

Program – 10

AIM: Write a program to find similar documents with cosine similarity.

Introduction and Theory

Cosine Similarity

Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.

The cosine for two non-zero vector can be calculated from the Euclidean dot product.

$$A \cdot B = ||A|| ||B|| \cos(\theta)$$

The cosine similarity is then calculated as:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{||A|| ||B||} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Code

```
Cosine.R
1 library(twitteR)
2 library(tidytext)
3 library(dplyr)
4 library(tm)
5 library(SnowballC)
6
7 workingDir <- "~/Documents/R programs"
8 setwd(workingDir)
9
10 url <- "http://www.rdatamining.com/data/rdmTweets-201306.RData"
11 download.file(url, destfile = "rdmTweets-201306.RData")
12
13 load(file = "rdmTweets-201306.RData")
14 tweets <- twListToDF(tweets)
15 load(file = "rdmTweets-201306.RData")
16 tweets <- twListToDF(tweets)
17
18 tweets <- tweets %>%
19   mutate(text=gsub("(http|https).+?$|\\n|&|[:punct:]", "", text),
20          rowIndex=as.numeric(row.names(.))) %>%
21   select(text, retweetCount, rowIndex)
22
23 docList <- as.list(tweets$text)
```

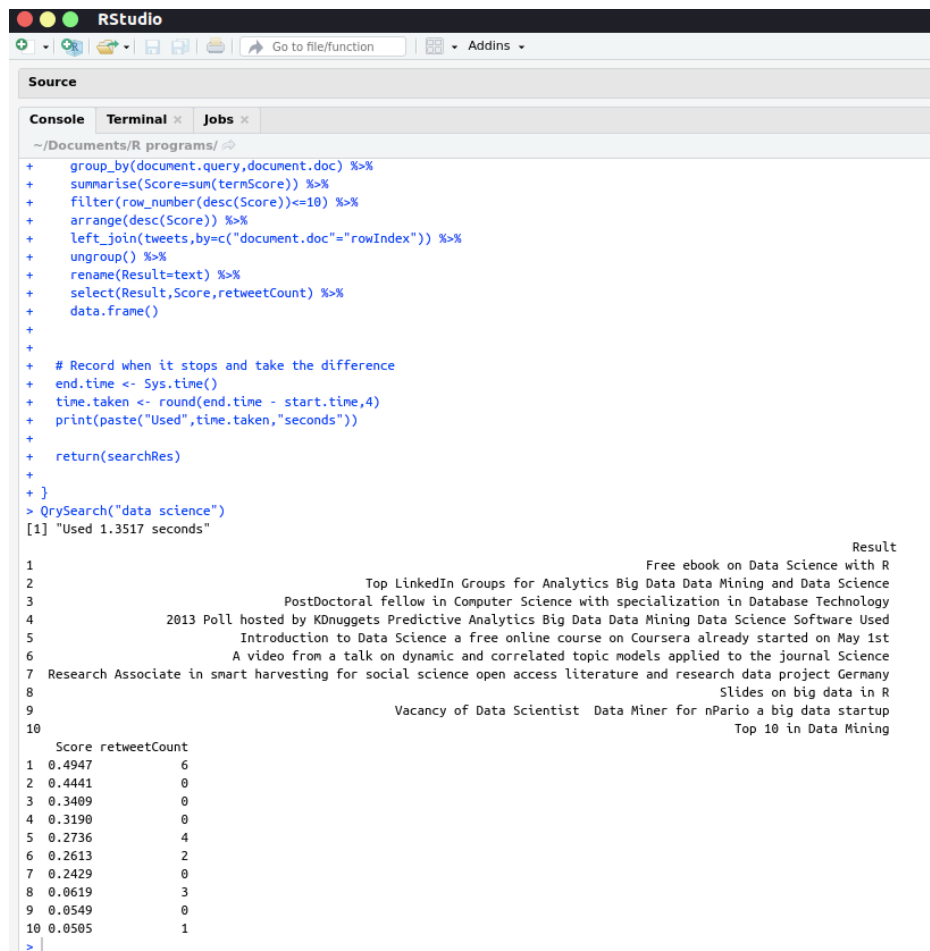
Program – 10

```
24 N.docs <- length(docList)
25
26 QrySearch <- function(queryTerm) {
27
28   # Record starting time to measure your search engine performance
29   start.time <- Sys.time()
30
31   # store docs in Corpus class which is a fundamental data structure in
text mining
32   my.docs <- VectorSource(c(docList, queryTerm))
33
34
35   # Transform/standardize docs to get ready for analysis
36   my.corpus <- VCorpus(my.docs) %>%
37     tm_map(stemDocument) %>%
38     tm_map(removeNumbers) %>%
39     tm_map(content_transformer(tolower)) %>%
40     tm_map(removeWords, stopwords("en")) %>%
41     tm_map(stripWhitespace)
42
43
44   # Store docs into a term document matrix where rows=terms and cols=docs
45   # Normalize term counts by applying TDiF weightings
46   term.doc.matrix.stm <- TermDocumentMatrix(my.corpus,
47                                             control=list(
48                                             weighting=function(x)
weightSMART(x, spec="ltc"),
49                                             wordLengths=c(1, Inf)))
50
51
52
53   # Transform term document matrix into a dataframe
54   term.doc.matrix <- tidy(term.doc.matrix.stm) %>%
55     group_by(document) %>%
56     mutate(vtrLen=sqrt(sum(count^2))) %>%
57     mutate(count=count/vtrLen) %>%
58     ungroup() %>%
59     select(term:count)
60   docMatrix <- term.doc.matrix %>%
61     mutate(document=as.numeric(document)) %>%
62     filter(document<N.docs+1)
63   qryMatrix <- term.doc.matrix %>%
64     mutate(document=as.numeric(document)) %>%
65     filter(document>=N.docs+1)
66
67   # Calculate top ten results by cosine similarity
68   searchRes <- docMatrix %>%
69     inner_join(qryMatrix, by=c("term"="term"),
70              suffix=c(".doc", ".query")) %>%
71     mutate(termScore=round(count.doc*count.query, 4)) %>%
72     group_by(document.query, document.doc) %>%
```

Program – 10

```
73 summarise(Score=sum(termScore)) %>%
74 filter(row_number(desc(Score))<=10) %>%
75 arrange(desc(Score)) %>%
76 left_join(tweets,by=c("document.doc"="rowIndex")) %>%
77 ungroup() %>%
78 rename(Result=text) %>%
79 select(Result,Score,retweetCount) %>%
80 data.frame()
81
82
83 # Record when it stops and take the difference
84 end.time <- Sys.time()
85 time.taken <- round(end.time - start.time,4)
86 print(paste("Used",time.taken,"seconds"))
87
88 return(searchRes)
89
90
91 }
92
93 QrySearch("data science")
```

Results & Outputs



The screenshot shows the RStudio interface with the 'Source' editor displaying the R code and the 'Console' showing the execution output. The code defines a function `QrySearch` that queries tweets, calculates scores, and returns a data frame of results. The console output shows the execution time and the resulting data frame.

```
Source
Console Terminal Jobs
~/Documents/R programs/
+ group_by(document.query,document.doc) %>%
+ summarise(Score=sum(termScore)) %>%
+ filter(row_number(desc(Score))<=10) %>%
+ arrange(desc(Score)) %>%
+ left_join(tweets,by=c("document.doc"="rowIndex")) %>%
+ ungroup() %>%
+ rename(Result=text) %>%
+ select(Result,Score,retweetCount) %>%
+ data.frame()
+
+ # Record when it stops and take the difference
+ end.time <- Sys.time()
+ time.taken <- round(end.time - start.time,4)
+ print(paste("Used",time.taken,"seconds"))
+
+ return(searchRes)
+
+ }
+
> QrySearch("data science")
[1] "Used 1.3517 seconds"

Result
1 Free ebook on Data Science with R
2 Top LinkedIn Groups for Analytics Big Data Data Mining and Data Science
3 PostDoctoral fellow in Computer Science with specialization in Database Technology
4 2013 Poll hosted by KDnuggets Predictive Analytics Big Data Data Mining Data Science Software Used
5 Introduction to Data Science a free online course on Coursera already started on May 1st
6 A video from a talk on dynamic and correlated topic models applied to the journal Science
7 Research Associate in smart harvesting for social science open access literature and research data project Germany
8 Slides on big data in R
9 Vacancy of Data Scientist Data Miner for nPario a big data startup
10 Top 10 in Data Mining

  Score retweetCount
1 0.4947             6
2 0.4441             0
3 0.3409             0
4 0.3190             0
5 0.2736             4
6 0.2613             2
7 0.2429             0
8 0.0619             3
9 0.0549             0
10 0.0505            1
> |
```

output

Program – 10

Findings and Learnings:

1. Cosine similarity is one of the most commonly used similarity metrics in text processing
2. R provides easy to use tools for performing text analysis.
3. We have successfully implemented document retrieval using cosine similarity.