

# 程式人雜誌 -- 2014 年 2 月號 (開放公益出版品)

- 講題分享 - 用 R 進行中文 text Mining (作者:陳嘉葳@Taiwan R User Group)

## 講題分享 - 用 R 進行中文 text Mining (作者:陳嘉葳@Taiwan R User Group)

本文使用的分析方法，目前僅能在Windows上測試成功。

### 簡介

現今網路上有大量文字資料,例如 ptt, facebook, 或 mobile01等討論網站上都有大量文字留言, 由於這些資料繁多雜亂, 我們可藉由文字探勘技術萃取出有用的訊息, 讓人們有效率掌握這些網路文字所提供的訊息。而R語言是一款非常適合資料分析的工具,有一系列文字探勘的套件可供使用,本文將簡單介紹中文文字探勘套件的使用方法。

### 安裝需要工具

我的環境: Windows 7 + R 版本 2.15.3 + RStudio 0.98.484

安裝以下套件

```
install.packages("rJava")
install.packages("Rwordseg", repos="http://R-Forge.R-project.org")
install.packages("tm")
install.packages("tmcn", repos="http://R-Forge.R-project.org", type="source")
install.packages("wordcloud")
install.packages("XML")
install.packages("RCurl")
```

Windows上安裝rJava的注意事項:

- 將 jvm.dll 加到環境變數PATH之中
- 注意 java 的版本(32-bit or 64-bit)必須要和R一致 (Windows binary)之後再安裝，以避免

### 抓取ptt笨版文章列表

我們以分析ptt笨版文章為範例, 首先到ptt web 版抓取文章的url連結。我們利用 RCurl套件的 getURL和 XML套件的 htmlParseApply 抓取笨版文章列表的 html網頁, 再用xpathSApply 去擷取出所有文章的url連結並儲存。或是可以到[已下載ptt笨版文章](#)

```
library(XML)
library(RCurl)
```

```

data <- list()

for( i in 1058:1118) {
  tmp <- paste(i, '.html', sep='')
  url <- paste('www.ptt.cc/bbs/StupidClown/index', tmp, sep='')
  html <- htmlParse(getURL(url))
  url.list <- xpathSApply(html, "//div[@class='title']/a[@href]", xmlAttrs)
  data <- rbind(data, paste('www.ptt.cc', url.list, sep=''))
}
data <- unlist(data)

```

下載列表中的笨版文章之後，接著才開始利用所有文章的url連結去抓所有文章的html網頁，並用xpathSApply去解析出文章的內容並儲存。

```

getdoc <- function(line) {
  start <- regexpr('www', line)[1]
  end <- regexpr('html', line)[1]

  if(start != -1 & end != -1) {
    url <- substr(line, start, end+3)
    html <- htmlParse(getURL(url), encoding='UTF-8')
    doc <- xpathSApply(html, "//div[@id='main-content']", xmlValue)
    name <- strsplit(url, '/')[[1]][4]
    write(doc, gsub('html', 'txt', name))
  }
}

```

開始下載文章

先用 setwd(填入資料夾位置) 這指令, 選定文件下載位置

或是用 getwd() 確定目前的工作資料夾位置

```
sapply(data, getdoc)
```

## 開始文字處理

首先載入以下text mining 套件

```

library(tm)
library(tmcn)

```

```
## # tmcn Version: 0.1-2

library(Rwordseg)

## Loading required package: rJava
## # Version: 0.2-1
```

匯入剛才抓完的笨版文章，doc 是儲存下載ptt文章的資料夾，這些文章變成我們分析的語料庫。

```
d.corpus <- Corpus(DirSource("doc"), list(language = NA))
```

## 進行數據清理

1 清除標點符號, 數字

```
d.corpus <- tm_map(d.corpus, removePunctuation)
d.corpus <- tm_map(d.corpus, removeNumbers)
```

2 清除大小寫英文與數字

```
d.corpus <- tm_map(d.corpus, function(word) {
  gsub("[A-Za-z0-9]", "", word)
})
```

## 進行中文斷詞

首先，由於ptt有自己獨特的詞彙，例如發文不附沒圖、沒圖沒真相...等等，因此我們另外安裝 ptt 常用詞彙來協助斷詞。ptt常用詞彙可以從[搜狗細胞辭庫](#)下載。

```
words <- readLines("http://wubi.sogou.com/dict/download_txt.php?id=9182")
words <- toTrad(words)
insertWords(words)
```

接著，我們利用我們 Rwordseg套件裡的segmentCN來進行斷詞。Rwordseg是李艦所撰寫的R套件，利用rJava去連結java分詞工具ansj來進行斷詞。另外，斷詞後的詞彙有詞性，例如動詞、名詞、形容詞、介係詞等等，我們只挑出名詞來進行分析。

```
d.corpus <- tm_map(d.corpus[1:100], segmentCN, nature = TRUE)
d.corpus <- tm_map(d.corpus, function(sentence) {
  noun <- lapply(sentence, function(w) {
    w[names(w) == "n"]
  })
  unlist(noun)
```

```
})
d.corpus <- Corpus(VectorSource(d.corpus))
```

接著進行清除停用字符，停用字符指的是一些文章中常見的單字，但卻無法提供我們資訊的冗字。例如有些、以及、因此...等等字詞。

```
myStopWords <- c(stopwordsCN(), "編輯", "時間", "標題", "發信", "實業", "作者")
d.corpus <- tm_map(d.corpus, removeWords, myStopWords)
```

我們可以看看有哪些停用字符，這裡抽出前20個停用字符來看

```
head(myStopWords, 20)

## [1] "第二"      "一番"      "一直"      "一<U+4E2A>" "一些"      "<U+8BB8>多"
## [7] "种"        "有的是"    "也就是<U+8BF4>" "末"        "啊"        "阿"
## [13] "哎"        "哎呀"      "哎<U+54DF>" "唉"        "俺"        "俺<U+4EEC>"
## [19] "按"        "按照"
```

## 建立 TermDocumentMatrix

中文斷詞結束後，我們用矩陣將結果儲存起來。TermDocumentMatrix 指的是關鍵字為列，文件是行的矩陣。儲存的數字是關鍵字在這些文件中出現的次數。其中 wordLengths=c(2,Inf) 表示我們挑至少兩個字的詞。

```
tdm <- TermDocumentMatrix(d.corpus, control = list(wordLengths = c(2, Inf)))
```

我們可以看看TermDocumentMatrix裡面，前兩篇文章的前10個關鍵字

```
inspect(tdm[1:10, 1:2])

## A term-document matrix (10 terms, 2 documents)
##
## Non-/sparse entries: 3/17
## Sparsity           : 85%
## Maximal term length: 3
## Weighting          : term frequency (tf)
##
##          Docs
## Terms    M. 1384834727. A. 0CB. txt M. 1384838698. A. 957. txt
##  一生                0                0
##  一家                1                0
##  一家子              0                0
##  一線                0                0
```



```
## Weighting      : term frequency (tf)
##
##              Terms
## Docs          一生 一家
## M. 1384834727. A. 0CB. txt    0    1
## M. 1384838698. A. 957. txt    0    0
## M. 1384840050. A. 414. txt    0    0
## M. 1384840304. A. EF5. txt    0    0
## M. 1384842495. A. 5B8. txt    0    0
## M. 1384842609. A. A5B. txt    0    0
## M. 1384847473. A. 6A5. txt    0    0
## M. 1384847771. A. 729. txt    0    0
## M. 1384848469. A. AD8. txt    0    0
## M. 1384849471. A. B71. txt    0    0
```

可以用 `findFreqTerms` 看看在所有文件裡出現30次以上的關鍵字有哪些。

```
findFreqTerms(d.dtm, 30)
```

```
## [1] "同學" "百合" "老闆" "朋友" "東西" "時候" "閃光" "問卷" "結果"
```

再來可以用 `findAssocs` 找出最常與"同學"關程度0.5以上的關鍵字。

```
findAssocs(d.dtm, "同學", 0.5)
```

```
## 同學. 原因
```

```
##      0.56
```

## 結尾

以上我們介紹了如何將中文文章進行清理、斷詞等處理，最後轉換成矩陣，再進行一些簡單分析與繪圖。本文介紹之操作，還可以繼續進行關鍵字分群、主題模型、情緒分析等進階應用，有興趣繼續深入，可以自行搜尋 `tm`, `tmcn`, `Rwordseg` 這三個套件，可以發現許多資源自行學習。

## 作者

- 陳嘉葳：國立高雄大學資管所, [Kaohsiung useR! Meetup](#), [Taiwan R User Group](#)

## 參考資料

- [Rwordseg](#)
- [中文文字資料探勘](#) (英國Mango Solutions上海分公司資深顧問李艦)
- [MLDM Monday - 如何用 R 作中文斷詞](#)
- [using-the-r java-package-on-win7-64-bit-with-r](#)

程式人雜誌，採用 [創作共用：姓名標示、相同方式分享](#) 授權，歡迎加入 [雜誌社團](#)