



國立臺灣大學  
National Taiwan University

# 中文文字探勘與機器學習

---

國立臺灣大學共同教育中心

蔡芸琤

# 文字探勘目標

---

<https://buzzorange.com/techorange/2017/04/13/data-to-pxmart-hsu/>

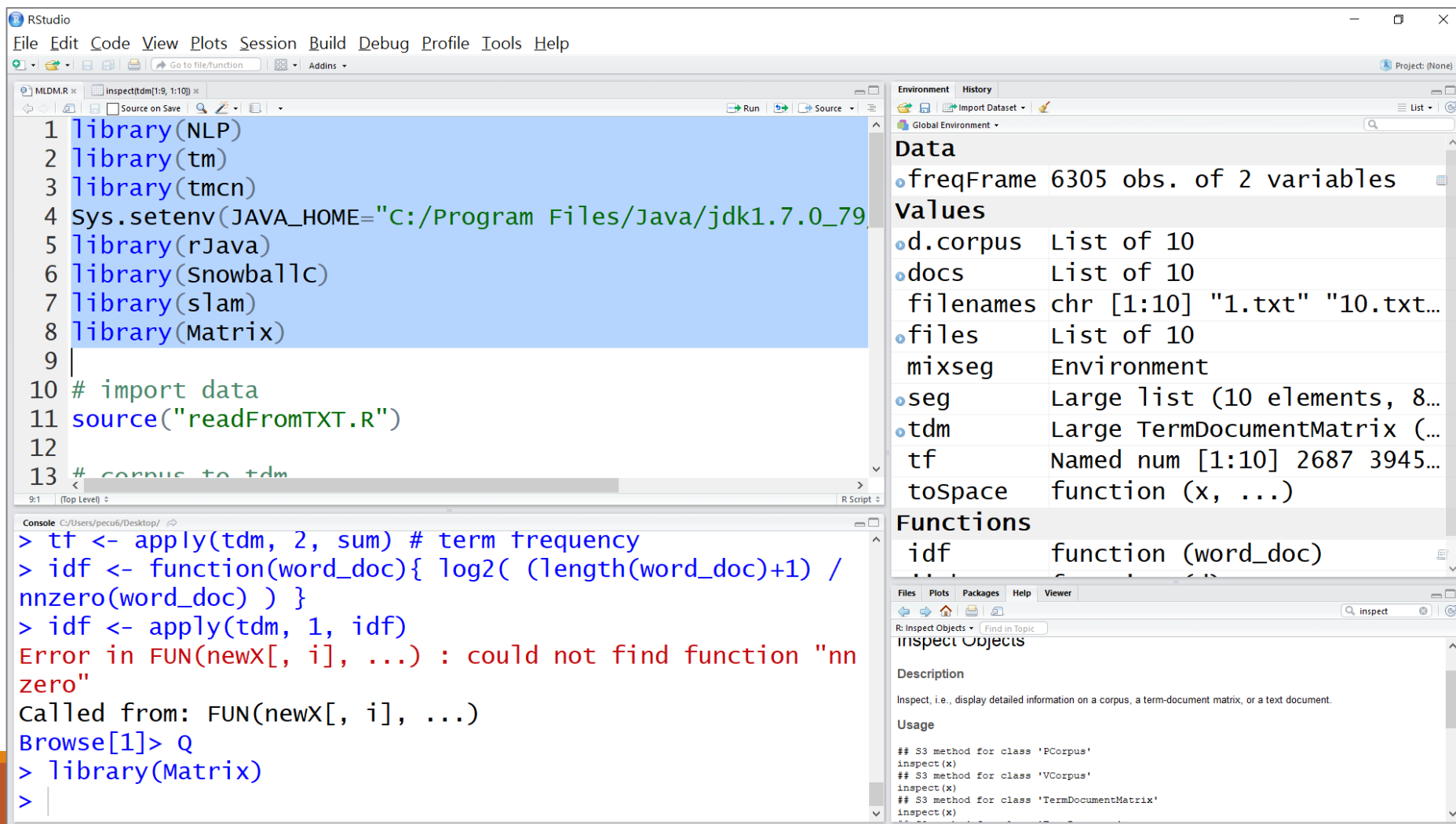
1. 文字雲 - done
2. 詞語詞間的關係
3. 文本關聯

# 套件安裝

---

1. library(tmcn) : [https://r-forge.r-project.org/R/?group\\_id=1571](https://r-forge.r-project.org/R/?group_id=1571) , tm的中文包
2. libart(tm) : 主要的 text mining 套件
3. library(rJava) : 給 Rwordseg 連到 java 的套件 ,  
<http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html> ,  
Sys.setenv(JAVA\_HOME="C:/Program Files/Java/jdk1.7.0\_79/")
4. SnowballC、slam、Matrix : 輔助套件

# 套件安裝



The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for installing and loading packages, setting the Java environment, and importing data. Lines 1-8 are highlighted in blue.
- Console:** Shows the execution of the code, including an error message for the `nnzero` function and the loading of the `Matrix` package.
- Environment Panel:** Lists objects in the Global Environment, including `freqFrame`, `d.corpus`, `docs`, `files`, `mixseg`, `seg`, `tdm`, `tf`, and `toSpace`.
- Functions Panel:** Lists functions, including `idf`.
- Inspector Panel:** Provides a description and usage for the selected object, `inspect(x)`.

```
1 library(NLP)
2 library(tm)
3 library(tmcn)
4 Sys.setenv(JAVA_HOME="C:/Program Files/Java/jdk1.7.0_79")
5 library(rJava)
6 library(SnowballC)
7 library(slam)
8 library(Matrix)
9
10 # import data
11 source("readFromTXT.R")
12
13 # corpus to tdm

> tf <- apply(tdm, 2, sum) # term frequency
> idf <- function(word_doc){ log2( (length(word_doc)+1) /
nnzero(word_doc) ) }
> idf <- apply(tdm, 1, idf)
Error in FUN(newX[, i], ...) : could not find function "nn
zero"
Called from: FUN(newX[, i], ...)
Browse[1]> Q
> library(Matrix)
>
```

**Environment Panel:**

**Data**

- freqFrame 6305 obs. of 2 variables

**Values**

- d.corpus List of 10
- docs List of 10
- files chr [1:10] "1.txt" "10.txt..."
- mixseg Environment
- seg Large list (10 elements, 8...)
- tdm Large TermDocumentMatrix (...)
- tf Named num [1:10] 2687 3945...
- toSpace function (x, ...)

**Functions**

- idf function (word\_doc)

**Inspector Panel:**

**Description**

Inspect, i.e., display detailed information on a corpus, a term-document matrix, or a text document.

**Usage**

```
## S3 method for class 'PCorpus'
inspect(x)
## S3 method for class 'VCorpus'
inspect(x)
## S3 method for class 'TermDocumentMatrix'
inspect(x)
```

# 資料測試

延續上週抓到的文本，轉成 Corpus 資料格式繼續進行接下來的分析技巧

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for loading libraries and creating a corpus. Lines 9-12 are highlighted with a red box.
- Environment:** Displays the objects created in the global environment.
- Console:** Shows the output of the `inspect(tdm)` command.
- Files:** A file explorer showing the project directory structure.

```
1 library(tm)
2 library(tmcn)
3 Sys.setenv(JAVA_HOME="C:/Program Files/Java/jdk1.7.0_79/")
4 library(rJava)
5 library(Rwordseg)
6 library(SnowballC)
7 library(slam)
8
9 # import data
10 filenames <- list.files(getwd(), pattern="*.txt")
11 files <- lapply(filenames, readLines)
12 docs <- Corpus(VectorSource(files))
13
14
15
```

**Environment:**

- Global Environment
- values
  - d.corpus List of 3
  - docs List of 10
    - filenamechr [1:10] "1.txt" "2..."
    - filena... chr [1:10] "1.txt" "1..."
  - files List of 10
    - id int [1:10] 1 2 3 4 5 ...
    - q chr [1:3] "部" "北" "..."
    - q.num num [1:3] 1 1 1
  - tdm List of 6
    - i : int [1:66] 1 2 3 4 5 6 7...
    - j : int [1:66] 1 1 1 1 1 1 1...
    - v : num [1:66] 39 133 121 75...

**Console:**

```
> inspect(tdm)
<<TermDocumentMatrix (terms: 66, documents: 3)>>
Non-/sparse entries: 66/132
Sparsity : 67%
Error in nchar(Terms(x), type = "chars") :
  invalid multibyte string, element 33
>
```

**Files:**

Name	Size	Modified
SHPaper		
SCI Paper.Ink	880 B	Sep 30, 2016, 1:13 AM
python教科書		
NTU-CSX		
MLDM.R	1.2 KB	May 2, 2017, 1:38 PM
Google雲端硬碟.Ink	987 B	Apr 20, 2017, 10:18 AM
FinTech		
ctbctx.Ink	865 B	Apr 28, 2017, 4:55 PM
Azure		
AWS		
main.R	247 B	Apr 23, 2017, 12:03 PM
pttTestFunction.R	747 B	Apr 25, 2017, 7:12 PM

# TermDocumentMatrix

Documents



Vector-space  
representation

However, complexity  
We will see how small  
Given a function based  
Using entropy of traffic  
We study the complexity  
of influencing elections  
through bribery: How  
computationally complex  
is it for an external actor  
to determine whether by  
a certain amount of  
bribing voters a specified  
candidate can be made  
the election's winner? We  
study this problem for  
election systems as varied  
as scoring ...

	D1	D2	D3	D4	D5
complexity	2		3	2	3
algorithm	3			4	4
entropy	1			2	
traffic		2	3		
network		1	4		

Term-document matrix

# TermDocumentMatrix

The screenshot shows the RStudio interface. The script editor contains the following code:

```
11  
12 # corpus to tdm  
13 d.corpus <- Corpus(VectorSource(seg))  
14 tdm <- TermDocumentMatrix(d.corpus,  
15                           control = list(wordLengths = c(2, Inf)))  
16 inspect(tdm)  
17  
18
```

The console output shows the execution of `inspect(tdm)`:

```
> inspect(tdm)  
<<TermDocumentMatrix (terms: 5937, documents: 10)>>  
Non-/sparse entries: 11907/47463  
Sparsity           : 80%  
Error in nchar(Terms(x), type = "chars") :  
  invalid multibyte string, element 10  
>
```

The Environment pane on the right shows the following objects:

- d.c..List of 1...
- doc..List of 1...
- fil..chr [1:10..
- fil..List of 1...
- mix..Environme...
- seg Large lis...
- tdm Large Ter...
- i : int [1:1...
- j : int [1:1...

A red box highlights the error message in the console and the corresponding line in the script. Below the red box, there is a note in Chinese:

Windows 有可能造成編碼錯誤而產生部分詞無法顯示

# TermDocumentMatrix

**View(inspect(tdm[1:9, 1:10]))**

The screenshot shows the RStudio interface with a TermDocumentMatrix (tdm) loaded. The top-left pane displays a 9x10 matrix of counts for the first 9 documents and the first 10 terms. The top-right pane shows the Environment window with the Data section expanded, listing variables like freqFrame, d.corpus, docs, filenames, files, mixseg, seg, tdm, tf, and toSpace. The bottom-left pane shows the Console output of the command View(inspect(tdm[1:9, 1:10])), displaying the same 9x10 matrix with row and column labels. The bottom-right pane shows the inspect Objects window, which provides a description and usage for the inspect function.

	1	2	3	4	5	6	7	8	9
了	29	27	46	33	30	39	29	22	21
了解	2	2	1	8	0	5	4	3	0
了点	1	0	0	0	0	0	0	0	0
二	3	7	3	0	7	5	2	2	2
二下	1	1	0	0	0	0	0	0	0
二部	1	0	0	2	0	0	0	0	0
人	9	18	14	21	5	13	12	11	17
人士	1	0	0	0	1	0	0	0	0
人家	4	0	0	0	1	0	0	0	0

```
## S3 method for class 'PCorpus'
inspect(x)
## S3 method for class 'VCorpus'
inspect(x)
## S3 method for class 'TermDocumentMatrix'
inspect(x)
```



# TermDocumentMatrix

	1	2	3	4	5	6	7	8	9	10
了	29	27	46	33	30	39	29	22	21	32
了解	2	2	1	8	0	5	4	3	0	1
了點	1	0	0	0	0	0	0	0	0	0
二	3	7	3	0	7	5	2	2	2	9
二下	1	1	0	0	0	0	0	0	0	0
二類	1	0	0	2	0	0	0	0	0	0
人	9	18	14	21	5	13	12	11	17	21
人士	1	0	0	0	1	0	0	0	0	0
人家	4	0	0	0	1	0	0	0	0	0

在文章中出現的次數

# TermDocumentMatrix

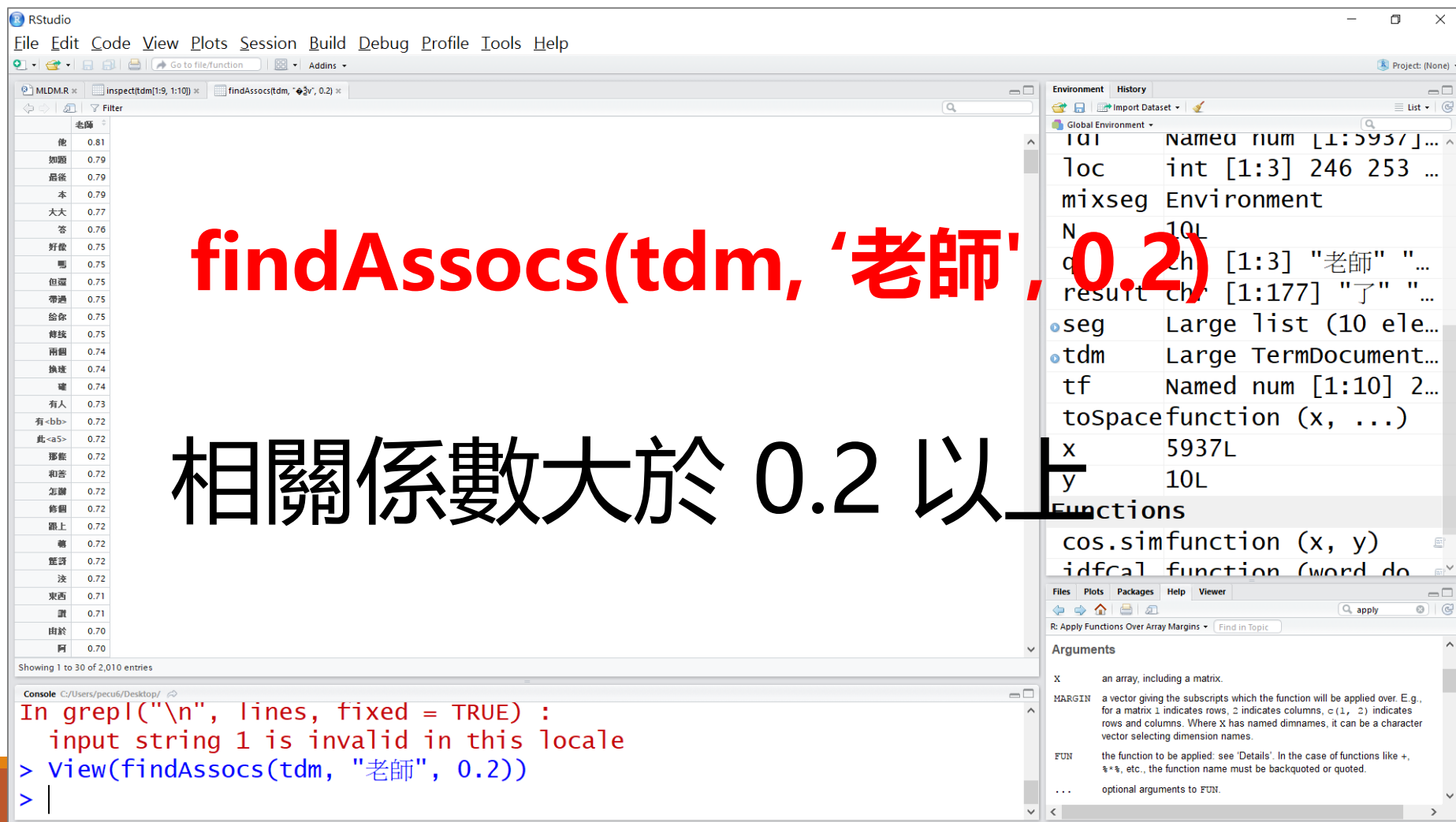
1. 可以用 `slam::row_sums` 來算出每個字的總出現次數
2. `tm::findFreqTerms` 直接找最常出現的字
3. 兩兩之間可以算相關係數 or 歐幾里德距離

	1	2	3	4	5	6	7	8	9	10
了	29	27	46	33	30	39	29	22	21	32
了解	2	2	1	8	0	5	4	3	0	1
了點	1	0	0	0	0	0	0	0	0	0
二	3	7	3	0	7	5	2	2	2	9
二下	1	1	0	0	0	0	0	0	0	0
二類	1	0	0	2	0	0	0	0	0	0
人	9	18	14	21	5	13	12	11	17	21
人士	1	0	0	0	1	0	0	0	0	0
人家	4	0	0	0	1	0	0	0	0	0

# TermDocumentMatrix

**findAssocs(tdm, '老師', 0.2)**

相關係數大於 0.2 以上



The screenshot shows the RStudio interface. The left pane displays a list of terms and their associated scores. The right pane shows the Environment window with the results of the findAssocs function. The console at the bottom shows the command being executed.

Term	Score
他	0.81
如題	0.79
最後	0.79
本	0.79
大大	0.77
答	0.76
好像	0.75
哪	0.75
但還	0.75
帶過	0.75
給你	0.75
傳統	0.75
兩個	0.74
換班	0.74
確	0.74
有人	0.73
有<bb>	0.72
此<a5>	0.72
那樣	0.72
和答	0.72
怎辦	0.72
飾個	0.72
罷上	0.72
哪	0.72
驚訝	0.72
淡	0.72
東西	0.71
讀	0.71
由於	0.70
阿	0.70

```
in grepl("\n", lines, fixed = TRUE) :  
  input string 1 is invalid in this locale  
> View(findAssocs(tdm, "老師", 0.2))  
> |
```

# TF-IDF (Term Frequency - Inverse Document Frequency)

---

$$w_{x,y} = \text{tf}_{x,y} \times \log \left( \frac{N}{\text{df}_x} \right)$$

**TF-IDF**

Term  $x$  within document  $y$

$\text{tf}_{x,y}$  = frequency of  $x$  in  $y$

$\text{df}_x$  = number of documents containing  $x$

$N$  = total number of documents

# TF-IDF

---

1. TF-IDF：是一種用於資訊檢索與文字探勘的常用加權技術，為一種統計方法，**用來評估單詞對於文件的集合或詞庫中一份文件的重要程度。**
2. TF ( Term Frequency ) :
  - y 是「某一特定文件」
  - x 是該文件中所使用單詞或單字的「其中一種」
  - $n(x,y)$  就是 x 在 y 當中的「出現次數」
  - **$tf(x,y) = n(x,y) / (n(1,y)+n(2,y)+n(3,y)+...+n(i,j))$ 。**

# TF (Term Frequency)

---

1. 例如第一篇文件篩選出「健康」、「富有」
  - 「健康」在該篇文件中出現 70 次，「富有」出現 30 次，
  - 「健康」的  $tf = 70 / (70+30) = 70/100 = 0.7$ ，
  - 「富有」的  $tf = 30 / (70+30) = 30/100 = 0.3$ ；
2. 第二篇文件同樣篩選出「健康」、「富有」
  - 「健康」在該篇文件中出現 40 次，「富有」出現 60 次，
  - 「健康」的  $tf = 40 / (40+60) = 40/100 = 0.4$ ，
  - 「富有」的  $tf = 60 / (40+60) = 60/100 = 0.6$ ，
3. TF 值愈高，其單詞愈重要。「健康」對第一篇文件比較重要，「富有」對第二篇文件比較重要。若搜尋「健康」，那第一篇文件會在較前面的位置；而搜尋「富有」，則第二篇文章會出現在較前面的位置。

# IDF (Inverse Document Frequency)

---

1.  $N$  是「所有的文件總數」， $x$  是文件中所使用的單詞， $df(x)$  是該單詞在所有文件總數中出現的「文件數」。
  - $idf(x) = \log ( N/df(x) ) = \log N - \log df(x)$ 。
2. 例如有100份文件，「健康」出現在10個文件當中，而「富有」出現在100個文件當中。
  - $idf(\text{健康}) = \log( 100/10 ) = \log 100 - \log 10 = 2 - 1 = 1$ 。
  - $idf(\text{富有}) = \log( 100/100 ) = \log 100 - \log 100 = 2 - 2 = 0$ 。
3. 「健康」出現的機會小，與出現機會很大的「富有」比較起來，便顯得非常重要，因此要加重權重。

# TF-IDF

---

將  $tf(x,y) * idf(x)$  (例如：x = 「健康」一詞) 來進行計算，以某一特定文件內的高單詞頻率，乘上該單詞在文件總數中的低文件頻率，便可以產生 TF-IDF 權重值，且 TF-IDF 傾向於過濾掉常見的單詞，保留重要的單詞，如此一來，「富有」便不重要了。



# TF-IDF

The screenshot displays the RStudio interface. The main editor window contains R code for TF-IDF computation, with lines 21 through 29 highlighted by a red rectangle. The code defines a function `idfcal` to calculate inverse document frequency and applies it to a term-document matrix `tdm`. The console window shows the output of `tdm[1,]`, indicating a sparsity of 80% and a single term '1' with a frequency of 10. The environment pane on the right lists variables: `idf` (a named numeric vector), `mixseg` (an environment), `N` (10), `seg` (a list), and `tdm` (a TermDocumentMatrix).

```
19  
20  
21 # tf-idf computation  
22 N = tdm$ncol  
23 tf <- apply(tdm, 2, sum)  
24 idfcal <- function(word_doc)  
25 {  
26   log2( N / nnzero(word_doc) )  
27 }  
28 idf <- apply(tdm, 1, idfcal)  
29  
30 |  
31  
32
```

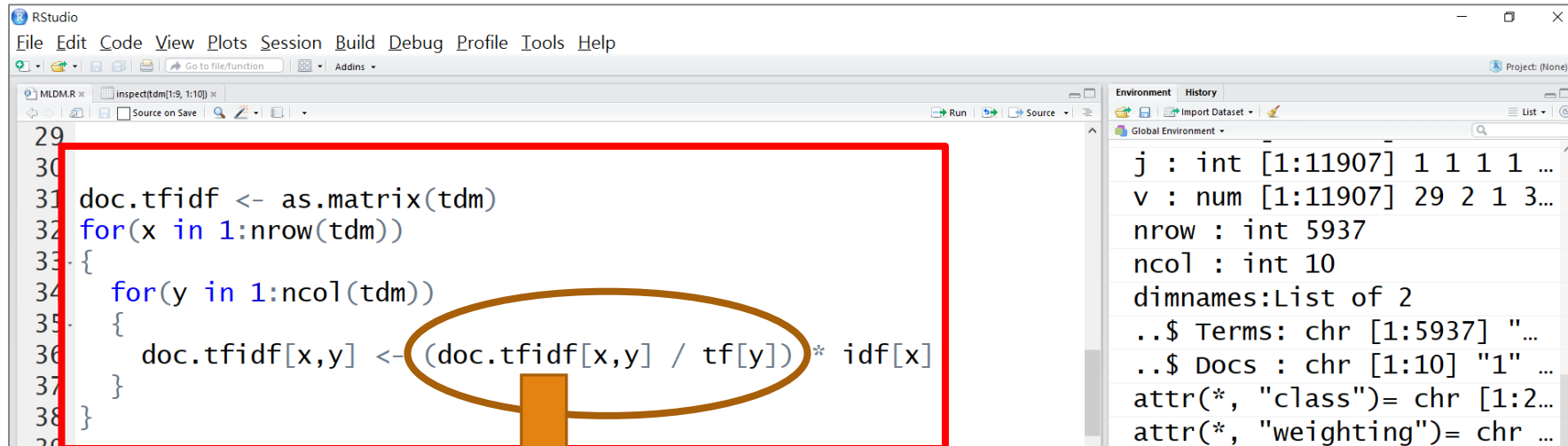
Console output:

```
sparsity : 80%  
Error in nchar(Terms(x), type = "chars") :  
  invalid multibyte string, element 10  
> tdm[1,]  
<<TermDocumentMatrix (terms: 1, documents: 10)>>  
Non-/sparse entries: 10/0  
Sparsity : 0%  
Maximal term length: 1  
weighting : term frequency (tf)  
>
```

Environment pane:

Variable	Value
idf	Named num [1:5937] ...
mixseg	Environment
N	10L
seg	Large list (10 elem...)
tdm	Large TermDocumentM...
i	int [1:11907] 1 2 3 4 ...
j	int [1:11907] 1 1 1 1 ...
v	num [1:11907] 29 2 1 3...
nrow	int 5937
ncol	int 10
dimnames	List of 2
..\$ Terms	chr [1:5937] "..."
..\$ Docs	chr [1:10] "1" ...
attr(*, "class")	chr [1:2...

# TF-IDF



The screenshot shows the RStudio interface. The main editor window contains the following R code for calculating TF-IDF:

```
29  
30  
31 doc.tfidf <- as.matrix(tdm)  
32 for(x in 1:nrow(tdm))  
33 {  
34   for(y in 1:ncol(tdm))  
35   {  
36     doc.tfidf[x,y] <- (doc.tfidf[x,y] / tf[y]) * idf[x]  
37   }  
38 }  
39
```

A red rectangle highlights the entire code block. A brown oval highlights the expression  $(\text{doc.tfidf}[x,y] / \text{tf}[y])$  in line 36. A large orange arrow points from this expression down to the mathematical formula below.

The Environment pane on the right shows the following variables:

```
j : int [1:11907] 1 1 1 1 ...  
v : num [1:11907] 29 2 1 3...  
nrow : int 5937  
ncol : int 10  
dimnames:List of 2  
..$ Terms: chr [1:5937] "...  
..$ Docs : chr [1:10] "1" ...  
attr(*, "class")= chr [1:2...  
attr(*, "weighting")= chr ...
```

$$w_{x,y} = \text{tf}_{x,y} \times \log \left( \frac{N}{df_x} \right)$$

```
> for(x in 1:nrow(tdm))  
+ {
```

etc., the function name must be backquoted or quoted.  
... optional arguments to FUN.

# 畫出 tf-idf 統計圖

<https://www.slideshare.net/chaolinliu/2015jan27>



The screenshot displays the RStudio interface. The source editor on the left contains R code for plotting a tf-idf scatter plot. The code is as follows:

```
44  
45 # 畫出 tf-idf 統計圖  
46 library(plotly)  
47 topID = lapply(rownames(as.data.frame(ass)), function(x)  
48   which(rownames(tdm) == x))  
49 topID = unlist(topID)  
50 plot_ly(data = as.data.frame(doc.tfidf),  
51   x = as.numeric(colnames(doc.tfidf)),  
52   y = doc.tfidf[topID[10],],  
53   name = rownames(doc.tfidf)[topID[10]],  
54   type = "scatter", mode= "box") %>%  
55 add_trace(y = doc.tfidf[topID[2],],  
56   name = rownames(doc.tfidf)[topID[2]])  
57
```

The Environment pane on the right shows the following objects:

Object	Type	Value
d.cor...	List of 10	
docs	List of 10	
filen...	chr [1:10]	"1.txt"...
files	List of 10	
idf	Named num [1:5937]...	
loc	int [1:3]	246 253 ...
mixseg	Environment	
N	10L	
q	chr [1:3]	"老師" "..."
result	chr [1:15]	"了" "..."
seg	Large list (10 ele...	
tdm	Large TermDocument...	
tf	Named num [1:10]	2...
topID	int [1:12]	129 226...

Below the code, the console shows the execution of the plotly code:

```
+ type = "scatter", mode= "box") %>%  
+ add_trace(y = doc.tfidf[topID[2],],  
+ name = rownames(doc.tfidf)[topID[2]])  
> doc.tfidf[topID[10],]  
1  
0.0000000000 C  
6  
0.0000000000 C  
>
```

**Problem -> Info : 已經先有想關心的關鍵字  
想找出這些關鍵字和文本的關係**

# 尋找關鍵字與文章之間的關聯

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for TF-IDF analysis. A red box highlights lines 58-66, which filter for non-zero terms in the document-term matrix.
- Environment:** Lists objects in the global environment, including `s.tdm` (Large matrix), `ass` (List of 1), `d.cor...` (List of 10), `docs` (List of 10), `filen...` (chr [1:10]), `files` (List of 10), `idf` (Named num [1:5937]), `loc` (int [1:5818]), `mixseg` (Environment), `N` (10L), `nonze...` (Named int [1:5818]), and `q` (chr [1:5818]).
- Console:** Shows the output of `view(s.tdm)`, displaying cosine similarity values for various terms. The output is truncated after 5718 rows.

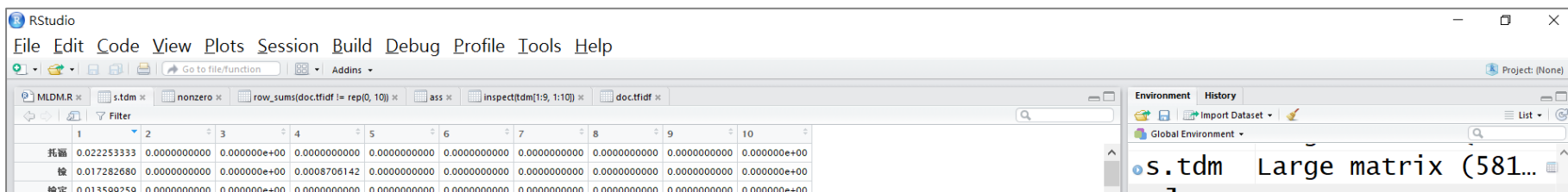
**Console Output:**

```
αéαă 0.000000e+00
αé»y 0.000000e+00
αìμP 0.000000e+00
αñ 9.966814e-05
αñαè»i 0.000000e+00
αýα] 0.000000e+00
[ reached getOption("max.print") -- omitted 5718 rows ]
> view(s.tdm)
>
```

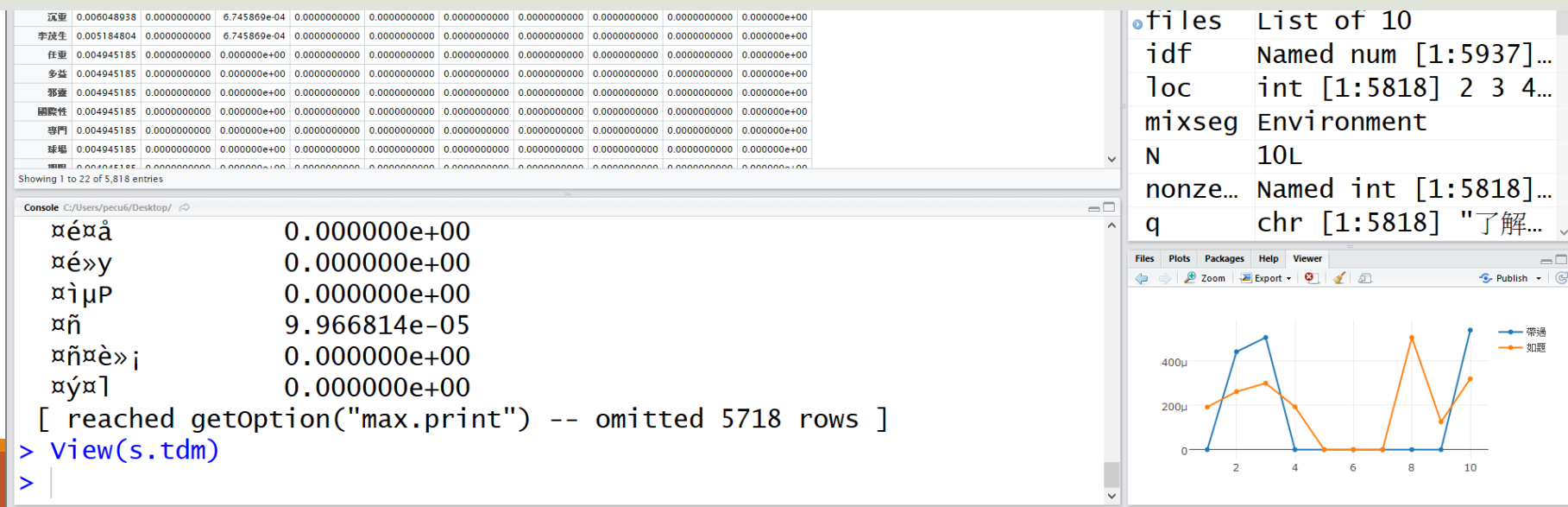
**DATA -> Info**

A line plot is visible in the bottom right corner of the RStudio window. The x-axis ranges from 0 to 10, and the y-axis ranges from 0 to 400. Two series are plotted: '帶過' (blue line) and '如題' (orange line). The '帶過' series shows a peak around iteration 3, while the '如題' series shows a peak around iteration 8.

# 尋找關鍵字與文章之間的關聯



DATA -> Info：從資料中去發掘甚麼才是重要的關鍵字  
找出托福、檢定是對第一份文本最重要的關鍵字



# 餘弦相似性 (cos similarity ranking)

---

## 1. 餘弦相似性用在找出相似的文章。

- 句子A：我喜歡看電視，不喜歡看電影。
- 句子B：我不喜歡看電視，也不喜歡看電影。

## 2. 請問怎樣才能計算上面兩句話的相似程度？

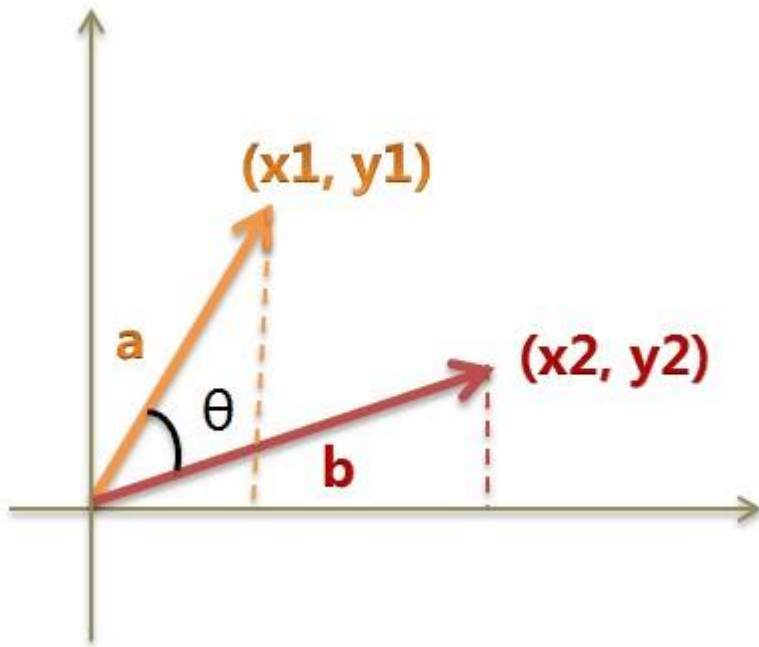
- 基本思路是：如果這兩句話的用詞越相似，它們的內容就應該越相似。因此，可以從詞頻入手，計算它們的相似程度。

# 餘弦相似性 (cos similarity ranking)

---

把文章對應到的 IF-IDF 想像成空間中的向量。兩條線段之間形成一個夾角，如果夾角為0度，意味著方向相同、線段重合；如果夾角為90度，意味著形成直角，方向完全不相似；如果夾角為180度，意味著方向正好相反。因此，我們可以通過夾角的大小，來判斷向量的相似程度。夾角越小，就代表越相似。

# 餘弦相似性 (cos similarity ranking)

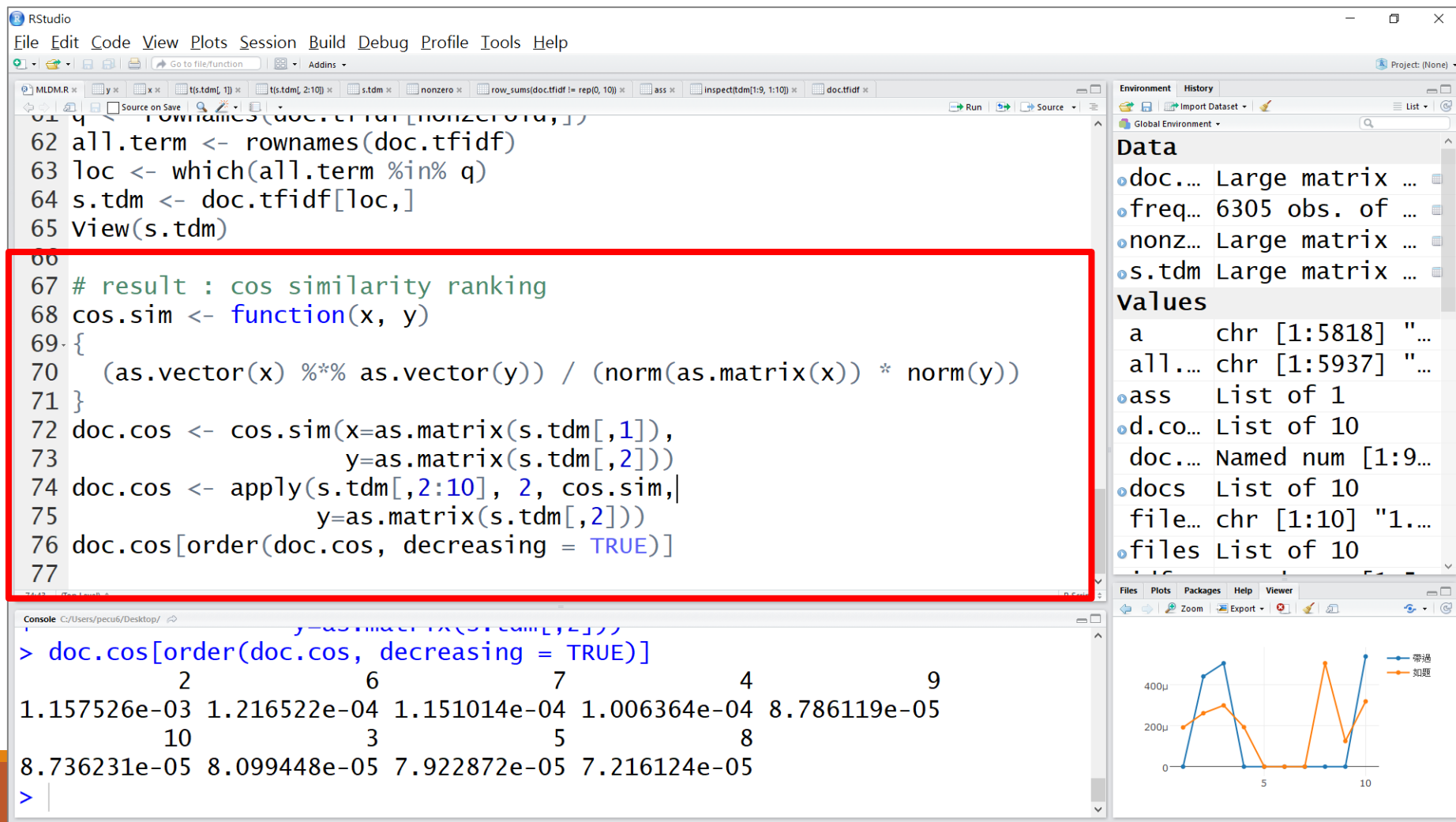


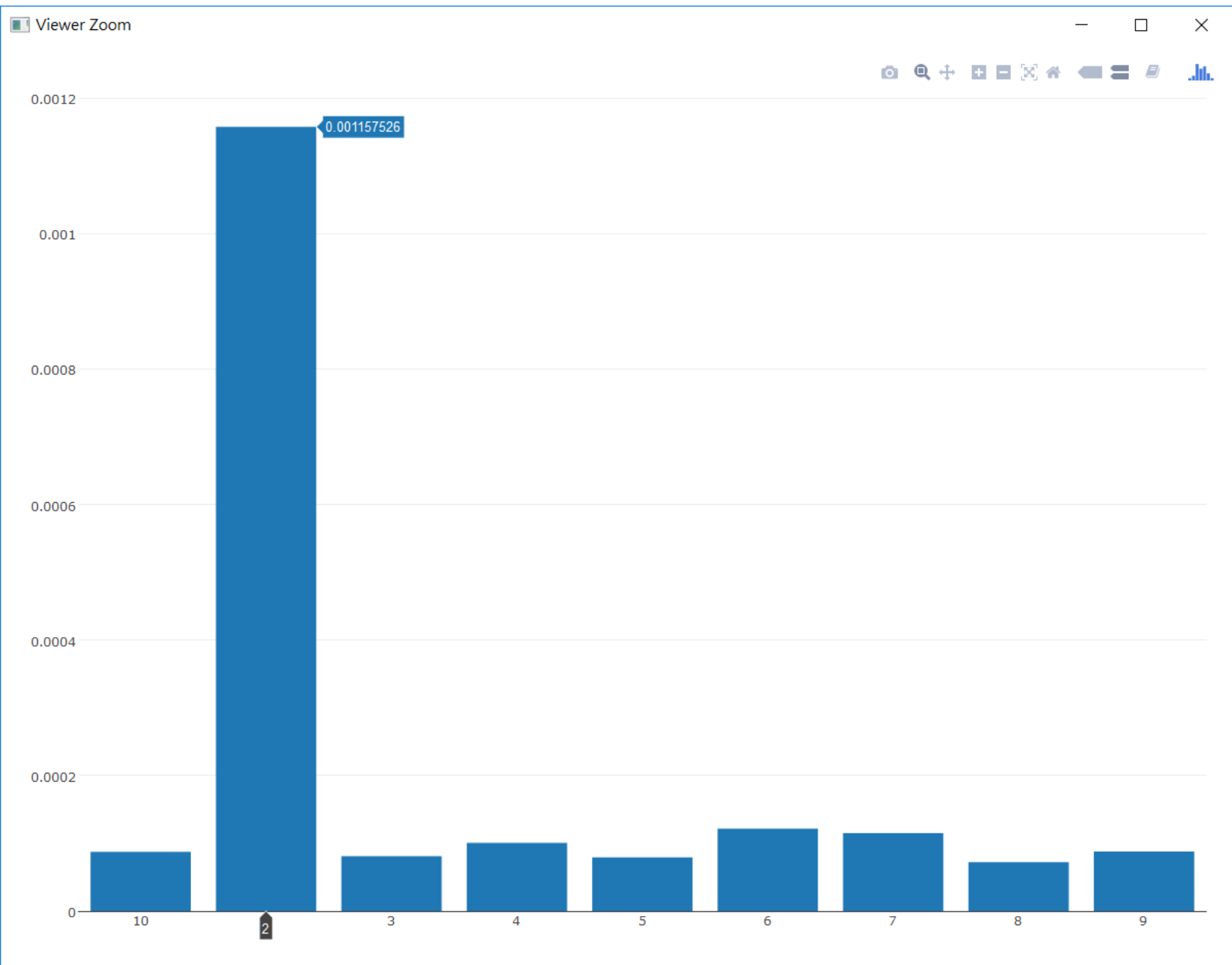
$$\cos\theta = \frac{x_1x_2 + y_1y_2}{\sqrt{x_1^2 + y_1^2} \times \sqrt{x_2^2 + y_2^2}}$$

$$\begin{aligned}\cos\theta &= \frac{\sum_{i=1}^n (A_i \times B_i)}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}} \\ &= \frac{A \cdot B}{|A| \times |B|}\end{aligned}$$



# 哪個文本和第一個文本最相似？





RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

MLDM.R x a x orderDoc x y x x x t(s.tdm[, 1]) x t(s.tdm[, 2:10]) x s.tdm x nonzero x row\_sums(doc.tfidf != rep(0, 10)) x ass x inspect(tdm[1:9, 1:10]) x doc.tfidf x

```
63 loc <- which(all.term %in% q)
64 s.tdm <- doc.tfidf[loc,]
65 view(s.tdm)
66
67 # result : cos similarity ranking
68 cos.sim <- function(x, y)
69 {
70   (as.vector(x) %*% as.vector(y)) / (norm(as.matrix(x)) * norm(y))
71 }
72 doc.cos <- cos.sim(x=as.matrix(s.tdm[,1]),
73                   y=as.matrix(s.tdm[,2:10]))
74 doc.cos <- apply(s.tdm[,2:10], 2, cos.sim,
75                 v=as.matrix(s.tdm[,2:10]))
76 orderDoc <- doc.cos[order(doc.cos, decreasing = TRUE)]
77 plot_ly(data = as.data.frame(orderDoc),
78         x = rownames(as.data.frame(orderDoc)),
79         y = orderDoc,
80         name = rownames(doc.tfidf)[topID[10]],
81         type = "bar", mode = "box")
82
```

Environment History

Global Environment

Data

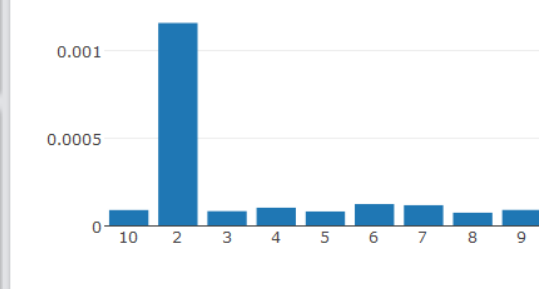
a	9 obs. of 1 v...
doc...	Large matrix ...
freq...	6305 obs. of ...
nonz...	Large matrix ...
s.tdm	Large matrix ...

Values

all...	chr [1:5937] "...
ass	List of 1
d.co...	List of 10
doc...	Named num [1:9...
docs	List of 10
file...	chr [1:10] "1....
files	List of 10

Files Plots Packages Help Viewer

Zoom Export



Console C:/Users/pecu6/Desktop/ R Script

```
> rownames(as.data.frame(orderDoc))
[1] "2" "6" "7" "4" "9" "10" "3" "5" "8"
>
```

# 文字探勘應用面

---

設計基礎：

文字與文章 -> 數值矩陣

TermDocumentMatrix、TF-IDF、詞頻、word2vec

推廣應用：

以文找文、關聯詞分析、關鍵詞分析、  
文件分類、文件分群

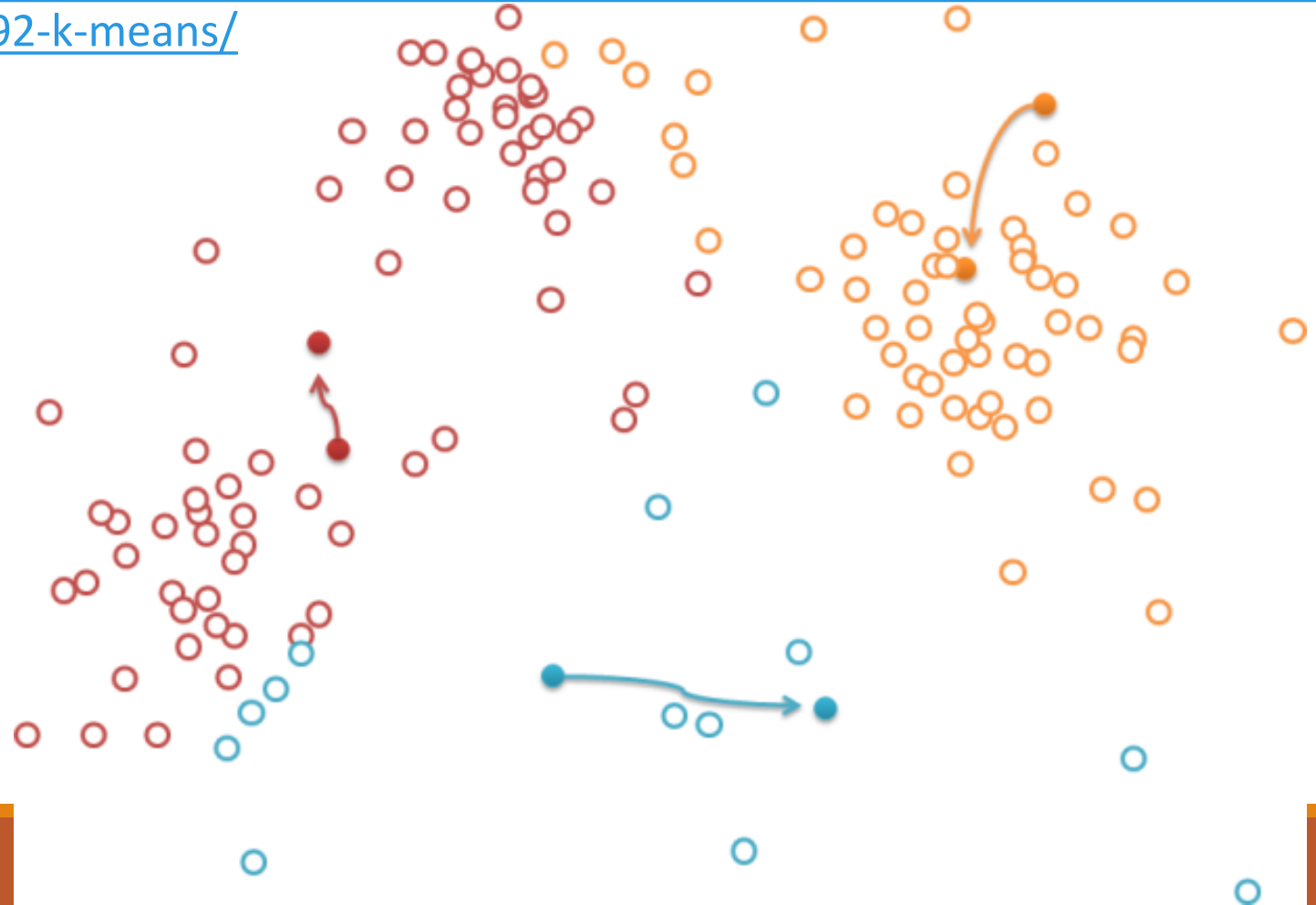
# 相關演算法

---

1. <https://docs.microsoft.com/zh-tw/azure/machine-learning/machine-learning-algorithm-choice>
2. <http://fecbob.pixnet.net/blog/post/38189923-%E8%B3%87%E6%96%99%E6%8E%A1%E7%A4%A6%E5%8D%81%E5%A4%A7%E7%B6%93%E5%85%B8%E6%BC%94%E7%AE%97%E6%B3%95>
3. <https://www.inside.com.tw/2016/07/22/what-is-data-mining>

# 以 TF-IDF 進行 K-Means 分群

<https://chtseng.wordpress.com/2017/03/03/%E9%9D%9E%E7%9B%A3%E7%9D%A3%E5%BC%8F%E5%AD%B8%E7%BF%92-k-means/>



# 以 TF-IDF 進行 K-Means 分群

---

```
# Kmeans 分群
```

```
library(stats)
```

```
kmeansOut <- kmeans(doc.tfidf, 10, nstart = 20)
```

# 練習問自己問題

---

[https://www.slideshare.net/tw\\_dsconf/dsc-2016-r-67850142](https://www.slideshare.net/tw_dsconf/dsc-2016-r-67850142)

1. 計算人與人之間的距離
2. 計算商品與商品之間的距離
3. 計算每個人購買種商品的機率
4. 計算每個人購買各種商品的最佳順序
5. 只要有資料的受眾就可以納入計算
6. 從資料出發，不受既定印象與偏見影響，因此可以發掘潛在客戶
7. 成功要素：資料量、使用者涵蓋範圍、反應時間