# 第 18 天

## 實用 R 語言技巧彙整（2）

在第 17 天與第 18 天的內容中我們彙整了一些解決常見問題的 R 語言技巧，提供給讀者參考，希望在您有需要的時候，這些函數或語法能夠幫上忙。

### 問題

我們希望可以讀取網頁資料。

#### 實用技巧

readLines() 函數可以協助我們將網頁的 html 原始檔讀入，並且以一個向量儲存，向量中的每個元素就代表著每一行 html 原始檔。我們以 [IMDb](http://www.imdb.com/) 網路電影資料庫為例選取一部電影[樂來越愛你](http://www.imdb.com/title/tt3783958/)來示範。

> url <- "http://www.imdb.com/title/tt3783958/"  
> lalaland <- readLines(url)  
> class(lalaland)  
[1] "character"  
> lalaland[10:19]  
 [1] "<!DOCTYPE html>"   
 [2] "<html"   
 [3] "xmlns:og=\"http://ogp.me/ns#\""   
 [4] "xmlns:fb=\"http://www.facebook.com/2008/fbml\">"   
 [5] " <head>"   
 [6] " <meta charset=\"utf-8\">"   
 [7] " <meta http-equiv=\"X-UA-Compatible\" content=\"IE=edge\">"   
 [8] ""   
 [9] " <meta name=\"apple-itunes-app\" content=\"app-id=342792525, app-argument=imdb:///title/tt3783958?src=mdot\">"  
[10] " "

在原始檔中如果有空行，也會原封不動被 readLines() 函數讀入，如果你檢視網頁的原始檔（在網頁按右鍵，點選**檢視網頁原始檔**），將會發現它跟向量中的內容完全相同。



圖 18-1 檢視網頁原始檔

透過 readLines() 函數讀入的 html 原始檔基本上就是一個文字向量，如果我們想要從中萃取我們想要的資訊（像是：電影片名、電影片長與電影類型等），就必須得自行想辦法利用文字函數與正規表示式（Regular Expression）來萃取，而正規表示式的使用對於初學者是有難度的，因此我們要使用 rvest 套件來進行讀取網頁 html 原始檔與萃取資訊的工作。

> install.packages("rvest") # 安裝 rvest 套件  
> library(rvest) # 載入 rvest 套件  
Loading required package: xml2  
>   
> url <- "http://www.imdb.com/title/tt3783958/"  
> lalaland <- read\_html(url)  
> class(lalaland)  
[1] "xml\_document" "xml\_node"   
> mode(lalaland)  
[1] "list"

透過 rvest 套件提供的 read\_html() 函數讀取網頁的 html 原始檔，我們會得到一個被命名為 "xml\_document" "xml\_node" 的清單物件，這是跟 readLines() 函數最大的差別。針對這個清單物件，我們就可以透過 CSS 選擇器（網頁資料中標註資料所在路徑的方法），把這個頁面重要的資訊萃取出來。舉例來說：評分。

> install.packages("rvest") # 安裝 rvest 套件  
> library(rvest) # 載入 rvest 套件  
Loading required package: xml2  
>   
> url <- "http://www.imdb.com/title/tt3783958/"  
> lalaland <- url %>%  
+ read\_html()  
> rating <- lalaland %>%  
+ html\_nodes(css = "strong span") %>%  
+ html\_text() %>%  
+ as.numeric()  
> rating  
[1] 8.2

透過類似的方式，我們可以將[樂來越愛你](http://www.imdb.com/title/tt3783958/)這個頁面的重要資訊都萃取出來，並且存在一個清單中。

> install.packages("rvest") # 安裝 rvest 套件  
> library(rvest) # 載入 rvest 套件  
Loading required package: xml2  
>   
> url <- "http://www.imdb.com/title/tt3783958/"  
> lalaland <- url %>%  
+ read\_html()  
>  
> # 不同資料的 CSS 選擇器路徑  
> title\_css <- "h1"  
> rating\_css <- "strong span"  
> time\_css <- "#title-overview-widget time"  
> type\_css <- ".subtext .itemprop"  
> in\_theater\_date\_css <- ".subtext a~ .ghost+ a"  
> cast\_css <- ".itemprop .itemprop"  
> # 準備一個空的清單存放資料  
> lalaland\_info <- list()  
>   
> infos <- c(title\_css, time\_css, type\_css, in\_theater\_date\_css, cast\_css)  
> for (i in 1:length(infos)) {  
+ info <- lalaland %>%  
+ html\_nodes(css = infos[i]) %>%  
+ html\_text()  
+ lalaland\_info[[i]] <- info  
+ }  
> names(lalaland\_info) <- c("title", "time", "type", "in\_theater\_date", "cast")  
>   
> rating <- lalaland %>%  
+ html\_nodes(css = "strong span") %>%  
+ html\_text() %>%  
+ as.numeric()  
> lalaland\_info$rating <- rating  
> lalaland\_info  
$title  
[1] "The leading information resource for the entertainment industry"  
[2] "La La Land (2016) "   
  
$time  
[1] "\n 2h 8min\n "  
  
$type  
[1] "Comedy" "Drama" "Music"   
  
$in\_theater\_date  
[1] "9 December 2016 (Taiwan)\n"  
  
$cast  
 [1] "Ryan Gosling" "Emma Stone" "Amiée Conn"   
 [4] "Terry Walters" "Thom Shelton" "Cinda Adams"   
 [7] "Callie Hernandez" "Jessica Rothe" "Sonoya Mizuno"   
[10] "Rosemarie DeWitt" "J.K. Simmons" "Claudine Claudio"  
[13] "Jason Fuchs" "D.A. Wallach" "Trevor Lissauer"   
  
$rating  
[1] 8.2

我們已經成功將資料萃取出來存放到 lalaland\_info 這個清單中，觀察一下後發現電影片名、電影片長與上映日期有一些多餘的空格或換行符號（\n），於是最後我們再加入一點清理資料的程式。

> install.packages("rvest") # 安裝 rvest 套件  
> library(rvest) # 載入 rvest 套件   
Loading required package: xml2  
>   
> url <- "http://www.imdb.com/title/tt3783958/"  
> lalaland <- url %>%  
+ read\_html()  
>   
> # 不同資料的 CSS 選擇器路徑  
> title\_css <- "h1"  
> rating\_css <- "strong span"  
> time\_css <- "#title-overview-widget time"  
> type\_css <- ".subtext .itemprop"  
> in\_theater\_date\_css <- ".subtext a~ .ghost+ a"  
> cast\_css <- ".itemprop .itemprop"  
> # 準備一個空的清單存放資料  
> lalaland\_info <- list()  
>   
> infos <- c(title\_css, time\_css, type\_css, in\_theater\_date\_css, cast\_css)  
> for (i in 1:length(infos)) {  
+ info <- lalaland %>%  
+ html\_nodes(css = infos[i]) %>%  
+ html\_text()  
+ lalaland\_info[[i]] <- info  
+ }  
> names(lalaland\_info) <- c("title", "time", "type", "in\_theater\_date", "cast")  
>   
> rating <- lalaland %>%  
+ html\_nodes(css = "strong span") %>%  
+ html\_text() %>%  
+ as.numeric()  
> lalaland\_info$rating <- rating  
> # 清理電影片名  
> title\_end\_index <- regexpr(pattern = ")", lalaland\_info$title[2])  
> lalaland\_info$title <- lalaland\_info$title[2] %>%  
+ substr(start = 1, stop = title\_end\_index)  
> # 清理電影片長  
> lalaland\_info$time <- gsub(pattern = "\n\\s+", lalaland\_info$time, replacement = "")  
> # 清理上映日期  
> lalaland\_info$in\_theater\_date <- gsub(pattern = "\n", lalaland\_info$in\_theater\_date, replacement = "")  
> # 檢視清理過後的清單  
> lalaland\_info  
$title  
[1] "La La Land (2016)"  
  
$time  
[1] "2h 8min"  
  
$type  
[1] "Comedy" "Drama" "Music"   
  
$in\_theater\_date  
[1] "9 December 2016 (Taiwan)"  
  
$cast  
 [1] "Ryan Gosling" "Emma Stone" "Amiée Conn"   
 [4] "Terry Walters" "Thom Shelton" "Cinda Adams"   
 [7] "Callie Hernandez" "Jessica Rothe" "Sonoya Mizuno"   
[10] "Rosemarie DeWitt" "J.K. Simmons" "Claudine Claudio"  
[13] "Jason Fuchs" "D.A. Wallach" "Trevor Lissauer"   
  
$rating  
[1] 8.2

最後我們可以將萃取資料與清理資料包裝成一個函數 IMDb\_crawler()，讓使用者只要輸入不同的電影網址，就可以回傳包含電影片名、電影片長、電影類型、上映日期、演員陣容與評等這五項資訊。

install.packages("rvest")  
library(rvest)  
  
IMDb\_crawler <- function(url) {  
 html\_doc <- url %>%  
 read\_html()  
  
 # 不同資料的 CSS 選擇器路徑  
 title\_css <- "h1"  
 rating\_css <- "strong span"  
 time\_css <- "#title-overview-widget time"  
 type\_css <- ".subtext .itemprop"  
 in\_theater\_date\_css <- ".subtext a~ .ghost+ a"  
 cast\_css <- ".itemprop .itemprop"  
 # 準備一個空的清單存放資料  
 movie\_info <- list()  
  
 infos <- c(title\_css, time\_css, type\_css, in\_theater\_date\_css, cast\_css)  
 for (i in 1:length(infos)) {  
 info <- html\_doc %>%  
 html\_nodes(css = infos[i]) %>%  
 html\_text()  
 movie\_info[[i]] <- info  
 }  
 names(movie\_info) <- c("title", "time", "type", "in\_theater\_date", "cast")  
  
 rating <- html\_doc %>%  
 html\_nodes(css = "strong span") %>%  
 html\_text() %>%  
 as.numeric()  
 movie\_info$rating <- rating  
 # 清理電影片名  
 title\_end\_index <- regexpr(pattern = ")", movie\_info$title[2])  
 movie\_info$title <- movie\_info$title[2] %>%  
 substr(start = 1, stop = title\_end\_index)  
 # 清理電影片長  
 movie\_info$time <- gsub(pattern = "\n\\s+", movie\_info$time, replacement = "")  
 # 清理上映日期  
 movie\_info$in\_theater\_date <- gsub(pattern = "\n", movie\_info$in\_theater\_date, replacement = "")  
 # 回傳清理好的清單  
 return(movie\_info)  
}

> # 呼叫函數  
> wonder\_woman <- IMDb\_crawler(url = "http://www.imdb.com/title/tt0451279/")  
> wonder\_woman  
$title  
[1] "Wonder Woman (2017)"  
  
$time  
[1] "2h 21min"  
  
$type  
[1] "Action" "Adventure" "Fantasy"   
  
$in\_theater\_date  
[1] "30 May 2017 (Taiwan)"  
  
$cast  
 [1] "Gal Gadot" "Chris Pine" "Connie Nielsen"   
 [4] "Robin Wright" "Danny Huston" "David Thewlis"   
 [7] "Saïd Taghmaoui" "Ewen Bremner" "Eugene Brave Rock"   
[10] "Lucy Davis" "Elena Anaya" "Lilly Aspell"   
[13] "Lisa Loven Kongsli" "Ann Wolfe" "Ann Ogbomo"   
  
$rating  
[1] 8.1

### 問題

我們希望可以將資料框依輸入的比例切割為訓練（Train）與測試（Test）資料。

#### 實用技巧

在分割訓練與測試資料之前，我們要先針對資料做洗牌（shuffle），透過在[第 17 天 實用 R 語言技巧彙整](chapter17.md)使用的 sample() 函數可以完成對資料洗牌的工作，我們接著以內建的 iris 資料示範。

> n <- nrow(iris) # 計算 iris 有幾列資料  
> set.seed(123)  
> iris\_shuffled <- iris[sample(n), ]  
> head(iris) # 原本的 iris 前六列  
 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  
6 5.4 3.9 1.7 0.4 setosa  
> head(iris\_shuffled) # 洗牌後的 iris 前六列  
 Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
44 5.0 3.5 1.6 0.6 setosa  
118 7.7 3.8 6.7 2.2 virginica  
61 5.0 2.0 3.5 1.0 versicolor  
130 7.2 3.0 5.8 1.6 virginica  
138 6.4 3.1 5.5 1.8 virginica  
7 4.6 3.4 1.4 0.3 setosa

觀察列數的索引值就可以發現我們已經完成了對資料的洗牌，接著就是要找一個切點將資料分成上下兩部分，上面是訓練資料，下面是測試資料。我們先以 70% 訓練資料、30% 測試資料這個標準的比例分配來示範。

> n <- nrow(iris) # 計算 iris 有幾列資料  
> set.seed(123)  
> iris\_shuffled <- iris[sample(n), ]  
> train\_test\_cut <- round(0.7 \* n)  
> iris\_train <- iris\_shuffled[1:train\_test\_cut, ]  
> iris\_test <- iris\_shuffled[(train\_test\_cut + 1):n, ]  
> nrow(iris\_train)  
[1] 105  
> nrow(iris\_test)  
[1] 45

接下來我們可以把這段程式包裝成一個函數 train\_test\_split()，使用者只要將資料輸入，指定一個訓練資料的比例，就可以得到訓練資料與測試資料的輸出。

# 自訂函數  
train\_test\_split <- function(x, train\_percentage) {  
 n <- nrow(x) # 計算輸入有幾列資料  
 data\_shuffled <- x[sample(n), ]  
 train\_test\_cut <- round(train\_percentage \* n)  
 train\_data <- data\_shuffled[1:train\_test\_cut, ]  
 test\_data <- data\_shuffled[(train\_test\_cut + 1):n, ]  
 outputs <- list(train = train\_data, test = test\_data)  
 return(outputs)  
}

> # 呼叫函數  
> iris\_train\_test <- train\_test\_split(iris, train\_percentage = 0.8)  
> iris\_train <- iris\_train\_test$train  
> iris\_test <- iris\_train\_test$test  
> nrow(iris\_train)  
[1] 120  
> nrow(iris\_test)  
[1] 30

### 問題

我們希望可以建立一個線性迴歸的模型。

#### 實用技巧

使用 lm() 函數；我們用[世界第一簡單統計學迴歸分析篇](http://www.books.com.tw/products/0010479438)裡面的一組氣溫與冰紅茶銷量的資料來示範。

> temperatures <- c(29, 28, 34, 31, 25, 29, 32, 31, 24, 33, 25, 31, 26, 30)  
> iced\_tea\_sales <- c(77, 62, 93, 84, 59, 64, 80, 75, 58, 91, 51, 73, 65, 84)  
>   
> lm\_fit <- lm(iced\_tea\_sales ~ temperatures)  
>   
> # 印出係數  
> lm\_fit$coefficients[2]  
temperatures   
 3.737885   
>   
> # 印出截距  
> lm\_fit$coefficients[1]  
(Intercept)   
 -36.36123

建立線性迴歸模型之後，冰紅茶店的老闆就可以量測氣溫，藉此來預測冰紅茶銷量，更精準地掌握原料的管理，預測新資料的時候我們要使用 predict() 函數。

> temperatures <- c(29, 28, 34, 31, 25, 29, 32, 31, 24, 33, 25, 31, 26, 30)  
> iced\_tea\_sales <- c(77, 62, 93, 84, 59, 64, 80, 75, 58, 91, 51, 73, 65, 84)  
>   
> lm\_fit <- lm(iced\_tea\_sales ~ temperatures)  
>   
> # 假設明日的氣溫預測為 30 度  
> to\_be\_predicted <- data.frame(temperatures = 30)  
> predicted\_sales <- predict(lm\_fit, newdata = to\_be\_predicted)  
>   
> # 預測的冰紅茶銷量  
> predicted\_sales  
 1   
75.77533

如果明日的氣溫預測為 30 度，那麼這個線性迴歸模型預測明日的冰紅茶銷量約為 76 杯。我們可以將用 Base Plotting System 畫出來這條直線與預測點：

temperatures <- c(29, 28, 34, 31, 25, 29, 32, 31, 24, 33, 25, 31, 26, 30)  
iced\_tea\_sales <- c(77, 62, 93, 84, 59, 64, 80, 75, 58, 91, 51, 73, 65, 84)  
  
lm\_fit <- lm(iced\_tea\_sales ~ temperatures)  
  
# 假設明日的氣溫預測為 30 度  
to\_be\_predicted <- data.frame(temperatures = 30)  
predicted\_sales <- predict(lm\_fit, newdata = to\_be\_predicted)  
  
plot(iced\_tea\_sales ~ temperatures, bg = "blue", pch = 16)  
points(x = to\_be\_predicted$temperatures, y = predicted\_sales, col = "red", cex = 2, pch = 17)  
abline(reg = lm\_fit$coefficients, col = "blue", lwd = 4)

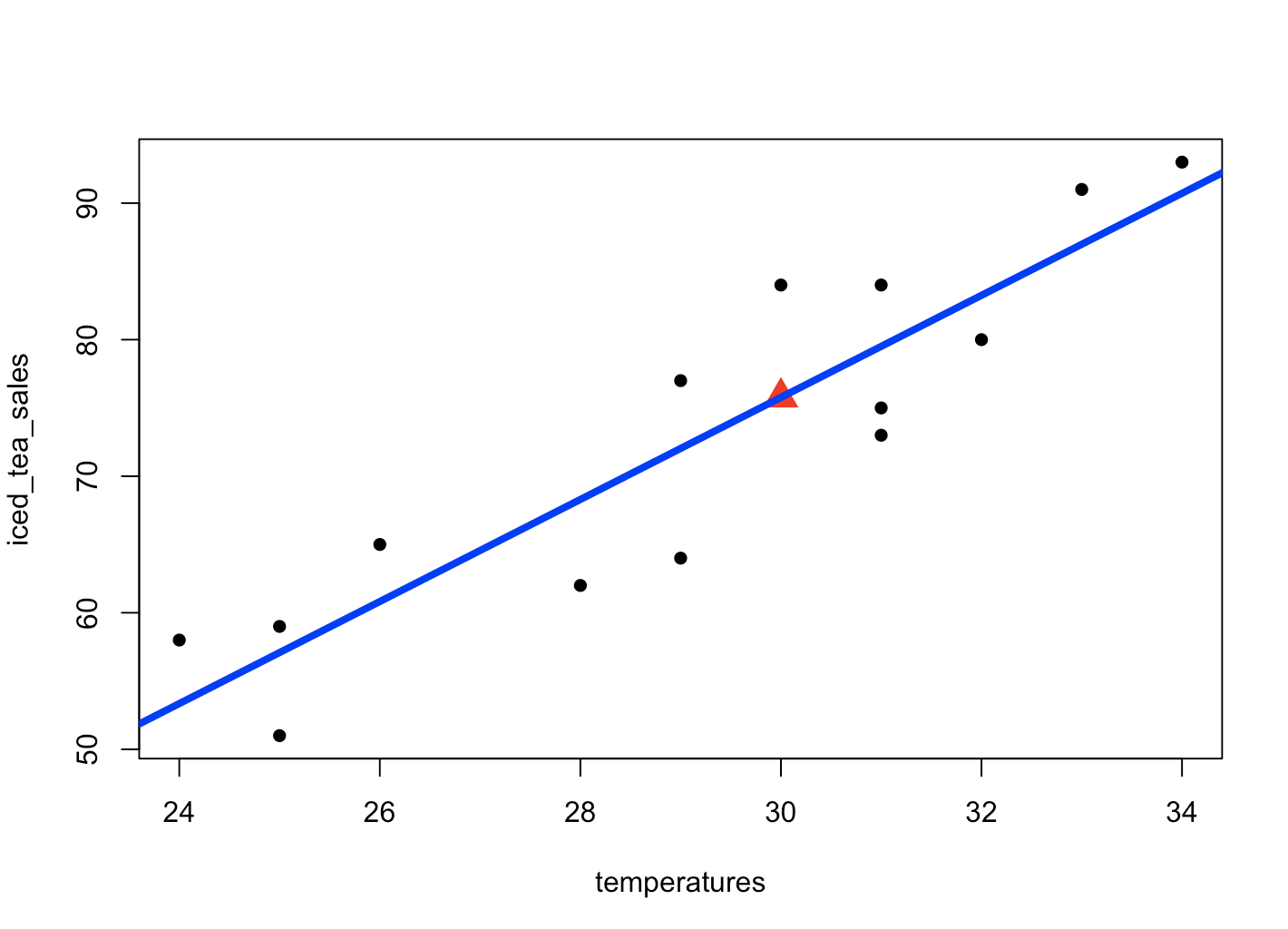


圖 18-2 線性迴歸模型與預測

### 問題

我們希望可以建立一個簡單的分類器。

#### 實用技巧

使用 rpart 套件的 rpart() 函數可以建立一個決策樹分類器，決策樹分類器是可以處理多元分類問題的演算法，它可以同時處理連續型與類別型變數，也不需要對資料預先進行太多處理。我們利用內建資料 iris 來示範如何建立一個決策樹分類器來分類鳶尾花的品種，使用分類器之前，我們會利用前面建立的 train\_test\_split() 函數來協助我們切分訓練與測試資料。

install.packages("rpart")  
library(rpart)  
  
# 自訂函數  
train\_test\_split <- function(x, train\_percentage) {  
 n <- nrow(x) # 計算輸入有幾列資料  
 data\_shuffled <- x[sample(n), ]  
 train\_test\_cut <- round(train\_percentage \* n)  
 train\_data <- data\_shuffled[1:train\_test\_cut, ]  
 test\_data <- data\_shuffled[(train\_test\_cut + 1):n, ]  
 outputs <- list(train = train\_data, test = test\_data)  
 return(outputs)  
}  
  
# 切分訓練與測試資料  
iris\_train\_test <- train\_test\_split(iris, train\_percentage = 0.7)  
iris\_train <- iris\_train\_test$train  
iris\_test <- iris\_train\_test$test  
  
# 建立分類器  
iris\_clf <- rpart(Species ~ ., data = iris\_train, method = "class")  
  
# 預測  
predicted <- predict(iris\_clf, iris\_test, type = "class")

接著我們比對 predicted 與 iris\_test$Species 來得知這個決策樹分類器的準確率為多少。利用 table() 可以建立混淆矩陣，在衡量分類器的表現時，第一步都是從建立混淆矩陣開始。

> install.packages("rpart")  
> library(rpart)  
>  
> # 自訂函數  
> train\_test\_split <- function(x, train\_percentage) {  
+ n <- nrow(x) # 計算輸入有幾列資料  
+ data\_shuffled <- x[sample(n), ]  
+ train\_test\_cut <- round(train\_percentage \* n)  
+ train\_data <- data\_shuffled[1:train\_test\_cut, ]  
+ test\_data <- data\_shuffled[(train\_test\_cut + 1):n, ]  
+ outputs <- list(train = train\_data, test = test\_data)  
+ return(outputs)  
+ }  
>   
> # 切分訓練與測試資料  
> iris\_train\_test <- train\_test\_split(iris, train\_percentage = 0.7)  
> iris\_train <- iris\_train\_test$train  
> iris\_test <- iris\_train\_test$test  
>   
> # 建立分類器  
> iris\_clf <- rpart(Species ~ ., data = iris\_train, method = "class")  
>   
> # 預測  
> predicted <- predict(iris\_clf, iris\_test, type = "class")  
>   
> # 績效  
> conf\_mat <- table(iris\_test$Species, predicted)  
> accuracy <- sum(diag(conf\_mat)) / sum(conf\_mat)  
> accuracy  
[1] 0.9555556

### 問題

我們希望可以利用 K-Means 演算法為資料點進行分群。

#### 實用技巧

使用 kmeans() 函數；我們使用內建資料 iris 示範。

> # 讀入鳶尾花資料  
> iris\_kmeans <- iris[, -5]  
>   
> # KMeans 演算法  
> kmeans\_fit <- kmeans(iris\_kmeans, nstart = 20, centers = 3)  
>   
> # 印出分群結果  
> kmeans\_fit$cluster  
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
 [34] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3  
 [67] 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
[100] 3 1 3 1 1 1 1 3 1 1 1 1 1 1 3 3 1 1 1 1 3 1 3 1 3 1 1 3 3 1 1 1 1  
[133] 1 3 1 1 1 1 3 1 1 1 3 1 1 1 3 1 1 3

接著我們使用 Total WSS（Total Within Cluster Sum of Squares）/Total SS（Total Sum of Squares），來衡量資料分群的績效，這個比例愈低表示績效愈好。

> # 讀入鳶尾花資料  
> iris\_kmeans <- iris[, -5]  
>   
> # KMeans 演算法  
> kmeans\_fit <- kmeans(iris\_kmeans, nstart = 20, centers = 3)  
> ratio <- kmeans\_fit$tot.withinss / kmeans\_fit$totss  
> ratio  
[1] 0.1157247

隨著 k 值的增加，K-Means 演算法的績效一定會愈來愈好，當 k 跟觀測值數目相同的時候，我們會必定會得到一個**組間差異最大，組內差異最小**的結果，但這不是我們希望的資料點分群，這時會畫一個陡坡圖（Scree Plot）來選擇一個適合的 k。

# 讀入鳶尾花資料  
iris\_kmeans <- iris[, -5]  
  
# 迴圈  
ratio <- rep(NA, times = 10)  
for (k in 2:length(ratio)) {  
 kmeans\_fit <- kmeans(iris\_kmeans, centers = k, nstart = 20)  
 ratio[k] <- kmeans\_fit$tot.withinss / kmeans\_fit$betweenss  
}  
plot(ratio, type = "b", xlab = "k")

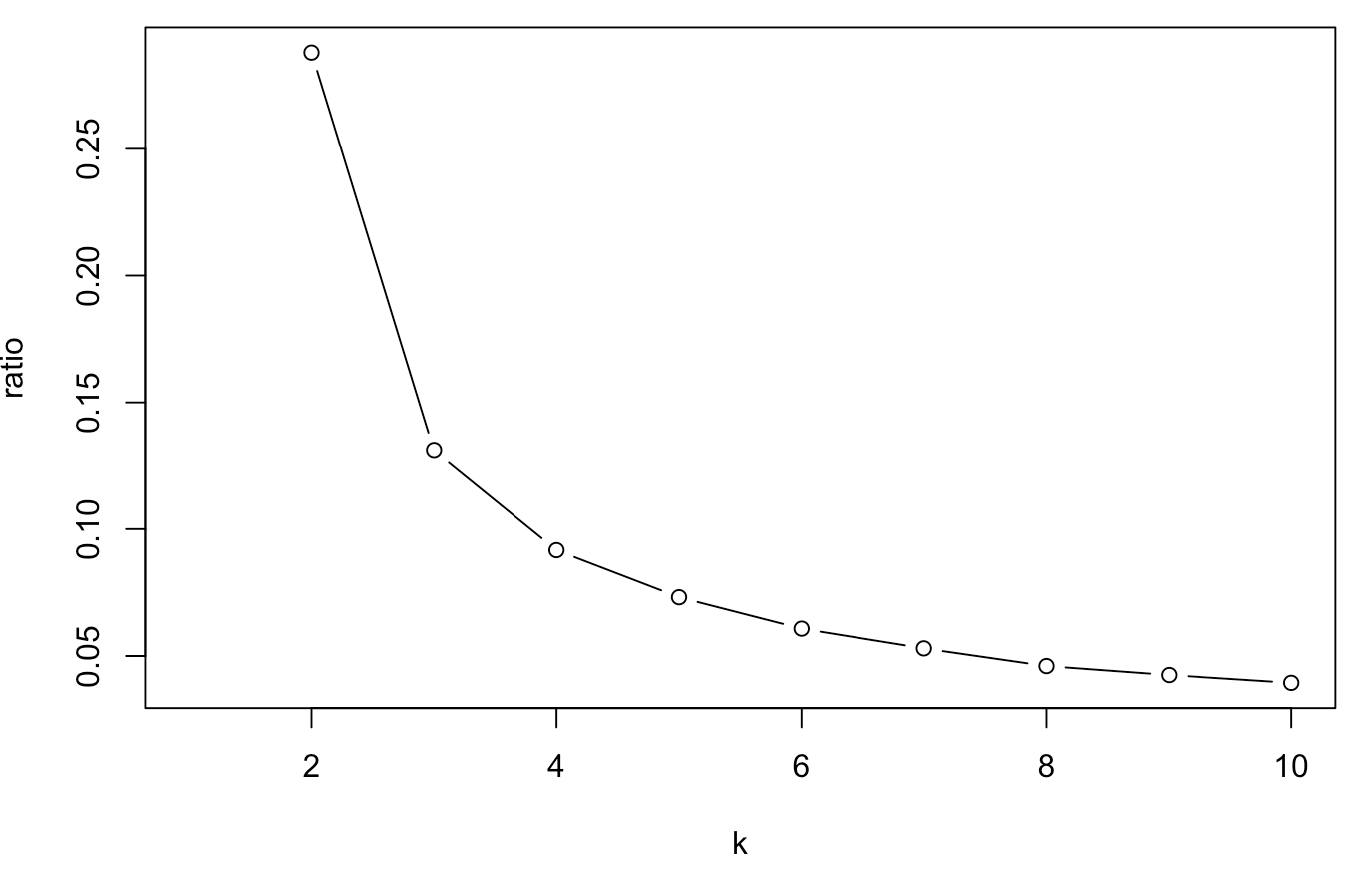


圖 18-3 選擇合適的 k

在 k = 2 或者 k = 3 的時候斜率開始變小，這意味著 Total WSS/Total SS 比例下降的效率趨緩，所以理想的資料點分群可能是二群或三群。

### 延伸資訊

* [世界第一簡單統計學迴歸分析篇](http://www.books.com.tw/products/0010479438)