欄位包起來的符號
- escape_backslash, 預設 FALSE，是否用/作為逃脫符號
- escape_double, 預設 TRUE，是否用 quote 符號作為逃脫符號
- col_names, 是否有欄位名稱 (表頭)(T/F)
- col_types, 每一個欄位的類別，用向量表示
- comment, 備註標示符號，在備註標示符號之後的文字不會被讀入
- skip, 要跳過幾行？

```
library(readr)
dataset <- read_delim(" 檔案路徑與名稱", delim="\t")
```

4.1.3 CSV 檔案.csv

readr (?) package 也提供 CSV (逗號分隔) 檔案的讀取功能，read_csv()

```
library(readr)
dataset <- read_csv(" 檔案路徑與名稱")
```

4.1.4 Excel 檔案.xls

readxl (?) package 提供讀取 Excel 檔案 (xls, xlsx) 的函數 read_excel()，除了常用的 col_names 參數外，也可使用 sheet 參數設定要讀取的工作表 (sheet)

```
library(readxl)
dataset <- read_excel(" 檔案路徑與名稱")
```

4.1.5 R 物件.rds

R 物件有檔案小與讀取快速的優點，如果在 R 程式處理資料後必須儲存一份以供後續分析的話，使用 R 物件儲存是最佳的方式，讀取 R 物件有多種函數可供選擇，推薦使用 readRDS() 函數 (參考資料:A better way of saving and loading objects in R)

```
dataset <- readRDS(" 檔案路徑與名稱")
```

4.1.6 R 程式.R

source, 讀 R 的 Obejct or script, 執行, ASCII (dump 的相反)

4.1.7 純文字資料 (無分隔)

`readLines`, 逐行讀取文字資料

4.1.8 其他讀檔注意事項

讀檔的時候 R 會自動

- 跳過 # 開頭的任何行 (Row)
- 判斷要讀幾行
- 判斷每個列 (Column) 的類別
- 把欄位包起來的符號

如果讀取時已指定 **Column 類別**以及**把欄位包起來的符號**，讀取速度會快很多。

4.2 從網路匯入資料

4.2.1 Open Data

開放資料 (Open data) 指的是一種經過挑選與許可的資料，這些資料不受著作權、專利權，以及其他管理機制所限制，可以開放給社會公眾，任何人都可以自由出版使用，不論是要拿來出版或是做其他的運用都不加以限制。Open data 運動希望達成的目標與開放原始碼、內容開放、開放獲取等其他「開放」運動類似。Open data 背後的核心思想由來已，但 Open data 這名詞直到近代才出現，拜網際網路崛起而為人所知，尤其是 Data.gov 等 Open data 政府組織的設立。維基百科

台灣政府從 2011 年開始大力推動開放政府與開放資料的概念，多個機關與縣市政府架設開放資料平台，供民眾擷取或再利用各項資料

- 政府資料開放平台
- Data Taipei
- 開放資料 x 開放桃園
- 內政資料開放平台

Open Data 常見的儲存方式為: `CSV`Chapter 4.1.3、`JSON`Chapter 4.2.3、`XML`Chapter 4.2.4，開放資料網站通常有提供民眾**直接下載**檔案的服務，針對可下載的 CSV 格式資料，可以下載完成後，透過上述**由檔案匯入資料** Chapter 4.1方法匯入即可。

4.2.2 API (Application programming interfaces)

應用程式介面 Application programming interfaces (API) 是軟體系統不同組成部分銜接的約定。由於近年來軟體的規模日益龐大，常常需要把複雜的系統劃分成小的組成部分，編程介面的設計十分重要。程式設計的實踐中，編程介面的設計首先要使軟體系統的職責得到合理劃分。良好的介面設計可以降低系統各部分的相互依賴，提高組成單元的內聚性，降低組成單元間的耦合程度，從而提高系統的維護性和擴充功能性。Wiki 但若檔案更新頻繁，以臺北市開放認養動物資料資料為例，更新頻率為每日，若使用手動下載相當耗時，所以許多開放資料也提供透過 **API** 下載的服務，透過 API 下載的資料格式會是 JSON 格式 Chapter 4.2.3，如臺北市開放認養動物 API 資訊所示，開放資料網站會提供**資料集 ID** 與**資料 RID**

- **資料集 ID**: 紀錄資料的基本參數，如包含欄位、更新頻率等
- **資料 RID**: 資料集

並同時提供擷取範例，如果需要下載原始資料，可直接從範例複製貼上即可，如 http://data.taipei/opendata/datalist/apiAccess?scope=resourceAquire&rid=f4a75ba9-7721-4363-884d-c3820b0b917c

4.2.3 JSON 格式檔案

JSON (**J**ava**s**cript **O**bject **N**otation) 是一種輕量級的資料交換語言 Wiki，特色如下:

- from **a**pplication **p**rogramming **i**nterfaces (APIs)
- JavaScript、Java、Node.js 應用
- 一些 NoSQL 非關係型資料庫用 JSON 儲存格資料：**MongoDB**
- 資料儲存格式
  - Numbers (double)
  - Strings (double quoted)
  - Boolean (*true* or *false*)
  - Array (ordered, comma separated enclosed in square brackets)
  - Object (unorderd, comma separated collection of **key:value** pairs in curley brackets *{}*)

JSON 檔案範例

許多 Open Data 也用 JSON 格式儲存，例如臺北市開放認養動物資料，根據資料的 API 資訊，可得資料擷取網址 http:// data.taipei/ opendata/ datalist/ apiAccess? scope=resourceAquire&rid=f4a75ba9-7721-4363-884d-c3820b0b917c 。

將 JSON 檔案匯入 R 可以使用 `jsonlite(?)` package，套件使用前必須安裝，安裝套件方法請參考 Chapter 1，載入後，可使用 `fromJSON()` 函數載入 JSON 資料。如需直接從 API 網址截取資料，需要載入 `RCurl(?)` package，並使用 `getURL()` 函數處理資料擷取網址。

```
library(jsonlite)
library(RCurl)
PetData<-fromJSON(getURL("http://data.taipei/opendata/datalist/apiAccess?scope=resourceAquire&rid=f4a75
str(PetData)
```

```
## List of 1
##  $ result:List of 5
##   ..$ offset : int 0
##   ..$ limit  : int 10000
##   ..$ count  : int 308
##   ..$ sort   : chr ""
##   ..$ results:'data.frame':  308 obs. of  20 variables:
##   .. ..$ _id            : chr [1:308] "1" "2" "3" "4" ...
##   .. ..$ Name           : chr [1:308] "威利" "" "妮妮 " "桔福" ...
##   .. ..$ Sex            : chr [1:308] "雄" "雄" "雌" "雄" ...
##   .. ..$ Type           : chr [1:308] "犬" "貓" "犬" "貓" ...
##   .. ..$ Build          : chr [1:308] "中" "中" "小" "中" ...
##   .. ..$ Age            : chr [1:308] "成年" "老年" "年輕" "成年" ...
##   .. ..$ Variety        : chr [1:308] "米克斯" "米克斯" "米克斯" "米克斯" ...
##   .. ..$ Reason       : chr [1:308] "民眾不擬續養" "民眾拾獲" "民眾不擬續養" "動物管制" ...
##   .. ..$ AcceptNum     : chr [1:308] "106020301" "106012706" "106012601" "106012602" ...
##   .. ..$ ChipNum        : chr [1:308] "965000000263051" "" "900073000086609" "" ...
##   .. ..$ IsSterilization: chr [1:308] "已絕育" "未絕育" "未絕育" "未絕育" ...
##   .. ..$ HairType       : chr [1:308] "黃白" "黃白" "黃" "黃白" ...
##   .. ..$ Note         : chr [1:308] "哈囉~我叫威利，我個性溫馴，但對陌生人有戒心，如果比較靠近可能會有攻
\n" "臉傷" ...
##   .. ..$ Resettlement  : chr [1:308] "臺北市動物之家 收容編號106020301" "臺北市動物之家 收容編號：106012
##   .. ..$ Phone        : chr [1:308] "02-87913062" "02-87913062" "02-87913062" "02-87913062" ...
##   .. ..$ Email        : chr [1:308] "tcapoa8@mail.taipei.gov.tw" "tcapoa8@mail.taipei.gov.tw" "tcapoa8@ma
##   .. ..$ ChildreAnlong  : chr [1:308] "" "" "" "" ...
##   .. ..$ AnimalAnlong   : chr [1:308] "" "" "" "" ...
##   .. ..$ Bodyweight     : chr [1:308] "" "" "" "" ...
```

```
##    .. ..$ ImageName        : chr [1:308] "http://163.29.39.183/uploads/images/medium/
6704940d-e354-478f-b136-2f5e167db2fd.jpg"  "http://163.29.39.183/uploads/images/medium/
e07b5bdb-5138-439d-b83e-9e0347a3cb70.jpg"  "http://163.29.39.183/uploads/images/medium/
24abcf2b-9b48-43f3-afd8-d027be7de948.jpg"  "http://163.29.39.183/uploads/images/medium/
af55b5d8-6953-4ec8-9882-b692bf130b81.jpg" ...
```

由資料結構可知，經過 `fromJSON()` 函數匯入的 JSON 檔案被轉存為列表 `list` 的型態，且在 result 元素中包含五個子元素 (offset, limit, count, sort, results)，其中，results 子元素的類別為資料框 data.frame，內含開放認養動物清單，因此，可使用 `$` 符號截取元素與子元素

```
knitr::kable(head(PetData$result$results))
```

_id	Name	Sex	Type	Build	Age	Variety	Reason	AcceptNum	ChipNum	IsSterilization	Hair
1	威利	雄	犬	中	成年	米克斯	民眾不擬續養	106020301	965000000263051	已絕育	黃白
2		雄	貓	中	老年	米克斯	民眾拾獲	106012706		未絕育	黃白
3	妮妮	雌	犬	小	年輕	米克斯	民眾不擬續養	106012601	900073000086609	未絕育	黃
4	桔福	雄	貓	中	成年	米克斯	動物管制	106012602		未絕育	黃白
5		雄	貓	大	老年	米克斯	民眾拾獲	106012510		未絕育	虎斑
6	Nana	雌	犬	中	成年	米克斯	民眾拾獲	106012504		未絕育	黑

results 資料框中包含 20 個欄位，可以像分析資料框一樣，針對此資料框做分析，舉例來說，可分析各項**開放認養理由**出現次數

```
table(PetData$result$results$Reason)
```

```
##
##                民眾不擬續養       民眾拾獲       動物救援       動物管制
##              28             41             23            105            111
```

分析可知開放認養理由以動物管制與未填寫居多。

如果需要將資料框轉換成 JSON 檔案可以使用 `jsonlite` package 所提供的 `toJSON()` 函數。

```
myjson <- toJSON(iris, pretty=TRUE)
str(myjson)
```

```
## Class 'json'  chr "[\n  {\n    \"Sepal.Length\": 5.1,\n    \"Sepal.Width\": 3.5,
\n    \"Petal.Length\": 1.4,\n    \"Petal.Width\": 0.2,\n    \"Spe"| __truncated__
```

4.2.4 XML 可延伸標記式語言

- **Ex**tensible **m**arkup **l**anguage
- 描述**結構化**資料的語言
- 處理 XML 檔案是網頁 **Html** 爬蟲的基礎
- Components
  - Markup 標記 - labels that give the text structure
  - Content 內文 - the actual text of the document
- XML Wiki

Tags, elements and attributes

- Tags correspond to general labels
  - Start tags `<breakfast_menu>`, `<price>`
  - End tags `</breakfast_menu>`,`</price>`
  - Empty tags `<line-break />`
- Elements are specific examples of tags
  - `<name>Belgian Waffles</name>`
- Attributes are components of the label
  - `<book category="web">`

許多 Open Data 也用 XML 格式儲存，例如臺北市水質監測資訊。如需將 XML 檔案匯入 R 中，需要安裝 `XML` (?) package，使用 `xmlTreeParse()` 函數將檔案匯入。

```
library(XML)
waterQ <- xmlTreeParse("http://data.taipei/opendata/datalist/datasetMeta/download?id=961ca397-4a59-45e8-
rootNode <- xmlRoot(waterQ) #access the top node
```

使用 `xpathSApply()` 函數取得指定標籤內的資料

```
# 取得所有"code_name" 標籤內的資料
xpathSApply(rootNode,"//code_name",xmlValue)[1:10]
```

```
##  [1] "雙溪淨水場"                "衛理女中"
##  [3] "雙溪國小                " "華興加壓站"
##  [5] "長興淨水場"                "市政大樓"
##  [7] "市議會"                    "捷運忠孝復興站"
##  [9] "南港高工"                  "南港加壓站"
```

```
# 取得各監測站的經度
xpathSApply(rootNode,"//longitude",xmlValue)[1:10]
```

```
##  [1] "121.56094" "121.54401" "121.55557" "121.53476" "121.54043" "121.55661"
##  [7] "121.55360" "121.53551" "121.59892" "121.60829"
```

4.2.5 網頁爬蟲 Webscraping, Crawling

Webscraping: Programatically extracting data from the HTML code of websites.

- 不是每個網站都提供 API
  - 但網頁上卻有你想要分析的資料 (像是 ptt 推文 !?)
  - 除了人工複製貼上以外，也可以將網頁處理程式化
- 可能違法……，一次讀太多太快，很可能被鎖 IP
- Webscraping Wiki
- 什麼是網頁爬蟲

What's the difference between scraping and crawling

- `Crawling`
  - dealing with `large data-sets` where you develop your own crawlers which crawl to the deepest of the `web pages`.
- `Data scraping`
  - retrieving information from `any source (not necessarily the web)`.
  - It's more often the case that irrespective of the approaches involved.

以長庚資管系網站為例，可直接逐行讀取 `readLines()`

```
htmlCode <-readLines(url("http://im.cgu.edu.tw/bin/home.php"))
```

```
## Warning in readLines(url("http://im.cgu.edu.tw/bin/home.php")): 於 'http://
## im.cgu.edu.tw/bin/home.php' 找到不完整的最後一列
```

```
htmlCode[1:5]
```

```
## [1] "<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.0 Transitional//EN\" \"http://
www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd\">"
## [2] "<html xmlns=\"http://www.w3.org/1999/xhtml\" lang=\"zh-tw\">"
## [3] "<head>"
## [4] "<meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\" />"
```

```
## [5] "<meta http-equiv=\"X-UA-Compatible\" content=\"IE=EmulateIE7\" /><meta name=\"keywords\" content=\
\xe9\x9a \x96\x8b\" />"
```

用 XML 工具讀取網頁 (XML package)

```
library(httr)
html <- htmlTreeParse(GET("http://im.cgu.edu.tw/bin/home.php"), useInternalNodes=T)
```

```
## No encoding supplied: defaulting to UTF-8.
```

```
xpathSApply(html, "//title", xmlValue)
```

```
## [1] "長庚大學 資訊管理學系 "
```

```
xpathSApply(html, "//span[@class='ptname ']", xmlValue)
```

```
##  [1] "畢業專題成果展"        "碩士班計畫書審查"      "畢業校友資料登錄"
##  [4] "長庚大學首頁"          "校務資訊系統"          "人事教育訓練資訊網"
##  [7] "國內資管系所"          "碩博士論文網"          "資管系內部行政系統"
## [10] "資管系分機表"          "資管系學會"            "TA課後輔導值班表"
## [13] "學生webmail"           "工商管理學系/研究所"   "工業設計學系/研究所"
## [16] "管理學院"              "醫務管理學系/研究所"   "商管專業學院"
## [19] "企業管理研究所博士班" "長庚大學行事曆"
```

其他爬蟲相關參考資源

- R Bloggers 有很多爬蟲範例 (英文)
- Ptt 爬蟲實作
- 大數學堂網頁爬蟲課程

4.3 Facebook 資料擷取

Facebook 提供Graph API，讓應用程式可透過 API 讀取與寫入 Facebook 相關資料，**Graph API** 會根據篩選條件，回傳 JSON 格式的資料。除此之外，Facebook 還提供Graph API Explorer，讓程式開發人員可以測試資料撈取方法和結果。

Get access token

4.3.1 Graph API in R

```
library(httr)
token<-"your token" # 將 token 複製到此處
FBData = GET(
    paste0("https://graph.facebook.com/v2.8/tsaiingwen?fields=posts%7Bmessage%7D&access_token=",
           token))
names(FBData)
```

```
## [1] "url"         "status_code" "headers"     "all_headers" "cookies"     "content"     "date"
## [8] "times"       "request"     "handle"
```

```
json1 = content(FBData)
names(json1)
```

```
## [1] "posts" "id"
```

```
names(json1$posts)
```

```
## [1] "data"   "paging"
```

```
head(json1$posts$data,3)
```

```
[[1]]
[[1]]$message
[1] "「國機國造」不是夢想，而是一個行動。今天啟動的高級教練機「自研自製」任務，是國防自主的重要里程碑。我

[[1]]$id
[1] "46251501064_10154006497451065"


[[2]]
[[2]]$message
[1] "今天，智慧機械推動辦公室正式啟動。「落實產學合作」、「支持創新研發」、「強化行銷通路」是辦公室的三項

[[2]]$id
[1] "46251501064_10154006456601065"


[[3]]
[[3]]$message
[1] "今天來向台商拜個晚年。我也邀請台商朋友們，共同參與台灣經濟轉型升級的世紀工程。\n\n無論是擴大對國內的

[[3]]$id
[1] "46251501064_10154001652641065"
```

```
json1$posts$data[[1]]$message
```

```
##[1] "「國機國造」不是夢想，而是一個行動。今天啟動的高級教練機「自研自製」任務，是國防自主的重要里程碑
```

除了直接使用 Graph API 外，也可使用 `Rfacebook` (?) package 來讀取 Facebook 資料

Use Rfacebook To Get Info from `tsaiingwen` Page

```
library(Rfacebook)
token<-"your token" # 將 token 複製到此處
getPage("tsaiingwen", token,n = 5)
```

```
5 posts       from_id           from_name
1 46251501064 蔡英文 Tsai Ing-wen
2 46251501064 蔡英文 Tsai Ing-wen
3 46251501064 蔡英文 Tsai Ing-wen
4 46251501064 蔡英文 Tsai Ing-wen
5 46251501064 蔡英文 Tsai Ing-wen

1 「國機國造」不是夢想，而是一個行動。今天啟動的高級教練機「自研自製」任務，是國防自主的重要里程碑。我們
2                                                 今天，智慧機械推動辦公室正式啟動。「落實產學合作」、「支持
3
4
5
              created_time  type
1 2017-02-07T08:02:45+0000 photo
2 2017-02-07T07:18:00+0000 photo
3 2017-02-05T07:12:52+0000 photo
4 2017-01-31T08:37:42+0000 photo
5 2017-01-30T11:41:07+0000 photo
                                                                                                    link
```

```
1 https://www.facebook.com/tsaiingwen/photos/a.390960786064.163647.46251501064/10154006497206065/?
type=3
2 https://www.facebook.com/tsaiingwen/photos/a.390960786064.163647.46251501064/10154006455396065/?
type=3
3 https://www.facebook.com/tsaiingwen/photos/a.390960786064.163647.46251501064/10154001652641065/?
type=3
4 https://www.facebook.com/tsaiingwen/photos/a.390960786064.163647.46251501064/10153989357181065/?
type=3
5 https://www.facebook.com/tsaiingwen/photos/a.390960786064.163647.46251501064/10153987089121065/?
type=3
                             id likes_count comments_count shares_count
1 46251501064_10154006497451065        2013            125           43
2 46251501064_10154006456601065        2217            163           57
3 46251501064_10154001652641065        9416            920          163
4 46251501064_10153989358051065       34116           1574          373
5 46251501064_10153987095776065       20592            665          269
```

How To Get More Data? Use **since** and **until**

Set the dates vector

```
lastDate<-Sys.Date()
DateVector<-seq(as.Date("2017-01-01"),lastDate,by="5 days")
DateVectorStr<-as.character(DateVector)
DateVectorStr
```

```
## "2017-01-01" "2017-01-06" "2017-01-11" "2017-01-16" "2017-01-21" "2017-01-26" "2017-01-31" "2017-02-05"
```

```
totalPage<-NULL
token<-'your token'
numberOfPost<-30
for(i in 1:(length(DateVectorStr)-1)){
    tempPage<-getPage("tsaiingwen", token,
                      since = DateVectorStr[i],until = DateVectorStr[i+1])
    totalPage<-rbind(totalPage,tempPage)
}
nrow(totalPage)
```

```
## 4 posts 8 posts 10 posts 3 posts 2 posts 14 posts 1 posts
## [1] 42
```

Other Useful Methods in Rfacebook Packages

- getUser()
- getPost()
- getLikes()
- Check https://github.com/pablobarbera/Rfacebook
- ?Rfacebook

4.4 資料匯出

在 R 中完成資料處理後，有多種匯出選擇，如果是要匯出供他人在其他環境 (如 Excel) 使用，建議匯出成 tab 分隔的文字檔 (.txt) 或是逗號分隔的文字檔 (.csv)；但若是要在 R 的環境繼續使用，建議匯出成 R 物件 (.rds)，除了可保留欄位型別設定外，讀取速度與檔案大小皆優於文字檔案。

4.4.1 文字檔.txt

使用 `write.table()` 函數寫入檔案，需要參數有

- `x` 要匯出的檔案，通常為 matrix 或是 data.frame 格式
- `file` 檔案名稱
- `append` T/F TRUE 表示在檔案後端加入文字，F 表示直接覆蓋原始檔案 (預設 F)
- `quote` 是否需要用雙引號將字串包起 (預設 T)
- `sep` 分隔符號 (預設空白)
- `eol` 換行符號
- `na` 表示空值的字串
- `dec` 小數點表示法
- `row.names` T/F 是否需要輸出 row names
- `col.names` T/F 是否需要輸出 column names
- `qmethod` 逃脫字串設定
- `fileEncoding` 編碼設定

```
write.table(iris,file="iris.txt",sep=",",row.names = F,col.names = T)
```

4.4.2 CSV 檔.csv

與 `write.table()` 類似，使用 `write.csv()` 函數寫入檔案

```
write.csv(iris,file="iris.csv",row.names = F)
```

4.4.3 R 物件.rds

若是要在 R 的環境繼續使用，建議匯出成 R 物件檔案 (.rds)

```
saveRDS(iris,"iris.rds")
```

Chapter 5

資料處理與清洗

5.1 Tidy Data

- 一個欄位 (Column) 內只有一個數值，最好要有凡人看得懂的 Column Name
- 不同的觀察值應該要在不同行 (Raw)
- 一張表裡面，有所有分析需要的資料
- 如果一定要多張表，中間一定要有 index 可以把表串起來
- One file, one table

5.2 資料型別轉換處理

在資料型態章節 Chapter 1.5中，曾介紹**數值 (numeric)**、**字串 (character)**、**布林變數 (logic)** 以及**日期 (Date)** 等資料型態，在此章節中將介紹如何檢查變數型別與各型別的轉換。

5.2.1 資料型別檢查

使用以下 `is.` 函數檢查資料型別，回傳布林變數，若為真，回傳 TRUE

- 是否為**數字** `is.numeric(變數名稱)`
- 是否為**文字** `is.character(變數名稱)`
- 是否為**布林變數** `is.logical(變數名稱)`

```
num<-100
cha<-'200'
boo<-T
is.numeric(num)
```

```
## [1] TRUE
```

```
is.numeric(cha)
```

```
## [1] FALSE
```

```
is.character(num)
```

```
## [1] FALSE
```

```
is.character(cha)
```

```
## [1] TRUE
```

```
is.logical(boo)
```

```
## [1] TRUE
```

或使用 `class(變數名稱)` 函數，直接回傳資料型別

```
class(num)
```

```
## [1] "numeric"
```

```
class(cha)
```

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

```
class(boo)
```

```
## [1] "logical"
```

```
class(Sys.Date())
```

```
## [1] "Date"
```

5.2.2 資料型別轉換

使用以下 `as.` 函數轉換型別

- 轉換為**數字** `as.numeric(變數名稱)`
- 轉換為**文字** `as.character(變數名稱)`
- 轉換為**布林變數** `as.logical(變數名稱)`

```
as.numeric(cha)
```

```
## [1] 200
```

```
as.numeric(boo)
```

```
## [1] 1
```

```
as.character(num)
```

```
## [1] "100"
```

```
as.character(boo)
```

```
## [1] "TRUE"
```

若無法順利完成轉換，會回傳空值 `NA`，並出現警告訊息 `Warning: NAs introduced by coercion`，`Warning:` 強制變更過程中產生了 `NA`

```
as.numeric("abc")
```

```
## Warning: 強制變更過程中產生了 NA
```

```
## [1] NA
```

日期的轉換則建議使用 `lubridate(?)` package，如果想要將年/月/日格式的文字轉換為日期物件，可使用 `ymd()` 函數 (y 表年 year，m 表月 month，d 表日 day)，如果想要將月/日/年格式的文字轉換為日期物件，則使用 `mdy()` 函數，以此類推。

```
library(lubridate)
ymd('2012/3/3')
```

```
## [1] "2012-03-03"
```

```
mdy('3/3/2012')
```

```
## [1] "2012-03-03"
```

5.3 文字字串處理

5.3.1 基本處理

- 切割 strsplit()
- 子集 substr()
- 大小寫轉換 toupper() tolower()
- 兩文字連接 paste() paste0()
- 文字取代 gsub()
- 前後空白去除 str_trim() 需安裝 stringr(?) package

```
strsplit ("Hello World"," ")
```

```
## [[1]]
## [1] "Hello" "World"
```

```
toupper("Hello World")
```

```
## [1] "HELLO WORLD"
```

```
tolower("Hello World")
```

```
## [1] "hello world"
```

```
paste("Hello", "World", sep='')
```

```
## [1] "HelloWorld"
```

```
substr("Hello World", start=2,stop=4)
```

```
## [1] "ell"
```

```
gsub("o","0","Hello World")
```

```
## [1] "Hell0 W0rld"
```

```
library(stringr)
str_trim(" Hello World ")
```

```
## [1] "Hello World"
```

5.3.2 搜尋字串

搜尋字串函數通常使用在**比對文字向量**，文字比對**有分大小寫**，依照回傳值的型態不同，有兩種常用函數，grep() 與 grepl():

- 回傳符合條件之向量位置 (index) grep(搜尋條件, 要搜尋的向量)
- 回傳每個向量是否符合條件 (TRUE or FALSE) grepl(搜尋條件, 要搜尋的向量)

```
grep("A",c("Alex","Tom","Amy","Joy","Emma")) ## 在姓名文字向量中尋找 A，回傳包含"A" 之元素位置
```

```
## [1] 1 3
```

```
grepl("A",c("Alex","Tom","Amy","Joy","Emma")) ## 在姓名文字向量中尋找 A，回傳各元素是否包含"A"
```

```
## [1]  TRUE FALSE  TRUE FALSE FALSE
```

```
grepl("a",c("Alex","Tom","Amy","Joy","Emma")) ## 在姓名文字向量中尋找 a，回傳各元素是否包含"a"
```

```
## [1] FALSE FALSE FALSE FALSE  TRUE
```

5.4 子集 Subset

5.4.1 一維資料 (向量)

在向量章節 {#vector} 有介紹使用 [] 取出單一或多個元素的方法

```
letters ##R 語言內建資料之一
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

```
letters[1] ## 取出 letters 向量的第一個元素
```

```
## [1] "a"
```

```
letters[1:10] ## 取出 letters 向量的前十個元素
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
```

```
letters[c(1,3,5)] ## 取出 letters 向量的第 1,3,5 個元素
```

```
## [1] "a" "c" "e"
```

```
letters[c(-1,-3,-5)] ## 取出 letters 向量除了第 1,3,5 個元素之外的所有元素
```

```
##  [1] "b" "d" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s" "t" "u" "v"
## [20] "w" "x" "y" "z"
```

若想要快速取得一向量的開頭與結尾元素，可使用 head() 和 tail() 函數

```
head(letters,5) ## 取出 letters 向量的前五個元素
```

```
## [1] "a" "b" "c" "d" "e"
```

```
tail(letters,3) ## 取出 letters 向量的後三個元素
```

```
## [1] "x" "y" "z"
```

5.4.2 二維資料

最常見的二維資料為 data.frame 資料框，二維資料可針對列 (Row) 和行 (Column) 做子集，子集選擇方式一樣是使用 []，但因應二維資料的需求，以，分隔列與行的篩選條件，資料篩選原則為**前 Row，後 Column，前列，後行**，若不想篩選列，則在，前方保持**空白**即可。

篩選方式可輸入位置 (index)、欄位名稱或輸入布林變數 (TRUE/FALSE)

- 輸入位置: dataFrame[row index,column index]
- 輸入布林變數: dataFrame[c(T,F,T),c(T,F,T)]
- 輸入欄位名稱: dataFrame[row name,column name]

```
iris[1,2] ## 第一列 Row，第二行 Column
```

```
## [1] 3.5
```

```
iris[1:3,] ## 第 1~3 列 Row，所有的行 Column
```

```
##   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
```

```
iris[,"Species"] ## 所有的列 Row，名稱為 Species 的行 Column
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

```
iris[1:10,c(T,F,T,F,T)] ## 第 1~10 列 Row，第 1,3,5 行 Column (TRUE)
```

```
##    Sepal.Length Petal.Length Species
## 1           5.1          1.4  setosa
## 2           4.9          1.4  setosa
## 3           4.7          1.3  setosa
## 4           4.6          1.5  setosa
## 5           5.0          1.4  setosa
## 6           5.4          1.7  setosa
## 7           4.6          1.4  setosa
## 8           5.0          1.5  setosa
## 9           4.4          1.4  setosa
## 10          4.9          1.5  setosa
```

也可使用 $ 符號做 **Column 的篩選**

```
iris$Species ## 所有的列 Row，名稱為 Species 的行 Column
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
```

```
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

Row 的篩選可使用 `subset()` 函數，使用方法為 `subset(資料表, 篩選邏輯)`

```
subset(iris,Species=="virginica") ##Species 等於"virginica" 的列 Row，所有的行 Column
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 101          6.3         3.3          6.0         2.5 virginica
## 102          5.8         2.7          5.1         1.9 virginica
## 103          7.1         3.0          5.9         2.1 virginica
## 104          6.3         2.9          5.6         1.8 virginica
## 105          6.5         3.0          5.8         2.2 virginica
## 106          7.6         3.0          6.6         2.1 virginica
## 107          4.9         2.5          4.5         1.7 virginica
## 108          7.3         2.9          6.3         1.8 virginica
## 109          6.7         2.5          5.8         1.8 virginica
## 110          7.2         3.6          6.1         2.5 virginica
## 111          6.5         3.2          5.1         2.0 virginica
## 112          6.4         2.7          5.3         1.9 virginica
## 113          6.8         3.0          5.5         2.1 virginica
## 114          5.7         2.5          5.0         2.0 virginica
## 115          5.8         2.8          5.1         2.4 virginica
## 116          6.4         3.2          5.3         2.3 virginica
## 117          6.5         3.0          5.5         1.8 virginica
## 118          7.7         3.8          6.7         2.2 virginica
## 119          7.7         2.6          6.9         2.3 virginica
## 120          6.0         2.2          5.0         1.5 virginica
## 121          6.9         3.2          5.7         2.3 virginica
## 122          5.6         2.8          4.9         2.0 virginica
## 123          7.7         2.8          6.7         2.0 virginica
## 124          6.3         2.7          4.9         1.8 virginica
## 125          6.7         3.3          5.7         2.1 virginica
## 126          7.2         3.2          6.0         1.8 virginica
## 127          6.2         2.8          4.8         1.8 virginica
## 128          6.1         3.0          4.9         1.8 virginica
```

```
## 129          6.4         2.8          5.6         2.1 virginica
## 130          7.2         3.0          5.8         1.6 virginica
## 131          7.4         2.8          6.1         1.9 virginica
## 132          7.9         3.8          6.4         2.0 virginica
## 133          6.4         2.8          5.6         2.2 virginica
## 134          6.3         2.8          5.1         1.5 virginica
## 135          6.1         2.6          5.6         1.4 virginica
## 136          7.7         3.0          6.1         2.3 virginica
## 137          6.3         3.4          5.6         2.4 virginica
## 138          6.4         3.1          5.5         1.8 virginica
## 139          6.0         3.0          4.8         1.8 virginica
## 140          6.9         3.1          5.4         2.1 virginica
## 141          6.7         3.1          5.6         2.4 virginica
## 142          6.9         3.1          5.1         2.3 virginica
## 143          5.8         2.7          5.1         1.9 virginica
## 144          6.8         3.2          5.9         2.3 virginica
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

Row 的篩選也可搭配字串搜尋函數 `grepl()`

```
knitr::kable(iris[grepl("color",iris$Species),]) ##Species 包含"color" 的列，所有的行
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
51	7.0	3.2	4.7	1.4	versicolor
52	6.4	3.2	4.5	1.5	versicolor
53	6.9	3.1	4.9	1.5	versicolor
54	5.5	2.3	4.0	1.3	versicolor
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
58	4.9	2.4	3.3	1.0	versicolor
59	6.6	2.9	4.6	1.3	versicolor
60	5.2	2.7	3.9	1.4	versicolor
61	5.0	2.0	3.5	1.0	versicolor
62	5.9	3.0	4.2	1.5	versicolor
63	6.0	2.2	4.0	1.0	versicolor
64	6.1	2.9	4.7	1.4	versicolor
65	5.6	2.9	3.6	1.3	versicolor
66	6.7	3.1	4.4	1.4	versicolor
67	5.6	3.0	4.5	1.5	versicolor
68	5.8	2.7	4.1	1.0	versicolor
69	6.2	2.2	4.5	1.5	versicolor
70	5.6	2.5	3.9	1.1	versicolor
71	5.9	3.2	4.8	1.8	versicolor
72	6.1	2.8	4.0	1.3	versicolor
73	6.3	2.5	4.9	1.5	versicolor
74	6.1	2.8	4.7	1.2	versicolor
75	6.4	2.9	4.3	1.3	versicolor
76	6.6	3.0	4.4	1.4	versicolor
77	6.8	2.8	4.8	1.4	versicolor
78	6.7	3.0	5.0	1.7	versicolor
79	6.0	2.9	4.5	1.5	versicolor
80	5.7	2.6	3.5	1.0	versicolor
81	5.5	2.4	3.8	1.1	versicolor
82	5.5	2.4	3.7	1.0	versicolor
83	5.8	2.7	3.9	1.2	versicolor
84	6.0	2.7	5.1	1.6	versicolor
85	5.4	3.0	4.5	1.5	versicolor
86	6.0	3.4	4.5	1.6	versicolor
87	6.7	3.1	4.7	1.5	versicolor
88	6.3	2.3	4.4	1.3	versicolor
89	5.6	3.0	4.1	1.3	versicolor
90	5.5	2.5	4.0	1.3	versicolor
91	5.5	2.6	4.4	1.2	versicolor
92	6.1	3.0	4.6	1.4	versicolor
93	5.8	2.6	4.0	1.2	versicolor
94	5.0	2.3	3.3	1.0	versicolor
95	5.6	2.7	4.2	1.3	versicolor
96	5.7	3.0	4.2	1.2	versicolor
97	5.7	2.9	4.2	1.3	versicolor
98	6.2	2.9	4.3	1.3	versicolor
99	5.1	2.5	3.0	1.1	versicolor
100	5.7	2.8	4.1	1.3	versicolor

若想要快速取得資料框的前幾列 (Raw) 或後幾列，也可使用 `head()` 和 `tail()` 函數

```
head(iris,5) ## 取出 iris 資料框的前五列
```

```
##   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
```

```
tail(iris,3) ## 取出 iris 資料框的後三列
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

5.5 排序

5.5.1 sort 向量排序

sort() 函數可直接對向量做**由小到大**的排序

```
head(islands) ## 排序前的前六筆資料
```

```
##       Africa   Antarctica         Asia    Australia Axel Heiberg       Baffin
##        11506         5500        16988         2968           16          184
```

```
head(sort(islands)) ## 由小到大排序後的前六筆資料
```

```
##       Vancouver          Hainan Prince of Wales           Timor          Kyushu
##              12              13              13              13              14
##          Taiwan
##              14
```

如需**由大到小**排序，可將 decreasing 參數設為 TRUE

```
head(sort(islands,decreasing = T)) ## 由大到小排序後的前六筆資料
```

```
##          Asia        Africa North America South America    Antarctica
##         16988         11506          9390          6795          5500
##        Europe
##          3745
```

5.5.2 order

如需對資料框做排序，可使用 order() 函數，order() 函數可回傳**由小到大之元素位置**，以 iris$Sepal.Length 為例，回傳的第一個位置為 14，表示 iris$Sepal.Length 中，數值最小的元素為第 14 個元素。

```
order(iris$Sepal.Length)
```

```
##   [1]  14   9  39  43  42   4   7  23  48   3  30  12  13  25  31  46   2  10
##  [19]  35  38  58 107   5   8  26  27  36  41  44  50  61  94   1  18  20  22
##  [37]  24  40  45  47  99  28  29  33  60  49   6  11  17  21  32  85  34  37
##  [55]  54  81  82  90  91  65  67  70  89  95 122  16  19  56  80  96  97 100
##  [73] 114  15  68  83  93 102 115 143  62  71 150  63  79  84  86 120 139  64
##  [91]  72  74  92 128 135  69  98 127 149  57  73  88 101 104 124 134 137 147
```

```
## [109]  52  75 112 116 129 133 138  55 105 111 117 148  59  76  66  78  87 109
## [127] 125 141 145 146  77 113 144  53 121 140 142  51 103 110 126 130 108 131
## [145] 106 118 119 123 136 132
```

```
iris$Sepal.Length[14]
```

```
## [1] 4.3
```

若將 decreasing 參數設定為 TRUE，則會回傳**由大到小**的元素位置，以 iris$Sepal.Length 為例，回傳的第一個位置為 132，表示 iris$Sepal.Length 中，數值最大的元素為第 132 個元素。

```
order(iris$Sepal.Length,decreasing = T)
```

```
##   [1] 132 118 119 123 136 106 131 108 110 126 130 103  51  53 121 140 142  77
##  [19] 113 144  66  78  87 109 125 141 145 146  59  76  55 105 111 117 148  52
##  [37]  75 112 116 129 133 138  57  73  88 101 104 124 134 137 147  69  98 127
##  [55] 149  64  72  74  92 128 135  63  79  84  86 120 139  62  71 150  15  68
##  [73]  83  93 102 115 143  16  19  56  80  96  97 100 114  65  67  70  89  95
##  [91] 122  34  37  54  81  82  90  91   6  11  17  21  32  85  49  28  29  33
## [109]  60   1  18  20  22  24  40  45  47  99   5   8  26  27  36  41  44  50
## [127]  61  94   2  10  35  38  58 107  12  13  25  31  46   3  30   4   7  23
## [145]  48  42   9  39  43  14
```

```
iris$Sepal.Length[132]
```

```
## [1] 7.9
```

依照 order 回傳的元素位置，重新排序 iris 資料框

```
head(iris) ## 排序前的前六筆資料
```

```
##   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
```

```
head(iris[order(iris$Sepal.Length),]) ## 依照 Sepal.Length 欄位數值大小排序後的前六筆資料
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 14          4.3         3.0          1.1         0.1  setosa
## 9           4.4         2.9          1.4         0.2  setosa
## 39          4.4         3.0          1.3         0.2  setosa
## 43          4.4         3.2          1.3         0.2  setosa
## 42          4.5         2.3          1.3         0.3  setosa
## 4           4.6         3.1          1.5         0.2  setosa
```

```
head(iris[order(iris$Sepal.Length,decreasing = T),]) ## 改為由大到小排序的前六筆資料
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 132          7.9         3.8          6.4         2.0 virginica
## 118          7.7         3.8          6.7         2.2 virginica
## 119          7.7         2.6          6.9         2.3 virginica
## 123          7.7         2.8          6.7         2.0 virginica
## 136          7.7         3.0          6.1         2.3 virginica
## 106          7.6         3.0          6.6         2.1 virginica
```

5.6 資料組合

有時需要在資料框新增一列，或新增一行，可以利用資料組合函數完成

- Row 列的組合 `rbind()`
- Column 行的組合 `cbind()`

`rbind()` 和 `cbind()` 的參數可以是向量，也可以是資料框，使用向量做資料整合範例:

```
rbind(c(1,2,3), # 第一列
      c(4,5,6)  # 第二列
      )
```

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

使用資料框與向量做資料整合範例:

```
irisAdd<-rbind(iris, # 資料框
      c(1,1,1,1,"versicolor")  # 新增一列
      )
tail(irisAdd)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 146          6.7           3          5.2         2.3  virginica
## 147          6.3         2.5            5         1.9  virginica
## 148          6.5           3          5.2           2  virginica
## 149          6.2         3.4          5.4         2.3  virginica
## 150          5.9           3          5.1         1.8  virginica
## 151            1           1            1           1 versicolor
```

使用向量做資料整合範例:

```
cbind(c(1,2,3), # 第一行
      c(4,5,6)  # 第二行
      )
```

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

使用資料框與向量做資料整合範例:

```
irisAdd<-cbind(iris, # 資料框
      rep("Add",nrow(iris))  # 新增一行
      )
tail(irisAdd)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
##     rep("Add", nrow(iris))
## 145                    Add
```

```
## 146                    Add
## 147                    Add
## 148                    Add
## 149                    Add
## 150                    Add
```

5.7 長表與寬表

在資料處理的過程中，常因各種需求，需要執行長寬表互換的動作，在 R 中有很好用的套件 reshape2(?) package，提供完整的轉換功能，最常使用的是

- 寬表轉長表 `melt(資料框/寬表,id.vars= 需要保留的欄位)`
- 長表轉寬表 `dcast(資料框/長表, 寬表分列依據 ~ 分欄位依據)`

原來的 `airquality` 資料框中，有 Ozone, Solar.R, Wind, Temp, Month, Day 等六個欄位 (Column)，屬於寬表，以下範例將保留 Month 和 Day 兩個欄位，並將其他欄位的名稱整合至 variable 欄位，數值整合至 value 欄位，寬表轉長表範例如下:

```
library(reshape2)
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
```

```
airqualityM<-melt(airquality,id.vars = c("Month","Day")) ## 欄位需要保留"Month","Day"
head(airqualityM)
```

```
##   Month Day variable value
## 1     5   1    Ozone    41
## 2     5   2    Ozone    36
## 3     5   3    Ozone    12
## 4     5   4    Ozone    18
## 5     5   5    Ozone    NA
## 6     5   6    Ozone    28
```

轉換過的長表 `airqualityM` 資料框中，剩下 Month, Day, variable, value 等四個欄位 (Column)，屬於長表，以下範例 variable 欄位的值轉換為新欄位，並將 value 欄位填回新增的欄位，長表轉寬表範例如下:

```
library(reshape2)
## 欄位保留"Month","Day" 外，其他欄位數目由 variable 定義
airqualityCast<-dcast(airqualityM, Month +Day~variable)
head(airqualityCast)
```

```
##   Month Day Ozone Solar.R Wind Temp
## 1     5   1    41     190  7.4   67
## 2     5   2    36     118  8.0   72
## 3     5   3    12     149 12.6   74
## 4     5   4    18     313 11.5   62
## 5     5   5    NA      NA 14.3   56
## 6     5   6    28      NA 14.9   66
```

5.8 遺漏值處理

遺漏值 (Missing Value) 常常出現在真實資料內，在數值運算時常會有問題，最簡單的方法是將有缺值的資料移除，如資料為向量，可使用 `is.na()` 來判斷資料是否為空值 `NA`，若為真 `TRUE`，則將資料移除。

```
naVec<-c("a","b",NA,"d","e")
is.na(naVec)
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE
```

```
naVec[!is.na(naVec)] ## 保留所有在 is.na() 檢查回傳 FALSE 的元素
```

```
## [1] "a" "b" "d" "e"
```

若資料型態為資料框，可使用 `complete.cases` 來選出完整的資料列，如果資料列是完整的，則會回傳真 TRUE

```
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
```

```
complete.cases(airquality)
```

```
##   [1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
##  [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [25] FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE
##  [49]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
##  [73]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE
##  [85]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
##  [97] FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## [109]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## [121]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [145]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
```

```
head(airquality[complete.cases(airquality),]) ## 保留所有在 complete.cases() 檢查回傳 TRUE 的元素
```

```
##   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
## 7    23     299  8.6   65     5   7
## 8    19      99 13.8   59     5   8
```

利用演算法補值也是一種解決辦法，可參考 _skydome20_ 的R 筆記–(10) 遺漏值處理 (Impute Missing Value)教學。

5.9 綜合練習範例

在本範例中，介紹使用 `SportsAnalytics` (?) package 撈取 NBA 各球員的數據，並加以觀察分析。

5.9.1 載入資料

首先用 library() 函數將 SportsAnalytics 套件載入 (若尚未安裝此套件者，必須先安裝套件，可參考 Chapter 1)，並利用套件內提供的 fetch_NBAPlayerStatistics() 函數，將對應年份之資料取出。

```
library(SportsAnalytics)
NBA1516<-fetch_NBAPlayerStatistics("15-16")
```

5.9.2 資料總覽

資料取出後，可用 str() 函數總覽 NBA1516 這個資料框的欄位與欄位類別

```
str(NBA1516)
```

```
## 'data.frame':    476 obs. of  25 variables:
##  $ League             : Factor w/ 1 level "NBA": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Name               : chr  "Quincy Acy" "Jordan Adams" "Steven Adams" "Arron Afflalo" ...
##  $ Team           : Factor w/ 31 levels "ATL","BOS","BRO",..: 27 15 22 20 19 13 28 26 12 15 ...
##  $ Position           : Factor w/ 5 levels "C","PF","PG",..: 4 5 1 5 1 1 2 2 2 5 ...
##  $ GamesPlayed        : int  59 2 80 71 59 60 74 9 79 64 ...
##  $ TotalMinutesPlayed : int  877 15 2019 2359 863 802 2260 37 1601 1622 ...
##  $ FieldGoalsMade     : int  119 2 261 354 150 134 536 5 191 215 ...
##  $ FieldGoalsAttempted: int  214 6 426 799 314 225 1045 10 370 469 ...
##  $ ThreesMade         : int  19 0 0 91 0 0 0 0 0 15 ...
##  $ ThreesAttempted    : int  49 1 0 238 1 0 16 0 0 42 ...
##  $ FreeThrowsMade     : int  50 3 114 110 52 60 259 0 46 90 ...
##  $ FreeThrowsAttempted: int  68 5 196 131 62 84 302 0 73 138 ...
##  $ OffensiveRebounds  : int  65 0 218 23 75 86 175 2 162 104 ...
##  $ TotalRebounds      : int  188 2 531 266 269 288 631 6 424 297 ...
##  $ Assists            : int  27 3 61 145 32 50 110 0 76 70 ...
##  $ Steals             : int  29 3 42 25 19 47 38 1 26 109 ...
##  $ Turnovers          : int  27 2 84 82 54 64 99 1 69 78 ...
##  $ Blocks             : int  24 0 89 10 36 68 81 2 42 18 ...
##  $ PersonalFouls      : int  103 2 223 142 134 139 151 1 147 175 ...
##  $ Disqualifications  : int  0 0 2 1 0 1 0 0 1 1 ...
##  $ TotalPoints        : int  307 7 636 909 352 328 1331 10 428 535 ...
##  $ Technicals         : int  3 0 2 1 2 0 0 0 0 1 ...
##  $ Ejections          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FlagrantFouls      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GamesStarted       : int  29 0 80 57 17 5 74 0 28 56 ...
```

可以發現此 NBA1516 資料框內有 476 筆球員資料 (觀察值, obs)，每筆資料有 25 個欄位 (variables)。### 資料預覽如果想看資料框內容，可用 head() 和 tail() 快速瀏覽部分資料

```
head(NBA1516)
```

```
##   League          Name Team Position GamesPlayed TotalMinutesPlayed
## 1    NBA    Quincy Acy  SAC       SF          59                877
## 2    NBA  Jordan Adams  MEM       SG           2                 15
## 3    NBA  Steven Adams  OKL        C          80               2019
## 4    NBA Arron Afflalo  NYK       SG          71               2359
## 5    NBA Alexis Ajinca  NOR        C          59                863
## 6    NBA  Cole Aldrich  LAC        C          60                802
##   FieldGoalsMade FieldGoalsAttempted ThreesMade ThreesAttempted FreeThrowsMade
## 1            119                 214         19              49             50
```

```
## 2              2                   6          0               1              3
## 3            261                 426          0               0            114
## 4            354                 799         91             238            110
## 5            150                 314          0               1             52
## 6            134                 225          0               0             60
##   FreeThrowsAttempted OffensiveRebounds TotalRebounds Assists Steals Turnovers
## 1                  68                65           188      27     29        27
## 2                   5                 0             2       3      3         2
## 3                 196               218           531      61     42        84
## 4                 131                23           266     145     25        82
## 5                  62                75           269      32     19        54
## 6                  84                86           288      50     47        64
##   Blocks PersonalFouls Disqualifications TotalPoints Technicals Ejections
## 1     24            103                 0         307          3         0
## 2      0              2                 0           7          0         0
## 3     89            223                 2         636          2         0
## 4     10            142                 1         909          1         0
## 5     36            134                 0         352          2         0
## 6     68            139                 1         328          0         0
##   FlagrantFouls GamesStarted
## 1             0           29
## 2             0            0
## 3             0           80
## 4             0           57
## 5             0           17
## 6             0            5
```

5.9.3 資料排序後篩選

觀察資料框的組成後，我們想要找出**出場數最高**的前五名選手的所有資料，此時可以利用 `order()` 函數先**由大到小**排序 (`decreasing = T`) 後，再用 `[,]` 取子集。

```
NBA1516Order<-NBA1516[order(NBA1516$GamesPlayed,decreasing = T),]
NBA1516Order[1:5,] ## 逗號前方放 1~5，表示取 1~5 列；逗號後方空白，表示要取所有欄位
```

```
##     League            Name Team Position GamesPlayed TotalMinutesPlayed
## 11     NBA Al-farouq Aminu  POR       SF          82               2342
## 37     NBA     Will Barton  DEN       SG          82               2355
## 48     NBA Bismack Biyombo  TOR       PF          82               1810
## 62     NBA    Corey Brewer  HOU       SG          82               1670
## 118    NBA    Gorgui Dieng  MIN        C          82               2222
##     FieldGoalsMade FieldGoalsAttempted ThreesMade ThreesAttempted
## 11             299                719        126             349
## 37             426                984        112             324
## 48             156                288          0               1
## 62             212                552         61             225
## 118            308                578          6              20
##     FreeThrowsMade FreeThrowsAttempted OffensiveRebounds TotalRebounds Assists
## 11             115                156                98           498     138
## 37             216                268                60           477     204
## 48             142                226               182           655      29
## 62             105                140                42           199     109
## 118            205                248               156           584     143
##     Steals Turnovers Blocks PersonalFouls Disqualifications TotalPoints
```

```
## 11      72       120     53           171                 0         839
## 37      71       139     39           147                 0        1180
## 48      19        71    133           225                 2         454
## 62      84        78     19           168                 1         590
## 118     94       140     96           219                 0         827
##     Technicals Ejections FlagrantFouls GamesStarted
## 11           3         0             0           82
## 37           2         0             0            1
## 48           3         0             0           22
## 62           0         0             0           12
## 118          1         0             0           39
```

如果我們想要出**出場分鐘數**最**高**的前十名選手的**名字**，一樣可以用 `order()` 函數先**由大到小**排序 (`decreasing = T`) 後，再用 `[,]` 取子集。

```
NBA1516OrderM<-NBA1516[order(NBA1516$TotalMinutesPlayed,decreasing = T),]
NBA1516OrderM[1:10,"Name"] ## 逗號前方取 1~10 列；逗號後方放"Name"，表示取名稱為 Name 之欄位
```

```
##  [1] "James Harden"     "Gordon Hayward"   "Kemba Walker"     "Trevor Ariza"
##  [5] "Khris Middleton"  "Kyle Lowry"       "Marcus Morris"    "Andrew Wiggins"
##  [9] "Paul George"      "Gi Antetokounmpo"
```

5.9.4 欄位值篩選

除了排序取值外，也可用欄位條件搜尋，舉例來說，可以取出所有波士頓賽爾迪克隊的選手資料，使用 `subset()` 函數

```
subset(NBA1516,Team=="BOS")
```

```
##     League            Name Team Position GamesPlayed TotalMinutesPlayed
## 60     NBA   Avery Bradley  BOS       PG          76                2536
## 89     NBA     Coty Clarke  BOS     <NA>           4                   8
## 102    NBA     Jae Crowder  BOS       SF          73                2310
## 213    NBA     R.j. Hunter  BOS       SG          36                 319
## 228    NBA   Jonas Jerebko  BOS       PF          78                1178
## 229    NBA    Amir Johnson  BOS       PF          79                1798
## 300    NBA   Jordan Mickey  BOS       PF          16                  59
## 340    NBA    Kelly Olynyk  BOS        C          69                1396
## 382    NBA    Terry Rozier  BOS       PG          39                 310
## 400    NBA    Marcus Smart  BOS       PG          61                1666
## 416    NBA Jared Sullinger  BOS       PF          81                1917
## 422    NBA   Isaiah Thomas  BOS       PG          82                2647
## 433    NBA     Evan Turner  BOS       SG          81                2270
## 471    NBA     James Young  BOS       SG          29                 200
## 476    NBA    Tyler Zeller  BOS        C          60                 714
##     FieldGoalsMade FieldGoalsAttempted ThreesMade ThreesAttempted
## 60             456                1018        147             406
## 89               2                   4          2               2
## 102            359                 812        122             363
## 213             36                  98         19              63
## 228            118                 286         43             108
## 229            250                 427         10              43
## 300              8                  22          0               0
## 340            253                 556         85             210
## 382             29                 106          6              27
## 400            184                 529         61             241
```

```
## 416            351                 807         29            104
## 422            591                1382        167            465
## 433            343                 753         20             83
## 471             11                  36          6             26
## 476            138                 290          0              0
##     FreeThrowsMade FreeThrowsAttempted OffensiveRebounds TotalRebounds Assists
## 60              96                 123                48           220     158
## 89               0                   0                 0             1       0
## 102            196                 239                70           373     135
## 213              6                   7                 2            37      13
## 228             61                  78                77           288      62
## 229             69                 121               178           505     137
## 300              5                  10                 6            13       1
## 340             96                 128                72           281     105
## 382              8                  10                24            63      37
## 400            129                 166                76           255     186
## 416            103                 161               194           673     187
## 422            474                 544                46           243     509
## 433            148                 179                50           397     359
## 471              1                   4                 4            26       9
## 476             88                 108                62           178      29
##     Steals Turnovers Blocks PersonalFouls Disqualifications TotalPoints
## 60     117       109     19           164                 2        1155
## 89       0         1      0             0                 0           6
## 102    126        83     35           198                 4        1036
## 213     14        11      4            29                 0          97
## 228     20        52     24           137                 2         340
## 229     52        94     83           214                 1         579
## 300      0         1     11             5                 0          21
## 340     52        74     33           163                 3         687
## 382      6        19      1            23                 0          72
## 400     91        80     18           183                 1         558
## 416     75       102     47           209                 2         834
## 422     91       220      9           167                 1        1823
## 433     80       169     28           139                 0         854
## 471      6         5      1            17                 0          29
## 476     10        46     22            97                 1         364
##     Technicals Ejections FlagrantFouls GamesStarted
## 60           0         0             0           72
## 89           0         0             0            0
## 102          3         0             0           73
## 213          0         0             0            0
## 228          1         0             0            0
## 229          0         0             0           76
## 300          0         0             0            0
## 340          1         0             0            8
## 382          0         0             0            0
## 400          2         0             0           10
## 416          2         0             0           73
## 422          9         0             0           79
## 433          2         0             0           12
## 471          0         0             0            0
## 476          0         0             0            3
```

5.9.5 字串條件搜尋後篩選

當然也可以結合**字串搜尋**函數 `grepl()`，將所有名字裡有“James”的選手資料取出

```
NBA1516[grepl("James",NBA1516$Name),]
```

```
##     League           Name Team Position GamesPlayed TotalMinutesPlayed
## 15     NBA James Anderson  SAC       SG          51                721
## 132    NBA    James Ennis  NOR       SF          22                329
## 178    NBA   James Harden  HOU       SG          82               3121
## 222    NBA   Lebron James  CLE       SF          76               2710
## 231    NBA  James Johnson  TOR       PF          57                924
## 239    NBA    James Jones  CLE       SG          48                466
## 286    NBA   James Mcadoo  GSW       SG          41                265
## 471    NBA    James Young  BOS       SG          29                200
##     FieldGoalsMade FieldGoalsAttempted ThreesMade ThreesAttempted
## 15              67                 178         23              86
## 132             54                 113         26              58
## 178            710                1617        236             656
## 222            737                1416         87             282
## 231            114                 240         20              66
## 239             59                 143         41             104
## 286             45                  84          1               2
## 471             11                  36          6              26
##     FreeThrowsMade FreeThrowsAttempted OffensiveRebounds TotalRebounds Assists
## 15              22                 29                13            86      41
## 132             25                 34                21            42      21
## 178            720                837                63           502     612
## 222            359                491               111           565     512
## 231             39                 68                28           126      67
## 239             21                 26                 8            50      14
## 286             26                 49                30            58      17
## 471              1                  4                 4            26       9
##     Steals Turnovers Blocks PersonalFouls Disqualifications TotalPoints
## 15      21        42     14            54                 0         179
## 132     16        19      5            28                 1         159
## 178    138       374     51           229                 1        2376
## 222    104       249     49           143                 0        1920
## 231     29        54     33            84                 0         287
## 239     11        13     10            50                 0         180
## 286     10        16      8            39                 0         117
## 471      6         5      1            17                 0          29
##     Technicals Ejections FlagrantFouls GamesStarted
## 15           0         0             0           15
## 132          0         0             0            5
## 178          2         0             0           82
## 222          3         0             0           76
## 231          0         0             0           32
## 239          1         0             0            0
## 286          0         0             0            1
## 471          0         0             0            0
```

Chapter 6

探索式資料分析

撰寫中

6.1 資料內容初步分析統計

6.2 data.table

Chapter 7

資料視覺化

撰寫中

Chapter 8

從小數據到大數據分析

8.1 R + Hadoop

8.2 RHadoop 安裝測試流程 (Cloudera)

安裝與測試日期 2016/05/12

8.2.1 系統/軟體版本資訊

- Cloudera Hadoop Platform: CDH-5.4.5 下載
- R for Linux 3.3.0 安裝說明
- RStudio Server 下載
- RHadoop (latest version on May 12, 2016) 下載
  - ravro-1.0.3
  - plyrmr-0.6.0
  - rmr-3.3.1
  - rhdfs-1.0.8
  - rhbase-1.2.1

8.2.2 參考資料

- RHadoop 安裝說明文件
- RHadoop 安裝步驟
- Setting persistent environment variable in CentOS 7 issue
- How to resolve “Permission denied” errors in CDH

8.2.3 安裝步驟

1. 下載 Cloudera CDH QuickStart VM Cloudera VM
2. 安裝 R 安裝說明
3. 安裝 RHadoop RHadoop 安裝步驟
4. 安裝 RStudio Server 說明

8.2.3.1 Cloudera CDH QuickStart VM

Cloudera CDH QuickStart VM 是由 Cloudera 提供的虛擬機器，內涵 Linux 系統與預載多項 Hadoop 相關服務，適合想了解 Hadoop 運作的初學者。

下載 VM 後，用 Virtural Box 開啟即可。

- Cloudera CDH QuickStart VM 下載處
- Virtural Box 下載處

以下安裝步驟都在 Cloudera CDH QuickStart VM 內進行

8.2.3.2 安裝 R

- Cloudera CDH 用的 Linux 作業系統是 CentOS
- 依照安裝說明，需要先安裝 Extra Packages for Enterprise Linux (EPEL)，但系統內有預載，所以可以不用按照說明重新下載安裝，直接執行 `sudo yum install epel-release` 指令即可
- 步驟：安裝最新 EPRL，更新 yum，安裝 R。打開 Terminal 輸入以下指令。

```
sudo yum install epel-release
sudo yum update
sudo yum install R
```

8.2.3.3 安裝 RHadoop-1 先進行環境設定

設定 `HADOOP_CMD` 與 `HADOOP_STREAMING` 兩項環境參數，路徑可能會不同 (尤其是 `HADOOP_STREAMING`)

1. 尋找 `HADOOP_STREAMING` 路徑方法

   ```
   find / -name hadoop-streaming-*.jar
   ```

2. 設定 `HADOOP_CMD` 與 `HADOOP_STREAMING` 兩項環境參數，路徑記得換成自己的

   ```
   echo export HADOOP_CMD="/usr/bin/hadoop">/etc/profile.d/hadoopenv.sh
   echo export HADOOP_STREAMING=
   "/opt/cloudera/parcels/CDH-5.4.5-1.cdh5.4.5.p0.7/lib/hadoop-mapreduce/
       hadoop-streaming-2.6.0-cdh5.4.5.jar" > /etc/profile.d/hadoopenv.sh
   chmod 0755 /etc/profile.d/hadoopenv.sh
   ```

8.2.3.4 安裝 RHadoop-2 rmr2

- 每個 Node 都要裝
- 安裝前先至說明檔看需要先安裝哪些其他的 packages，Depends 和 Imports 所列的 packages 都要裝
- 以下為安裝 packages 的程式碼，在 R 內執行 (在 Terminal 輸入 `R`，就能進入 R 軟體)

```
install.packages(c("methods","Rcpp", "RJSONIO", "digest", "functional",
                   "reshape2","stringr", "plyr", "caTools","quickcheck","testthat"),
                 dependencies=TRUE, repos='http://cran.rstudio.com/')
```

- 使用 `q()` 指令，跳出 R 軟體
- 下載 rmr2
- 安裝 (請將 `rmr2_2.3.0.tar.gz` 替換成剛剛下載的安裝檔路徑)

```
sudo R CMD INSTALL rmr2_2.3.0.tar.gz
```

8.2.3.5 安裝 RHadoop-3 rhdfs

- 只要裝在會跑 R 的那個 Node
- 在裝之前，先 Check 是否有安裝 JDK （測試 JDK 1.8.0_91 沒問題）
- Check 環境變數 JAVA_HOME 是否有設好

```
echo $JAVA_HOME
```

若什麼都沒有回傳，先設定環境變數（將/usr/java/jdk1.8.0_91 換成自己的路徑）

```
echo export JAVA_HOME="/usr/java/jdk1.8.0_91">/etc/profile.d/jdkenv.sh
```

為了讓 R 可以跑 JAVA，在 Terminal 輸入

```
R CMD javareconf
```

然後進到 R 程式（在 Terminal 輸入 R，就能進入 R 軟體），安裝 rJava package

```
install.packages("rJava",dependencies=TRUE, repos='http://cran.rstudio.com/'){target="_blank"}
```

最後跳出 R 程式，下載 rhdfs，安裝 rhdfs

- 將/usr/bin/hadoop 換成自己的 HADOOP_CMD 路徑
- rhdfs_1.0.8.tar.gz 換成下載的安裝檔路徑）

```
sudo HADOOP_CMD=/usr/bin/hadoop R CMD INSTALL rhdfs_1.0.8.tar.gz
```

8.2.4 測試前，先解決權限問題

- 預設 hdfs 的存取權限不足，所以要打開
- 將 user01 改為自己的使用者名稱

```
sudo -u hdfs hadoop fs -mkdir /user/user01
sudo -u hdfs hadoop fs -chown user01 /user/user01
```

8.2.5 測試

進入 R 測試以下程式碼是否能執行

```
Sys.setenv(HADOOP_CMD="/usr/bin/hadoop")
Sys.setenv(HADOOP_STREAMING="/opt/cloudera/parcels/CDH-5.4.5-1.cdh5.4.5.p0.7/lib/hadoop-mapreduce/hadoo
library(rmr2)
#test mapreduce
small.ints = to.dfs(1:100)
out<-mapreduce(
    input = small.ints,
    map = function(., v) cbind(v, v^2))
head(from.dfs(out))
```

8.2.6 安裝 RStudio Server

官方下載與安裝說明

在 Terminal 執行以下程式碼

- 檔案連結 https://download2.rstudio.org/rstudio-server-rhel-0.99.896-x86_64.rpm 可能有最新版，請 Check官網

```
wget https://download2.rstudio.org/rstudio-server-rhel-0.99.896-x86_64.rpm
sudo yum install --nogpgcheck rstudio-server-rhel-0.99.896-x86_64.rpm
```

打開瀏覽器，輸入 http://localhost:8787/，就能進入 RStudio Server 了！

測完收工～

8.3 RHadoop MapReduce: easy word count

```
Debate<-readLines("https://raw.githubusercontent.com/yijutseng/BigDataCGUIM/master/RepDebateMiami.txt")
DebateSplit<-unlist(strsplit(tolower(Debate),split = ' |\\.|\\,|\\?'))
#table(DebateSplit)
```

```
DebateSplitDFS = to.dfs(DebateSplit)
result = mapreduce(
    input = DebateSplitDFS,
    map = function(.,v) keyval(v, 1),
    reduce = function(k,vv) keyval(k, sum(vv)))
head(result)
```

8.4 R + Spark

Chapter 9

軟體安裝

撰寫中

9.1 R

9.2 RStudio

作者資訊

曾意儒 Yi-Ju Tseng

長庚大學資訊管理學系 助理教授

個人網站

Lab: 數位健康實驗室