

數位創新 · 行銷科技



## R語法使用解析 Part One

March 2024

國立中興大學 行銷系 教授  
澳洲 科廷大學 行銷博士  
美國 南加州大學 電腦碩士

曹修源

Hsiu-Yuan Tsao (Jody) Ph.D

## 〈曹修源 教授〉簡介

Dr.  
TSAO



### 〈學歷〉

- 澳洲**柯廷大學** 柯廷商學院 **行銷博士**
- 澳洲**雪梨大學** 商學院 **訪問學者**
- 美國**南加州大學** 電腦科學研究所 **電腦科學碩士**
- 東吳大學 電子計算機科學系 商學士

### 〈經歷〉

- 國立中興大學** 行銷系 教授
- 淡江大學 企管系 副教授
- 德明財經科技大學 資管系 副教授
- 明新科技大學 資管系 專任助理教授
- 澳洲國立柯廷科技大學 廷商學院 行銷系研究助理
- 明新技術 資訊中心教育訓練組 資工系兼任講師
- IBM台灣分公司**
- 東吳大學 電子計算機科學系 助教
- 中華民國陸軍總部資訊中心C3I 系統開發

### 〈榮譽〉

- 東吳大學 電算系年度畢業生群育獎
- 管理協會暨國際管理協會 商業類 Best Paper Award
- 中華商管科技學會年會暨學術研討會 行銷類 優勝論文獎
- 技職實務個案撰寫競賽 優勝
- (ANZMA) Conference 品牌與品牌管理類 最佳論文獎
- 指導3位研究生獲2016、2018、2019**崇越論文大賞**
- 指導學生獲**輿情分析比賽 冠軍**



### 〈研究專長〉

- 行銷科學：行銷分析、品牌管理、網路行銷、創新與新產品開發
- 資料科學：輿情分析、人工智慧於行銷上之應用

### 〈論文著作〉

From mining to meaning: How B2B marketers can leverage text to inform strategy  
Estimating numerical scale ratings from text-based service reviews

### 〈教科書編撰〉

**網路與數位行銷** (2019/高立圖書出版)



### 〈學術期刊發表論文〉

- Industrial Marketing Management ( SSCI )
- Journal of Service Management ( SSCI )
- Marketing Letters ( SSCI )
- European Journal of Marketing ( SSCI )
- Business Horizons ( SSCI )
- International Journal of Market Research ( SSCI )
- Journal of Brand Management ( SSCI )
- Journal of Consumer Behaviour ( SSC )
- Journal of The Operational Research Society ( SSCI )
- OMEGA-International Journal of Management Science ( SSCI )
- Asia Pacific Journal of Marketing and Logistics



## 讀取文字檔並增加行數的欄位

```
tenlong <- read.xlsx(filename, sheetIndex = 1) #讀取excel檔案第一個工作表
```

```
colnames(tenlong) <- c("text") #更改欄位名稱
```

```
tenlong <- tenlong %>%
```

```
  mutate(line=c(1:nrow(tenlong))) %>% #mutate():用於生成新欄位line#  
  relocate(line) #relocate:把生成新欄位line#欄位位置移到第一個
```

```
tenlong <- tenlong[which(nchar(tenlong$text) > 2),] # (nchar(tenlong$text) 計算tenlong$text這位欄位值的字元長度
```

```
  # tenlong[ row, column]
```

```
  # which: 篩選符合條件的值 就此為例是篩選出$text欄位字串
```

```
  # 有中文字且非空字串
```

```
colnames(tenlong) <- c("line","text") #更改欄位名稱
```

# 以ChatGPT協助學習函數使用 以gsub為例



You

請給我一個完整可執行的例子說明 R語言的gsub語法的使用並以繁體中文回覆



ChatGPT

當然可以。這裡有一個R語言中`gsub`函數的使用示例，這個例子將展示如何在一段文本中尋找並替換指定的字符串。

假設我們有一段文本，想要將文本中的所有"老師"二字替換為"教授"，下面是如何使用`gsub`函數來完成這個任務的完整R語言代碼：

```
R Copy code

# 創建一個包含原始文本的變量
original_text <- "這位老師非常專業。每位學生都非常尊敬這位老師。"

# 使用gsub函數替換文本
# 將"老師"替換為"教授"
modified_text <- gsub("老師", "教授", original_text)

# 打印修改後的文本
print(modified_text)
```

在這個例子中，`gsub`函數將`original\_text`變量中所有出現的"老師"替換成了"教授"。然後，修改後的文本會被儲存在`modified\_text`變量中，並通過`print`函數輸出到控制台。這樣，原始文本中的每一處"老師"都會被替換成"教授"。

🔊 📄 ↺ 🔍



# 以ChatGPT協助學習函數使用 以tidyverse (文字處理神器)函式庫為例

```
tokens <- AZDF02 %>%
```

```
  unnest_tokens(word, text, token = chi_tokenizerN) %>%
```

```
  anti_join(stopwords) unnest_tokens
```

```
  jieba_tokenizerN = worker(type="tag")
```

You  
請給我一個完整可執行的例子說明 R語言的語法unnest\_tokens的使用並以繁體中文回覆

ChatGPT  
'unnest\_tokens' 是 'tidytext' 包中的一個函數，用於將文本數據框中的文本列拆分成單個單詞或符號，這對於文本分析和自然語言處理任務非常有用。首先，您需要安裝和加載 'tidytext' 和 'dplyr' 包，因為 'unnest\_tokens' 函數通常與 'dplyr' 的數據處理功能一起使用。

以下是一個完整的例子，說明如何使用 'unnest\_tokens' 函數：

1. 安裝並加載所需的包。
2. 創建一個包含文本數據的數據框。
3. 使用 'unnest\_tokens' 函數將文本列拆分成單個單詞。

```
R
# 安裝和加載所需的包
if (!requireNamespace("tidytext", quietly = TRUE)) install.packages("tidytext")
if (!requireNamespace("dplyr", quietly = TRUE)) install.packages("dplyr")

library(tidytext)
library(dplyr)

# 創建一個包含文本數據的數據框
text_data <- data.frame(
  id = 1:2,
  text = c("R語言是統計分析和圖形的一種強大工具。", "使用R可以進行數據處理、分析和可視化。")
)

# 使用unnest_tokens拆分文本列成單個單詞
tidy_text <- text_data %>%
  unnest_tokens(word, text)

# 查看結果
print(tidy_text)
```

在這個例子中，'text\_data' 是一個包含兩條文本記錄的數據框。'unnest\_tokens' 函數將 'text' 列中的每個文本拆分成單個單詞，並創建一個新的列 'word' 來儲存這些單詞。'unnest\_tokens' 的第一個參數是新列的名稱（在此例中為 'word'），第二個參數是要拆分的文本列的名稱（在此例中為 'text'）。執行後，你將得到一個新的數據框 'tidy\_text'，其中包含分解後的單詞。

🔍 📄 🔄 📌

```
#-----tokenizer-----
```

```
chi_tokenizerN <- function(t) {
```

```
  lapply(t, function(x) {
```

```
    tokens <- segment(x, jieba_tokenizerN)
```

```
    tokens <- get_noun(tokens[nchar(tokens) > 1])
```

```
  return(tokens) })}
```

```
#----- get_noun -----
```

```
get_noun = function(x){
```

```
  stopifnot(inherits(x,"character")) index = names(x) %in%
```

```
  c("n","nr","nr1","nr2","nrj","nrf","ns","nsf","nt","nz",
```

```
  "nl","ng", "a","ad","an","ag","al",
```

```
  "v","vh","vg","vd","vn","vi*","vq*", "d","x","b")
```

```
  x[index]}
```



# 以ChatGPT協助學習函數使用 以join系統指令為例



You

User

請舉個子說明 R 語言中 inner\_join, left\_join, right\_join, outer\_join 的差別 請以繁體中文回覆

示例

假設我們有兩個資料框 `df1` 和 `df2`：

```
r
df1 <- data.frame(ID = c(1, 2, 3), Name = c("Alice", "Bob", "Charlie"))
df2 <- data.frame(ID = c(2, 3, 4), Age = c(25, 30, 35))
```

使用不同的 join 函式將會產生不同的結果：

```
r
library(dplyr)

# Inner Join
inner_join(df1, df2, by = "ID")

# Left Join
left_join(df1, df2, by = "ID")

# Right Join
right_join(df1, df2, by = "ID")

# Full Join
full_join(df1, df2, by = "ID")
```



ChatGPT

在R語言中，`inner\_join`、`left\_join`、`right\_join` 和 `outer\_join` 是用於合併兩個資料框（data frame）的函式，這些函式來自於 `dplyr` 套件。這些函式的主要差異在於它們如何處理不匹配的觀測值。

**inner\_join**

`inner\_join` 會返回兩個資料框中所有匹配的觀測值。如果某個觀測值在一個資料框中但不在另一個資料框中，則該觀測值不會出現在最終結果中。

**left\_join**

`left\_join` 會返回左側資料框的所有觀測值，並加上與右側資料框匹配的觀測值。如果左側資料框的某個觀測值在右側資料框中沒有匹配，則在最終結果中該觀測值的右側資料框列將會是 NA。

**right\_join**

`right\_join` 與 `left\_join` 相反，它返回右側資料框的所有觀測值，並加上與左側資料框匹配的觀測值。如果右側資料框的某個觀測值在左側資料框中沒有匹配，則在最終結果中該觀測值的左側資料框列將會是 NA。

**full\_join 或 outer\_join**

`full\_join`（也稱為 `outer\_join`）會返回兩個資料框中所有的觀測值。如果某個觀測值只出現在一個資料框中，則在最終結果中該觀測值的另一個資料框的列將會是 NA。



# 以ChatGPT協助學習函數使用 join 系統指令範例及圖示

ID	Name
1	Alice
2	Bob
3	Charlie

ID	Age
2	25
3	30
4	35

## ANTI\_JOIN ?

```
> anti_join(df1, df2, by = "ID")
  ID Name
1  1 Alice
```

```
> # Left Join
> left_join(df1, df2, by = "ID")
```

	ID	Name	Age
1	1	Alice	NA
2	2	Bob	25
3	3	Charlie	30

```
> # Right Join
> right_join(df1, df2, by = "ID")
```

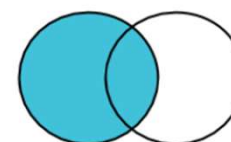
	ID	Name	Age
1	2	Bob	25
2	3	Charlie	30
3	4	<NA>	35

```
> # Inner Join
> inner_join(df1, df2, by = "ID")
```

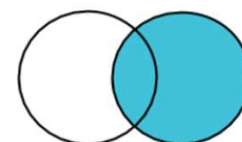
	ID	Name	Age
1	2	Bob	25
2	3	Charlie	30

```
> # Full Join
> full_join(df1, df2, by = "ID")
```

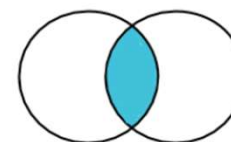
	ID	Name	Age
1	1	Alice	NA
2	2	Bob	25
3	3	Charlie	30
4	4	<NA>	35



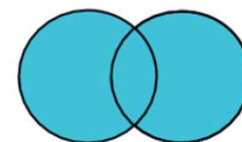
Left Join



Right Join



Inner Join



Full Outer Join

# 以ChatGPT協助學習函數使用 以系統lapply指令為例



You

舉例說明R語言中lapply指令的用法請以繁體中文回覆

< 2 / 2 >



ChatGPT

在R語言中，`lapply()` 函數是一種非常有用的指令，它允許您對列表(list)或向量(vector)的每個元素應用一個函數，並返回一個列表作為結果。這在數據分析和處理中非常常見，尤其是當您需要對數據集中的多個元素進行相同的操作時。

**lapply()的基本用法：**

```
lapply(X, FUN, ...)
```

Copy code

- `X`：一個向量、列表或表達式對象。
- `FUN`：要應用於`X`中每個元素的函數。
- `...`：額外的參數或選項可以傳遞給`FUN`。

.1.

範例：

假設您有一個數值列表，想要計算每個數字的平方根：

```
r
# 創建一個數值列表
numbers <- list(4, 9, 16)

# 使用lapply計算每個數字的平方根
result <- lapply(numbers, sqrt)

# 輸出結果
print(result)
```

Copy code

這段代碼將對`numbers`列表中的每個元素應用`sqrt()`函數（計算平方根的函數），並將結果存儲在`result`列表中。輸出將會是每個數字的平方根值的列表。

自行練習請ChatGPT協助說明

指令: filter , unique的用法



# 以ChatGPT協助學習函數使用 以groupby, summarise系統指令為例

## ChatGPT

在R語言中，`group\_by` 和 `summarise` (或 `summarize`) 指令通常與 `dplyr` 套件一起使用，以便於對數據進行分組處理和摘要統計。這對於數據分析非常有用，尤其是當你需要對特定類別或分組進行匯總或其他計算時。

首先，確保你已經安裝並載入了 `dplyr` 套件：

| :

我們有一個數據框 `df`，包含了學生的分數和班級信息：

```
r                                                                    Copy code
df <- data.frame(
  class = c("A", "A", "B", "B", "C", "C"),
  score = c(80, 90, 85, 95, 75, 85)
)
```

我們想要計算每個班級的平均分數：

```
r                                                                    Copy code
# 按班級分組，然後計算每個班級的平均分數
average_scores <- df %>%
  group_by(class) %>%
  summarise(average_score = mean(score))

# 輸出結果
print(average_scores)
```

這段代碼將 `df` 數據框按 `class` 變量分組，然後使用 `summarise` 計算每個班級的平均分數（`mean(score)`）。結果將是一個新的數據框，包含每個班級及其對應的平均分數。

## 以ChatGPT協助學習函數使用 以ggplot 繪圖為例

圖形種類	繪圖函數
散佈圖	<code>geom_point()</code>
線圖	<code>geom_line()</code>
直方圖	<code>geom_histogram()</code>
盒鬚圖	<code>geom_boxplot()</code>
長條圖	<code>geom_bar()</code>



The End