Sentimental_Analysis

Amal

11/10/2019

```
install.packages("RColorBrewer")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
install.packages("wordcloud")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(RColorBrewer)
library(wordcloud)
install.packages("tm")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
install.packages("twitteR")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(tm)
```

```
## Loading required package: NLP
```

```
library(twitteR)
install.packages("plyr")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
install.packages("ROAuth")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(ROAuth)
library(plyr)
```

```
##
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:twitteR':
##
##     id
```

```
install.packages("stringr")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(stringr)
install.packages("base64enc")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(base64enc)
install.packages("SnowballC")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/3.6'
## (as 'lib' is unspecified)
```

```
library(SnowballC)

#Setting twitter connection and authentication

# Getting a Setting twitter connection and authenticationcurl certification
download.file(url="http://curl.haxx.se/ca/cacert.pem",destfile="cacert.pem")

# Setting the certification for Twitter
requestURL <- "https://api.twitter.com/oauth/request_token"
accessURL <- "https://api.twitter.com/oauth/access_token"
authURL <- "https://api.twitter.com/oauth/authorize"

# Authorization from Twitter
consumerKey <- "12B8PGmPLG5JydHZNaAwAzeXM"
consumerSecret <- "5uaJxuutb3JJ5UqL7bdm1FW73NR0GD9IWGtvjIiNLpUb9ZIkxn"
accessToken <- "1085602973294579712-NpWMxkjvzW466W4LZlRHJFea3TxzgE"
accessTokenSecret <- "0N7fcVGm3j1uqPkY6LPeSeFhod5IbbQqo5diw9R3qwbqt"

setup_twitter_oauth(consumerKey,
                    consumerSecret,
                    accessToken,
                    accessTokenSecret)
```

```
## [1] "Using direct authentication"
```

```
## [1] "Using direct authentication"
#Now we can get it started!

#Extracting Tweets
searchString <- "#Bigdata"
bigdata <- searchTwitter(searchString, n=500, lang='en')

# Creating text files
bigdata_text <- sapply(bigdata, function(x) x$getText())
#It’s time to clean the data

bigdata_text <- gsub('https://','',bigdata_text) # removes https://
bigdata_text <- gsub('http://','',bigdata_text) # removes http://
bigdata_text <- gsub('[^[:graph:]]', ' ',bigdata_text) ## removes graphic characters like emotic
ons
bigdata_text <- gsub('[[:punct:]]', '', bigdata_text) # removes punctuation
bigdata_text <- gsub('[[:cntrl:]]', '', bigdata_text) # removes control characters
bigdata_text <- gsub('\\d+', '', bigdata_text) # removes numbers
bigdata_text <- str_replace_all(bigdata_text,"[^[:graph:]]", " ")

#Creating corpus
bigdata_text_corpus <- Corpus(VectorSource(bigdata_text))

#Transforming to lower case
bigdata_text_corpus <- tm_map(bigdata_text_corpus, content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(bigdata_text_corpus,
## content_transformer(tolower)): transformation drops documents
```

```
## content_transformer(tolower)): transformation drops documents
#Removing stop words
bigdata_text_corpus <- tm_map(bigdata_text_corpus, function(x)removeWords(x,stopwords()))
```

```
## Warning in tm_map.SimpleCorpus(bigdata_text_corpus, function(x)
## removeWords(x, : transformation drops documents
```

```
## Warning in tm_map.SimpleCorpus(trump_text_corpus, function(x)
## removeWords(x, : transformation drops documents
bigdata_text_corpus <- tm_map(bigdata_text_corpus, removeWords, c("bigdata","RT", "just","big",
"new","via","one","gppulipaka","now","reachratan"))
```

```
## Warning in tm_map.SimpleCorpus(bigdata_text_corpus, removeWords,
## c("bigdata", : transformation drops documents
```

```
## transformation drops documents
#Let's transform it into a dataframe to see the word frenquency

# building a term-document matrix
bigdata_2 <- TermDocumentMatrix(bigdata_text_corpus)
bigdata_2 <- as.matrix(bigdata_2)
bigdata_2 <- sort(rowSums(bigdata_2),decreasing=TRUE)

#Converting words to dataframe
bigdata_2 <- data.frame(word = names(bigdata_2),freq=bigdata_2)
#Taking a look at it

#Taking a look at frequency table
head(bigdata_2, 10)
```

```
##                                      word freq
## cybersecurity                 cybersecurity  131
## data                                  data   95
## machinelearning             machinelearning   87
## iot                                    iot   83
## analytics                        analytics   79
## databreach                      databreach   76
## datascience                    datascience   60
## fintech                           fintech   43
## tech                                 tech   41
## artificialintelligence artificialintelligence   40
```

```
#Creating the word cloud
set.seed(5678)
wordcloud(bigdata_text_corpus, min.freq=1, max.words=500, scale=c(1.5,0.5), colors=brewer.pal(8,
"Dark2"), random.color=T, random.order=F)
```

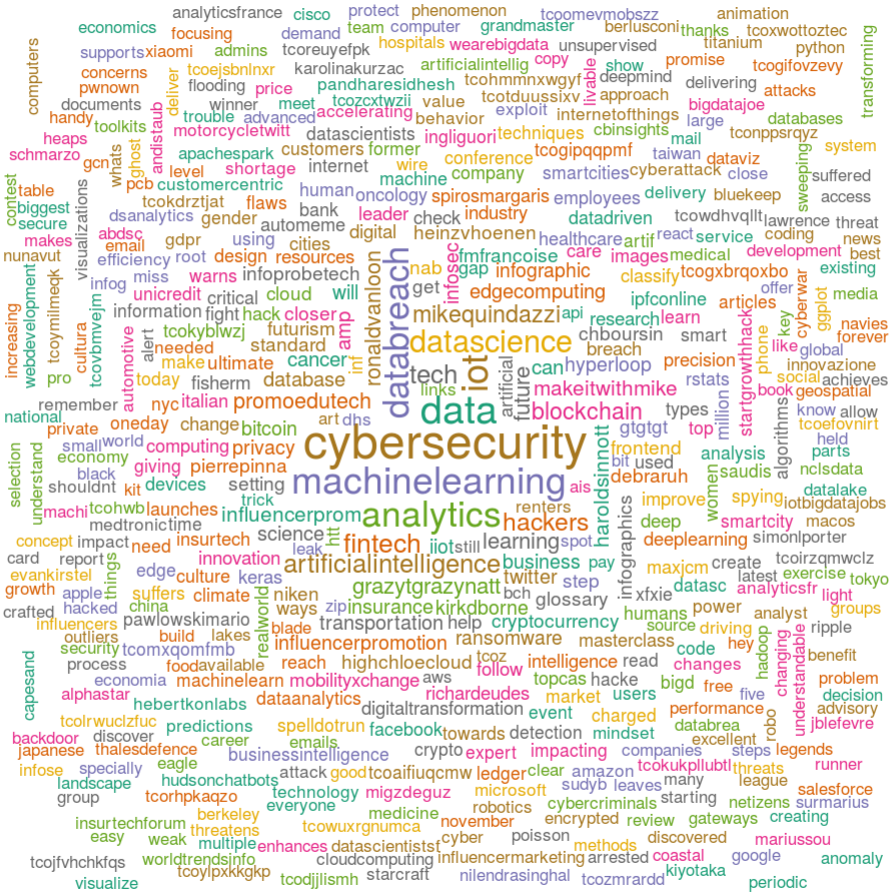
```
## Warning in wordcloud(bigdata_text_corpus, min.freq = 1, max.words = 500, :
## emergingtechnologies could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(bigdata_text_corpus, min.freq = 1, max.words = 500, :
## bigdataanalytics could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(bigdata_text_corpus, min.freq = 1, max.words = 500, :
## fabriziobustama could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(bigdata_text_corpus, min.freq = 1, max.words = 500, :
## containerized could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(bigdata_text_corpus, min.freq = 1, max.words = 500, :
## kubernetes could not be fit on page. It will not be plotted.
```

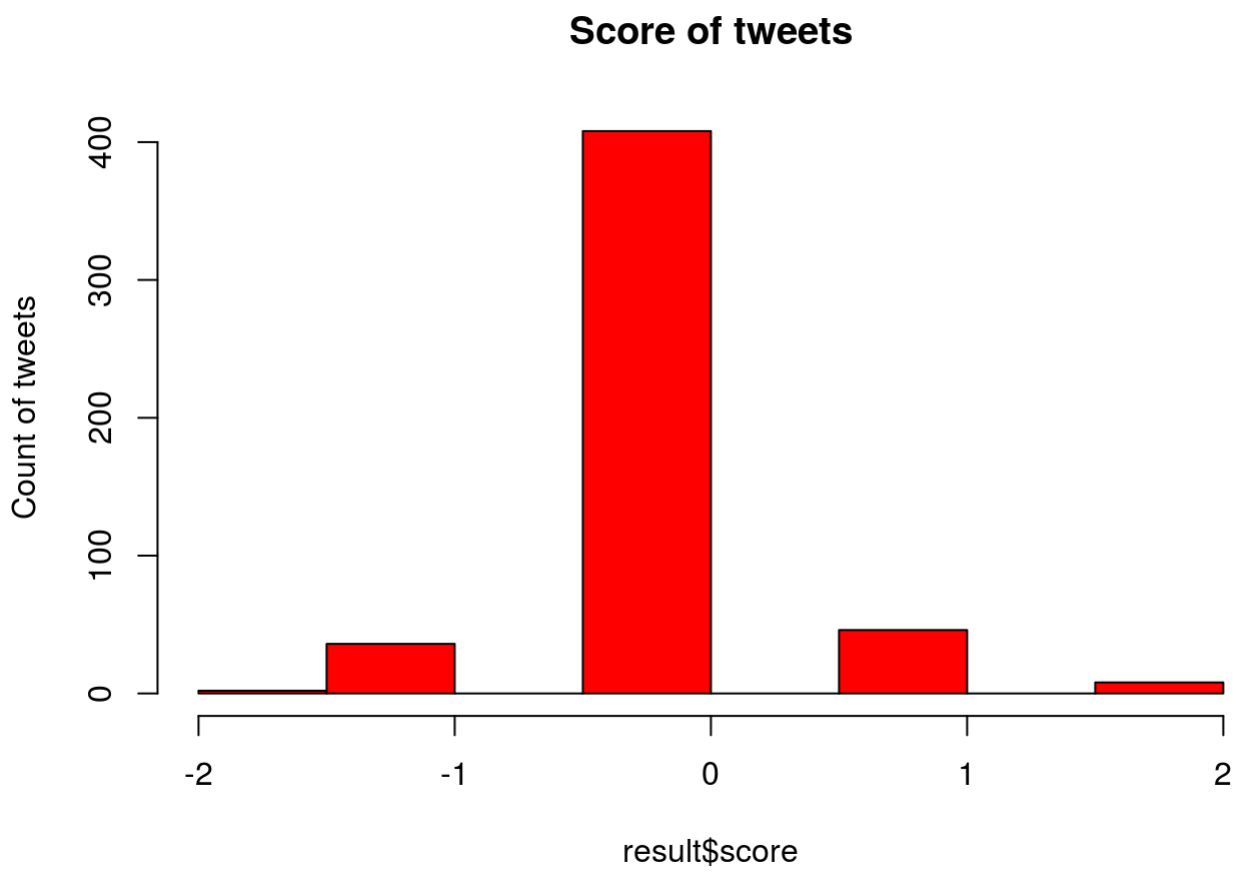


```
#Sentiment analysis
#We are going to use these lists to classify words as positive, negative and neutral: http://ww
w.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar

pos.words <- scan(file='positive-words.txt', what='character')
neg.words <- scan(file='negative-words.txt', what='character')
#It’s time to build a function help us with this project

score.sentiment = function(tweets, pos.words, neg.words)

{

  require(plyr)
  require(stringr)

  scores = laply(tweets, function(tweet, pos.words, neg.words) {

    tweet = gsub('https://','',tweet) # removes https://
    tweet = gsub('http://','',tweet) # removes http://
    tweet = gsub('[^[:graph:]]', ' ',tweet) ## removes graphic characters like emoticons
    tweet = gsub('[[:punct:]]', '', tweet) # removes punctuation
    tweet = gsub('[[:cntrl:]]', '', tweet) # removes control characters
    tweet = gsub('\\d+', '', tweet) # removes numbers
    tweet = iconv(tweet, "ASCII", "UTF-8", sub="")

    word.list = str_split(tweet, '\\s+') # splits the tweets by word in a list

    words = unlist(word.list) # turns the list into vector

    pos.matches = match(words, pos.words) ## returns matching
    #values for words from list
    neg.matches = match(words, neg.words)

    pos.matches = !is.na(pos.matches) ## converts matching values to true of false
    neg.matches = !is.na(neg.matches)

    score = sum(pos.matches) - sum(neg.matches) # true and false are
    #treated as 1 and 0 so they can be added

    return(score)

  }, pos.words, neg.words )

  scores.df = data.frame(score=scores, text=tweets)

  return(scores.df)

}
#Finally we will apply this function to find out big data popularity.

#Putting tweets in a df
test <- ldply(bigdata,function(t) as.data.frame(t) )
```

```
#Applying sentiment function
result <- score.sentiment(test$text,pos.words,neg.words)
#Checking the results

#Scores
summary(result$score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -2.000   0.000   0.000   0.044   0.000   2.000
```

```
#plotting
hist(result$score,col ="red", main ="Score of tweets", ylab = " Count of tweets")
```




```
#counting
count(result$score)
```

```
##    x freq
## 1 -2    2
## 2 -1   36
## 3  0  408
## 4  1   46
## 5  2    8
```