WSMA\_Group\_Assignment.R

Group # 3

Sat Aug 26 21:03:32 2017

###################################################################################################################################  
# Group Assignent for WSMA  
# Author - PGPBABIH - Group #3  
###################################################################################################################################  
# Clean up and Garbage collection   
rm(list=ls(all=TRUE))  
gc(reset=T)

## used (Mb) gc trigger (Mb) max used (Mb)  
## Ncells 471813 25.2 940480 50.3 471813 25.2  
## Vcells 855689 6.6 1650153 12.6 855689 6.6

###################################################################################################################################  
# Load the required libraries for R code  
###################################################################################################################################  
library(twitteR)

## Warning: package 'twitteR' was built under R version 3.4.1

library(tm)

## Loading required package: NLP

library(ggplot2)

##   
## Attaching package: 'ggplot2'

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

library(ROAuth)

## Warning: package 'ROAuth' was built under R version 3.4.1

library(wordcloud)

## Loading required package: RColorBrewer

library(lubridate)

## Warning: package 'lubridate' was built under R version 3.4.1

##   
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':  
##   
## date

library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':  
##   
## hour, isoweek, mday, minute, month, quarter, second, wday,  
## week, yday, year

library(SnowballC)  
library(syuzhet)

## Warning: package 'syuzhet' was built under R version 3.4.1

library(stringr)  
  
###################################################################################################################################  
# User defined function for analysis  
###################################################################################################################################  
# encode tweet source as iPhone, iPad, Android or Web  
enodeSource <- function(x) {  
 if(x=="<a href=\"http://twitter.com/download/iphone\" rel=\"nofollow\">Twitter for iPhone</a>"){  
 gsub("<a href=\"http://twitter.com/download/iphone\" rel=\"nofollow\">Twitter for iPhone</a>", "iphone", x,fixed=TRUE)  
 }else if(x=="<a href=\"http://twitter.com/#!/download/ipad\" rel=\"nofollow\">Twitter for iPad</a>"){  
 gsub("<a href=\"http://twitter.com/#!/download/ipad\" rel=\"nofollow\">Twitter for iPad</a>","ipad",x,fixed=TRUE)  
 }else if(x=="<a href=\"http://twitter.com/download/android\" rel=\"nofollow\">Twitter for Android</a>"){  
 gsub("<a href=\"http://twitter.com/download/android\" rel=\"nofollow\">Twitter for Android</a>","android",x,fixed=TRUE)  
 } else if(x=="<a href=\"http://twitter.com\" rel=\"nofollow\">Twitter Web Client</a>"){  
 gsub("<a href=\"http://twitter.com\" rel=\"nofollow\">Twitter Web Client</a>","Web",x,fixed=TRUE)  
 } else if(x=="<a href=\"http://www.twitter.com\" rel=\"nofollow\">Twitter for Windows Phone</a>"){  
 gsub("<a href=\"http://www.twitter.com\" rel=\"nofollow\">Twitter for Windows Phone</a>","windows phone",x,fixed=TRUE)  
 }else if(x=="<a href=\"http://dlvr.it\" rel=\"nofollow\">dlvr.it</a>"){  
 gsub("<a href=\"http://dlvr.it\" rel=\"nofollow\">dlvr.it</a>","dlvr.it",x,fixed=TRUE)  
 }else if(x=="<a href=\"http://ifttt.com\" rel=\"nofollow\">IFTTT</a>"){  
 gsub("<a href=\"http://ifttt.com\" rel=\"nofollow\">IFTTT</a>","ifttt",x,fixed=TRUE)  
 }else if(x=="<a href=\"http://www.facebook.com/twitter\" rel=\"nofollow\">Facebook</a>"){  
 gsub("<a href=\"http://www.facebook.com/twitter\" rel=\"nofollow\">Facebook</a>","facebook",x,fixed=TRUE)  
 }else {  
 "others"  
 }  
}  
#  
# Sentiment analysis  
encodeSentiment <- function(x) {  
 if(x <= -0.5){  
 "very negative"  
 }else if(x > -0.5 & x < 0){  
 "negative"  
 }else if(x > 0 & x < 0.5){  
 "positive"  
 }else if(x >= 0.5){  
 "very positive"  
 }else {  
 "neutral"  
 }  
}  
  
###################################################################################################################################  
# Twitter Authentication for extracting tweets.  
# Creating a Twitter Application to get Twitter Authorization parameters.  
###################################################################################################################################  
#OAuth Protocol - Unique for each twitter user  
api\_key <- "AEW6KsgM2wEeIKDJUxNxn9DyL"  
api\_secret <- "zJuFWeRouwAMS0OmSL8VlLTPTe60vJJQlH9jvJ1UeYB1WaSAKc"  
access\_token <- "900794816254103552-8VvMJOfJHVu7ms2umtG6QlGzouiQLZN"  
access\_token\_secret <- "y2xdd8MFWhMQWTuH1YiL5v0v2ie1SoXWKH4RiC2J2fTaT"  
  
#Authorization for the Twitter account  
setup\_twitter\_oauth(api\_key,api\_secret,access\_token,access\_token\_secret)

## [1] "Using direct authentication"

###################################################################################################################################  
# Extract tweets from Twitter.  
###################################################################################################################################  
# Set keyword for search  
SearchString <- "@Narendramodi"  
  
# Search Twitter for recent tweets for Modi  
tweets <- searchTwitter(SearchString, resultType="recent", since='2017-06-25', until='2017-08-25',n=4000, lang='en')  
length(tweets)

## [1] 4000

# convert to dataframe  
tweets\_all.df <- twListToDF(tweets)  
dim(tweets\_all.df)

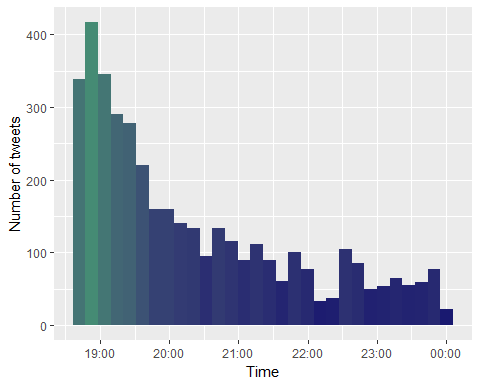
## [1] 4000 16

###################################################################################################################################  
#Data Cleaning on the text files [getTransformations()]  
###################################################################################################################################  
# Identiy & create text files to turn into a cloud  
tweets\_text <- sapply(tweets, function(x) x$getText())  
  
# Remove special characters  
nohandles <- str\_replace\_all(tweets\_text, "@\\w+", "")  
  
#Create a corpus from the collection of text files.  
tweets\_text\_corpus <- Corpus(VectorSource(nohandles))  
  
#Remove punctuation.  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, removePunctuation)  
  
#Transform text to lower case.  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, content\_transformer(tolower))  
  
#Remove stopwords.  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, removeWords, stopwords("english"))  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, removeWords, c("amp"))  
  
#Remove Numbers  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, removeNumbers)  
  
#Remove whitespace  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, stripWhitespace)  
  
#Remove URL's from text  
removeURL <- function(x) gsub("http[[:alnum:]]\*", "", x)  
tweets\_text\_corpus <- tm\_map(tweets\_text\_corpus, content\_transformer(removeURL))  
  
###################################################################################################################################  
# Twitter data exploration  
###################################################################################################################################  
#Create a variable called date and change to character:  
date <- Sys.Date()  
date <- as.character(date)  
name <- paste(date,".RData")  
  
#Finally we save the Rdata using as name the date from the download  
#The script below is to download the data from Twitter  
save(tweets\_all.df, file =name)  
  
# Cleanup/transformation  
tweets\_all.df$text <- sapply(tweets\_all.df$text,function(x) iconv(x,to='UTF-8'))  
tweets\_all.df$created <- ymd\_hms(tweets\_all.df$created)  
  
# Missing values per column basis  
sapply(tweets\_all.df, function(x) sum(is.na(x)))

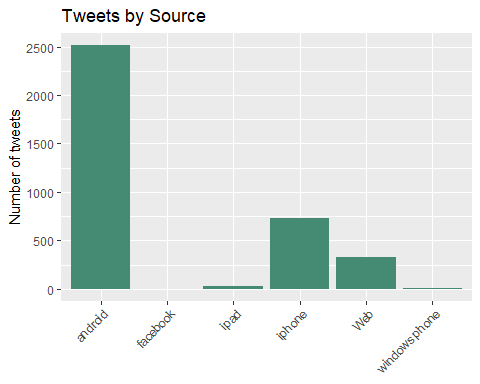
## text favorited favoriteCount replyToSN created   
## 0 0 0 3336 0   
## truncated replyToSID id replyToUID statusSource   
## 0 3511 0 3336 0   
## screenName retweetCount isRetweet retweeted longitude   
## 0 0 0 0 4000   
## latitude   
## 4000

# Plot of tweets by time  
ggplot(data = tweets\_all.df, aes(x = created)) +  
 geom\_histogram(aes(fill = ..count..)) +  
 theme(legend.position = "none") +  
 xlab("Time") + ylab("Number of tweets") +   
 scale\_fill\_gradient(low = "midnightblue", high = "aquamarine4")

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



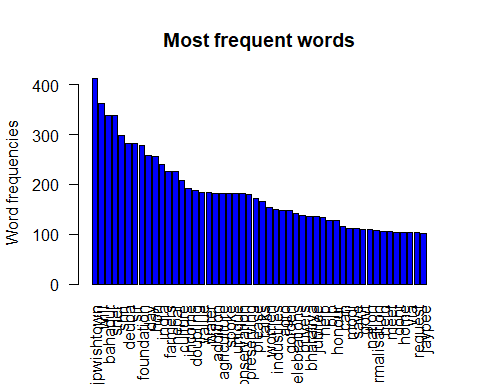
# plot tweets by source system (android, iphone, web, etc)  
tweets\_all.df$tweetSource = sapply(tweets\_all.df$statusSource,  
 function(sourceSystem) enodeSource(sourceSystem))  
  
  
ggplot(tweets\_all.df[tweets\_all.df$tweetSource != 'others',], aes(tweetSource)) +  
 geom\_bar(fill = "aquamarine4") +   
 theme(legend.position="none",   
 axis.title.x = element\_blank(),   
 axis.text.x = element\_text(angle = 45, hjust = 1)) +  
 ylab("Number of tweets") +   
 ggtitle("Tweets by Source")



###################################################################################################################################  
#Build document matrix table containing the frequency of the words.  
#Column names are words and row names are documents.  
###################################################################################################################################  
#Document matrix table  
tweets\_2 <- TermDocumentMatrix(tweets\_text\_corpus)  
tweets\_2 <- as.matrix(tweets\_2)  
tweets\_2 <- sort(rowSums(tweets\_2),decreasing=TRUE)  
  
#Converting words to dataframe  
tweets\_2 <- data.frame(word = names(tweets\_2),freq=tweets\_2)  
  
#The frequency table of words  
head(tweets\_2, 10)

## word freq  
## savejpwishtown savejpwishtown 412  
## will will 362  
## bahadur bahadur 339  
## sher sher 339  
## shri shri 299  
## deuba deuba 282  
## sir sir 281  
## foundation foundation 278  
## day day 257  
## hon hon 256

#Word frequencies Plot for the first 10 frequent words  
barplot(tweets\_2[1:50,]$freq, las = 2, names.arg = tweets\_2[1:50,]$word,col ="blue",   
 main ="Most frequent words",ylab = "Word frequencies")



###################################################################################################################################  
# Word Cloud  
###################################################################################################################################  
set.seed(16702)  
wordcloud(tweets\_text\_corpus,min.freq=1,max.words=80,scale=c(2.2,1),   
 colors=brewer.pal(8, "Dark2"), random.color=T, random.order=F)

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : ssccglnormalisation could not be fit on page. It will not  
## be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : country could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : pictures could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : sacrifice could not be fit on page. It will not be  
## plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : childrens could not be fit on page. It will not be  
## plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : president could not be fit on page. It will not be  
## plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : sharing could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : celebrating could not be fit on page. It will not be  
## plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : ramrahimverdict could not be fit on page. It will not be  
## plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : homage could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : people could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : remarks could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : must could not be fit on page. It will not be plotted.

## Warning in wordcloud(tweets\_text\_corpus, min.freq = 1, max.words = 80,  
## scale = c(2.2, : need could not be fit on page. It will not be plotted.

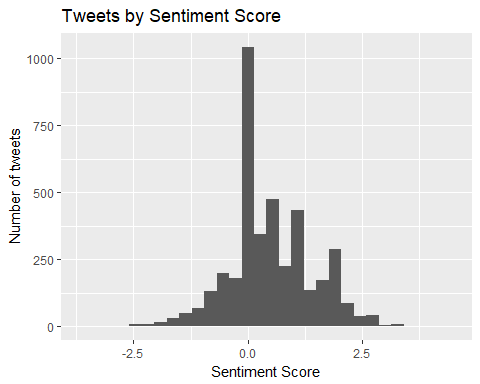


###################################################################################################################################  
#Sentiment Analysis - very positive, positive, negative, very negative and   
#neutral  
###################################################################################################################################  
#Get Tweet sentiment using syuzhet package  
tweetSentiments <- get\_sentiment (tweets\_all.df$text,method = "syuzhet")  
tweets <- cbind(tweets\_all.df, tweetSentiments)  
tweets$sentiment <- sapply(tweets$tweetSentiments,encodeSentiment)  
  
# table data for tweet sentiments  
table(tweets$sentiment)

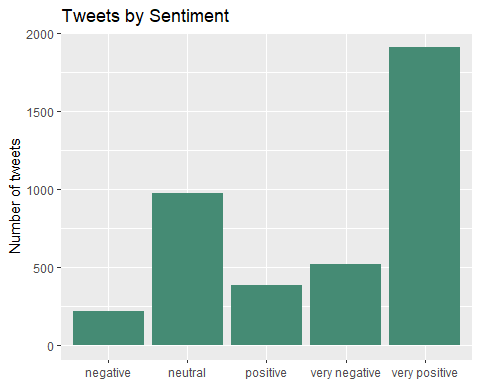
##   
## negative neutral positive very negative very positive   
## 219 976 384 515 1906

# visualization  
qplot(tweets$tweetSentiments) + theme(legend.position="none")+  
 xlab("Sentiment Score") +  
 ylab("Number of tweets") +   
 ggtitle("Tweets by Sentiment Score")

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



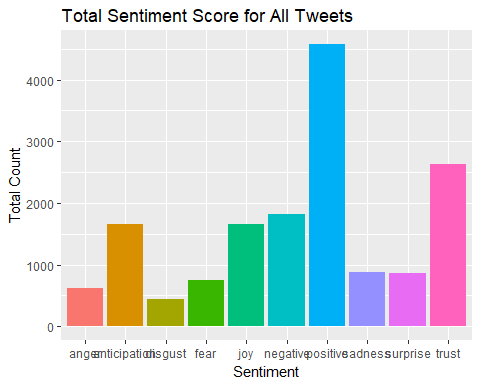
ggplot(tweets, aes(sentiment)) +  
 geom\_bar(fill = "aquamarine4") +   
 theme(legend.position="none", axis.title.x = element\_blank()) +  
 ylab("Number of tweets") +   
 ggtitle("Tweets by Sentiment")



###################################################################################################################################  
# Get Emotions and Valence from NRC Dictionary. the emotions are :  
# "anger", "anticipation", "disgust", "fear", "joy", "sadness", "surprise",   
# "trust", "negative", "positive."  
###################################################################################################################################  
tweetSentiments <- get\_nrc\_sentiment(tweets\_all.df$text)  
tweets <- cbind(tweets\_all.df, tweetSentiments)  
  
# Take out the emotions  
sentimentTotals <- data.frame(colSums(tweets[,c(18:27)]))  
  
names(sentimentTotals) <- "count"  
sentimentTotals <- cbind("sentiment" = rownames(sentimentTotals), sentimentTotals)  
rownames(sentimentTotals) <- NULL  
  
#Emotions count table   
sentimentTotals

## sentiment count  
## 1 anger 612  
## 2 anticipation 1664  
## 3 disgust 438  
## 4 fear 745  
## 5 joy 1664  
## 6 sadness 874  
## 7 surprise 862  
## 8 trust 2629  
## 9 negative 1818  
## 10 positive 4584

# Visualization  
ggplot(data = sentimentTotals, aes(x = sentiment, y = count)) +  
 geom\_bar(aes(fill = sentiment), stat = "identity") +  
 theme(legend.position = "none") +  
 xlab("Sentiment") + ylab("Total Count") + ggtitle("Total Sentiment Score for All Tweets")



###################################################################################################################################  
# Interpretation  
###################################################################################################################################  
# MAximum tweets where through android OS, stating source was mostly android based pj=hone or laptop/computer   
# We analysed 4000 tweets where most of the tweets scored near '0' (neutral) or more (positive)  
# 'savejpwishtown' tops the list of frequent words owing to Jaypee builders bankrupcy  
# Few words like 'will', 'can' made it to the list but are not helpful  
# Very Positive (47.65%) and neutral (24.4%) tweets have got more share compared to other sentiments  
# Emotions like 'anticipation', 'joy', 'trust' lead the chart for sentiments.