``

TWITTER SENTIMENT ANALYSIS

Twitter is an online news and social networking service that enables users to send and read short 140-character messages called "tweets". Registered users can read and post tweets, but those who are unregistered can only read them. Hence Twitter is a public platform with a mine of public opinion of people all over the world and of all age categories. As of October 2016, Twitter has more than 315 million monthly active users​. Twitter Sentiment Analysis is the process of determining the emotional tone behind a series of words, used to gain an understanding of the attitudes, opinions and emotions expressed within an online mention.

Sentiment analysis is in demand because of its efficiency. Thousands of text documents can be processed for sentiment in seconds, compared to the hours it would take a team of people to manually complete. Because it is so efficient (and accurate – Semantic has 80% accuracy for English content) many businesses are adopting text and sentiment analysis and incorporating it into their processes.

Tweets are imported using R and the data is cleaned by removing emoticons and URLs. Lexical Analysis is used to predict the sentiment of tweets and subsequently express the opinion graphically through ggplots, histogram, pie chart and tables.

In this report, we have built model to analyse the people’s emotion on covid-19 variant Omicron.

Steps

1. Extraction of Tweets

(i) Create twitter application

(ii) twitteR - Provides an interface to the Twitter web API

(iii) ROAuth - ​R Interface for OAuth

(iv)​Create twitter authenticated credential object, It is done using consumer key, consumer secret, access token, access secret.

​2. Cleaning Tweets

3. Algorithms used

● Lexical Analysis:​ ​By comparing uni-grams to the pre-loaded word database, the tweet is assigned sentiment score - positive, negative or neutral and overall score is calculated.

4. Calculating Score

​We have presented the score of positive/negative emotion in the text. This calculated using simple arithmetic to understand the overall sentiment in more better manner.

5. Visualization

Histograms of positive, negative and overall score are found under the Histogram tab for graphically analysing the intensity of emotion in the tweeters.

Code

```r  
#List Of Packages Used  
library(RColorBrewer)   
library(tm)

## Loading required package: NLP

library(twitteR)  
library(ROAuth)  
library(plyr)

##   
## Attaching package: 'plyr'

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

library(stringr)  
library(base64enc)  
library(SnowballC)  
library(ggplot2)

##   
## Attaching package: 'ggplot2'

## The following object is masked from 'package:NLP':  
##   
## annotate

library(maps)

##   
## Attaching package: 'maps'

## The following object is masked from 'package:plyr':  
##   
## ozone

## R Markdown

consumer\_key = 'ABCDEFGHIJKLMONPQ'  
consumer\_secret = 'ABCDEFGHIJKLMONPQ'   
access\_token= 'ABCDEFGHIJKLMONPQ  
access\_secret = 'ABCDEFGHIJKLMONPQ'

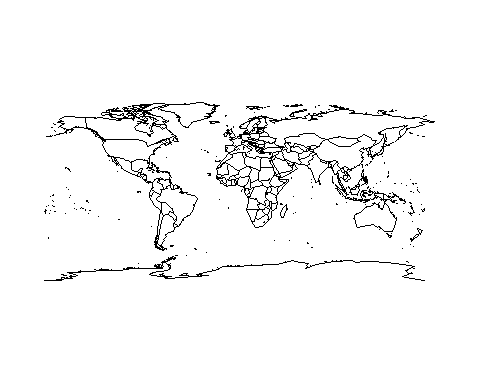
requestURL<- "https://api.twitter.com/oauth/request\_token"  
accessURL<- "https://api.twitter.com/oauth/access\_token"  
authURL<- "https://api.twitter.com/oauth/authorize"

setup\_twitter\_oauth(consumer\_key ,consumer\_secret,access\_token,access\_secret)

## [1] "Using direct authentication"

#search TWitter  
users<- searchTwitteR("#omicron", resultType="recent",n=300, lang="en")  
   
tweet.df = do.call("rbind",lapply(users,as.data.frame))

#Plotting data on map  
map('world')  
points(tweet.df$longitude,tweet.df$latitude, pch=20, cex=1, col="red")



#Viewing the data  
View(tweet.df)

#Reading sentiment analysis data from Txt document  
pos.words = scan('./positive-words.txt', what='character', comment.char=';')  
neg.words = scan('./negative-words.txt', what='character', comment.char=';')

#Appending some more words to actual words  
pos.words = c(pos.words, 'new','nice' ,'good', 'horizon')  
neg.words = c(neg.words, 'wtf', 'behind','feels', 'ugly', 'back','worse' , 'shitty', 'bad', 'no','freaking','sucks','horrible')

#converting Into dataFrame  
test <-ldply(users,function(t)t$toDataFrame())  
test

## text  
## 1 RT @ellymelly: If you lock the #unvaccinated up, you can't blame them for the #omicron outbreak. Have fun in your Catch22, #McClown.  
## 2 RT @Kaalateetham: Maybe in USA\n\nThis is not the scenario unfolding in India.\n\n#Omicron IMO will become endemic in India &amp; will cause automa…  
## 3 Watch: Does #Omicron lead to more hospitalisations? Here are some facts\n\nhttps://t.co/xxJgl2LE58  
## 4 RT @VivipPasanga: Finally online exam <U+0001F973><U+0001F60E> #Omicron #ONLINEEXAM #college https://t.co/ZCCtu3D93I  
## 5 RT @LawrenceSellin: Lawrence Sellin talks about the leaked Pentagon memo stating that #COVID19 was made in a Chinese lab with #USA funding,…  
## 6 RT @DrEricDing: Let this sink in—#Omicron many have peaked for adults… <U+0001F4CC>but not for younger kids—#COVID19 still spiking 5-11 &amp; not slowing…  
## 7 RT @Neelamk92304104: #postpone\_jpsc\_mains.kindly postpone jpsc main exam for few weeks due to surging cases of corona virus. #omicron.trave…  
## 8 RT @LawrenceSellin: Lawrence Sellin talks about the leaked Pentagon memo stating that #COVID19 was made in a Chinese lab with #USA funding,…  
## 9 RT @LawrenceSellin: Lawrence Sellin talks about the leaked Pentagon memo stating that #COVID19 was made in a Chinese lab with #USA funding,…  
## 10 RT @IndiaToday: The US facing shortage of medical staff amid surge in Covid cases.\n(@Milan\_reports) \n\n#COVID360 #Coronavirus #Omicron #Unit…  
<U+0001F606><U+0001F602><U+0001F3C3><U+0001F3FB><U+200D><U+2642><U+FE0F>\n\n#ONLINEEXAM #Omicron #TNCORONA https://t.co/NbJEBDLsSe  
## favorited favoriteCount replyToSN created truncated  
## 1 FALSE 0 <NA> 2022-01-21 07:40:19 FALSE  
## 2 FALSE 0 <NA> 2022-01-21 07:40:17 FALSE  
## 3 FALSE 0 <NA> 2022-01-21 07:40:06 FALSE  
## 4 FALSE 0 <NA> 2022-01-21 07:40:05 FALSE  
## 5 FALSE 0 <NA> 2022-01-21 07:40:00 FALSE  
## 6 FALSE 0 <NA> 2022-01-21 07:39:56 FALSE  
## 7 FALSE 0 <NA> 2022-01-21 07:39:49 FALSE  
## 8 FALSE 0 <NA> 2022-01-21 07:39:43 FALSE  
## 9 FALSE 0 <NA> 2022-01-21 07:39:33 FALSE  
## 10 FALSE 0 <NA> 2022-01-21 07:39:30 FALSE  
## replyToSID id replyToUID  
## 1 <NA> 1484430643220324353 <NA>  
## 2 <NA> 1484430634580058112 <NA>  
## 3 <NA> 1484430590606655489 <NA>  
## 4 <NA> 1484430584135176192 <NA>  
## 5 <NA> 1484430563230773248 <NA>  
## 6 <NA> 1484430547141087233 <NA>  
## 7 <NA> 1484430516762071047 <NA>  
## 8 <NA> 1484430492208230400 <NA>  
## 9 <NA> 1484430451649703937 <NA>  
## 10 <NA> 1484430438538297347 <NA>  
  
statusSource  
## 1 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>  
## 2 <a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>  
## 3 <a href="http://www.socialflow.com" rel="nofollow">SocialFlow</a>  
## 4 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>  
## 5 <a href="http://twitter.com/#!/download/ipad" rel="nofollow">Twitter for iPad</a>  
## 6 <a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>  
## 7 <a href="https://mobile.twitter.com" rel="nofollow">Twitter Web App</a>  
## 8 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>  
## 9 <a href="https://mobile.twitter.com" rel="nofollow">Twitter Web App</a>  
## 10 <a href="https://mobile.twitter.com" rel="nofollow">Twitter Web App</a>  
## screenName retweetCount isRetweet retweeted longitude latitude  
## 1 RFVNocon 193 TRUE FALSE NA NA  
## 2 indianmonk50 10 TRUE FALSE NA NA  
## 3 businessline 0 FALSE FALSE NA NA  
## 4 ganesh94853203 15 TRUE FALSE NA NA  
## 5 sandowman 15 TRUE FALSE NA NA

score.sentiment = function(sentences, pos.words, neg.words, .progress='none')  
{  
 require(plyr)  
 require(stringr)  
 scores = laply(sentences, function(sentence, pos.words, neg.words) {  
 sentence = gsub('[[:punct:]]', '', sentence)  
 sentence = gsub('[[:cntrl:]]', '', sentence)  
 sentence = gsub('\\d+', '', sentence)  
 sentence = tolower(sentence)  
 word.list = str\_split(sentence, '\\s+')  
 words = unlist(word.list)  
 pos.matches = match(words, pos.words)  
 neg.matches = match(words, neg.words)  
 pos.matches = !is.na(pos.matches)  
 neg.matches = !is.na(neg.matches)  
 score <- sum(pos.matches) - sum(neg.matches)  
 return(score)  
 }, pos.words, neg.words, .progress=.progress )  
   
 scores.df = data.frame(score=scores, text=sentences)  
 return(scores.df)  
}

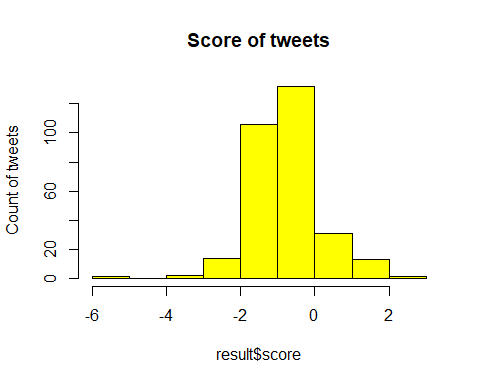
result <- score.sentiment(test$text,pos.words,neg.words)  
result

## score  
## 1 -1  
## 2 0  
## 3 1  
## 4 0  
## 5 0  
## 6 -1  
## 7 -1  
## 8 0  
## 9 0  
## 10 -1  
  
  
## text  
## 1 RT @ellymelly: If you lock the #unvaccinated up, you can't blame them for the #omicron outbreak. Have fun in your Catch22, #McClown.  
## 2 RT @Kaalateetham: Maybe in USA\n\nThis is not the scenario unfolding in India.\n\n#Omicron IMO will become endemic in India &amp; will cause automa…  
## 3 Watch: Does #Omicron lead to more hospitalisations? Here are some facts\n\nhttps://t.co/xxJgl2LE58  
## 4 RT @VivipPasanga: Finally online exam <U+0001F973><U+0001F60E> #Omicron #ONLINEEXAM #college https://t.co/ZCCtu3D93I  
## 5 RT @LawrenceSellin: Lawrence Sellin talks about the leaked Pentagon memo stating that #COVID19 was made in a Chinese lab with #USA funding,…  
## 6 RT @DrEricDing: Let this sink in—#Omicron many have peaked for adults… <U+0001F4CC>but not for younger kids—#COVID19 still spiking 5-11 &amp; not slowing…  
## 7 RT @Neelamk92304104: #postpone\_jpsc\_mains.kindly postpone jpsc main exam for few weeks due to surging cases of corona virus. #omicron.trave…  
## 8 RT @LawrenceSellin: Lawrence Sellin talks about the leaked Pentagon memo stating that #COVID19 was made in a Chinese lab with #USA funding,…  
## 9 RT @LawrenceSellin: Lawrence Sellin talks about the leaked Pentagon memo stating that #COVID19 was made in a Chinese lab with #USA funding,…  
## 10 RT @IndiaToday: The US facing shortage of medical staff amid surge in Covid cases.\n(@Milan\_reports) \n\n#COVID360 #Coronavirus #Omicron #Unit…

#summarlizing data  
summary(result$score)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -6.0000 -1.0000 0.0000 -0.2867 0.0000 3.0000

#Histogram  
hist(result$score,col="yellow", main="Score of tweets",ylab=" Count of tweets")

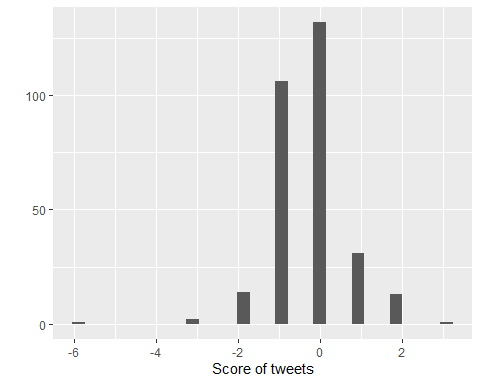


#Count No of Tweets  
count(result$score)

## x freq  
## 1 -6 1  
## 2 -3 2  
## 3 -2 14  
## 4 -1 106  
## 5 0 132  
## 6 1 31  
## 7 2 13  
## 8 3 1

#ploting the tweets on qplot  
qplot(result$score,xlab = "Score of tweets")

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



require(wordcloud)

## Loading required package: wordcloud

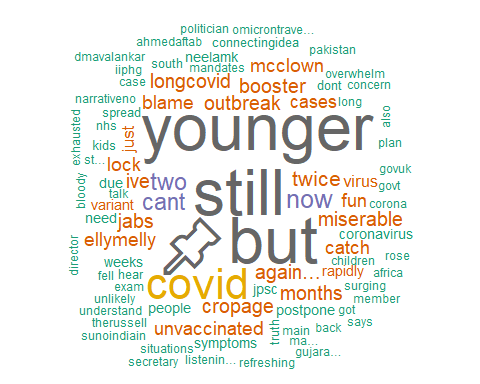
negativeTweets = subset(result, score < 0)$text

corpus = Corpus(VectorSource(negativeTweets))  
# corpus = Corpus(VectorSource(cmail))  
# create term-document matrix  
tdm = TermDocumentMatrix(  
 corpus,  
 control = list(  
 wordLengths=c(3,20),  
 removePunctuation = TRUE,  
 stopwords = c("the", "a", stopwords("english")),  
 removeNumbers = TRUE, tolower = TRUE) )

# convert as matrix  
tdm = as.matrix(tdm)

# get word counts in decreasing order  
word\_freqs = sort(rowSums(tdm), decreasing=TRUE)   
word\_freqs = word\_freqs[-(1:12)]  
# create a data frame with words and their frequencies  
dm = data.frame(word=names(word\_freqs), freq=word\_freqs)

#Plot corpus in a clored graph; need RColorBrewer package  
  
wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order=FALSE, colors=brewer.pal(8, "Dark2"))



positiveTweets = subset(result, score > 0)$text

corpus = Corpus(VectorSource(positiveTweets))  
# corpus = Corpus(VectorSource(cmail))  
# create term-document matrix  
tdm = TermDocumentMatrix(  
 corpus,  
 control = list(  
 wordLengths=c(3,20),  
 removePunctuation = TRUE,  
 stopwords = c("the", "a", stopwords("english")),  
 removeNumbers = TRUE, tolower = TRUE) )

# convert as matrix  
tdm = as.matrix(tdm)

# get word counts in decreasing order  
word\_freqs = sort(rowSums(tdm), decreasing=TRUE)   
word\_freqs = word\_freqs[-(1:12)]  
# create a data frame with words and their frequencies  
dm = data.frame(word=names(word\_freqs), freq=word\_freqs)

wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order=FALSE, colors=brewer.pal(8, "Dark2"))

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : suggests could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : despicable could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : effective could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : galnooks could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : mitigat… could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : rising could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : india could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : many could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : delhi could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : people could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : health could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : competent could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : intelligent could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : in… could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : leader could not be fit on page. It will not be plotted.

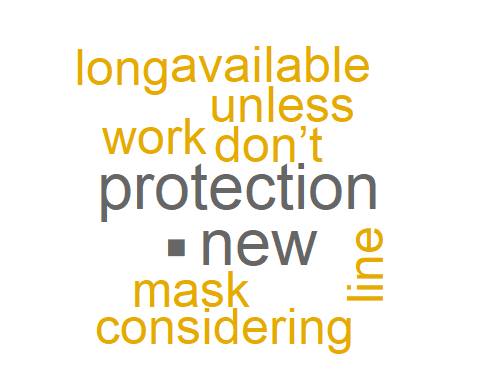
## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : national could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : need could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : needed could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : really could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : ishacabs could not be fit on page. It will not be plotted.



NeutralTweets = subset(result, score = 0)$text

corpus = Corpus(VectorSource(NeutralTweets))  
# corpus = Corpus(VectorSource(cmail))  
# create term-document matrix  
tdm = TermDocumentMatrix(  
 corpus,  
 control = list(  
 wordLengths=c(3,20),  
 removePunctuation = TRUE,  
 stopwords = c("the", "a", stopwords("english")),  
 removeNumbers = TRUE, tolower = TRUE) )

# convert as matrix  
tdm = as.matrix(tdm)

# get word counts in decreasing order  
word\_freqs = sort(rowSums(tdm), decreasing=TRUE)   
word\_freqs = word\_freqs[-(1:12)]  
# create a data frame with words and their frequencies  
dm = data.frame(word=names(word\_freqs), freq=word\_freqs)

wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order=FALSE, colors=brewer.pal(8, "Dark2"))

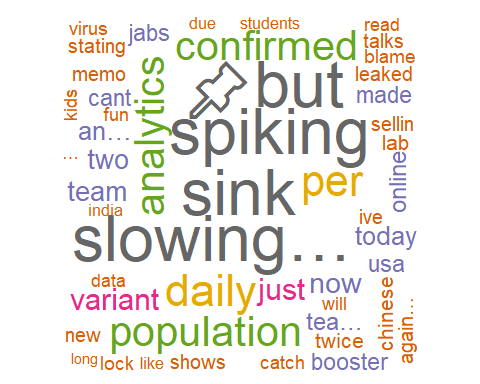
## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : younger could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : county could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : insights could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : njoyflyfishing could not be fit on page. It will not be plotted.

## Warning in wordcloud(head(dm$word, 100), head(dm$freq, 100), random.order =  
## FALSE, : deaths could not be fit on page. It will not be plotted.



Conclusion

Nowadays, sentiment analysis or opinion mining is a hot topic in machine learning. We are still far to detect the sentiments of texts very accurately because of the complexity in the English language and even more if we consider other languages such as Kannada. In this project we tried to show the basic way of classifying tweets into positive or negative category using lexical analysis.