Analysis of Tweets Data on Starbucks

Keer Jiang

12/15/2017

I. Introduction

In this generation of social media, almost everyone has a social media account to interact with friends of get to know news everyday. People express their feelings on any topics, for examples, what course do they take of what kind of coffee they drink in the morning. There are also several functions enables different kinds of interaction among users, like replying, following and forwading that further enlarges the impact of social media on users. For the huge number of users and the highly interacitve property, social media data plays an increasingly important rule in business analysis and financial analysis.

Twitter is one of the most popular social media which has its API available to the public that makes data analysis of social media data much easier. By using twitter API, one can connect to Twitter database to get data of a specific location during a specific time period on a chosen topic.

Starbucks is an American coffee company and coffeehouse chain that is very popular across the country and more than 13,107 Starbucks in the United States in total. In this project, I choose "Starbucks" as the keyword(or topic) to explore the twitter activity and sentiment that relevant to Starbucks. It is meaningful to see what attituden people hold towards this popular coffeehouse chain company.

This project will be arranged as following: First of all, I randomly select 100 places in the US, and take a look at the number of tweets in a circle of radius equals 100 miles which is centered on those places through a map and then a geolocation density plot. Secondly, I do an emojis analysis to show what emoji people prefer to use when their tweets are about Starbucks. Then, I use bar chart and word cloud to show what words are frequently used when people are talking about Starbucks. Furthermore, I do a sentiment analysis that analyzes the attitude people hold towards Starbucks and then use bootstrap to analyze the different effect of different sentimene on social media. Moreover, I do a k-means analysis and then a hierarchical clustering to get more details about the most frequently mentioned words in the tweets about Starbucks.

Shiny app is used for visualization here. In my Shiny app, you can adjust the minimum word frequency and maximum number of words to get different word cloud. This link leads to my shinyapp website: https://keer.shinyapps.io/final2/

II. Data

Firstly, I set up twitter API by linking URL and KEY to the server. Secondly, I collect 25568 tweetss in total that mention Starbucks and the dataset contains 7 variables.

```
###### GRAB TWEETS, PROCESS, AND WRITE TO DISK ######
# authenticate with twitter: get your credentials by creating an app at apps.twitter.com
```

```r
#api_key <-     "an6bEUywBhzxpTfltpelpvob9"
#api_secret <- "qCiSpxcwKxewngAwT4AVhHRyOvRcmepDWXxqoUpLlM3MEIXwdZ"
#access_token <- "927637333901545472-YSigYiWOhvy2X7qFmHJ9FDQ97KQhYPr"
#access_token_secret <- "VsR1DhFvt7Weyr5DvE2ibLAnTVYs4EiGnV4TAW5XXX3ex"
#setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)
```

```r
#read all datasets needed
all_states<-read.csv("plot_state5.csv")
coordinates<-read.csv("plot_data5.csv")
coordinates1<-coordinates[which(coordinates$number.of.tweets>60),]
tweets<-read.csv("tweets.csv")
starbucks<-read.csv("tweets.cleaned3_starbucks.csv")
```

III. Analysis

1. Variables

There are 7 variables in the original dataset from Twitter in total, which are text, created, url, latitude, longtitude, retweets and hashtag. The following table shows all the variables and their descriptions.

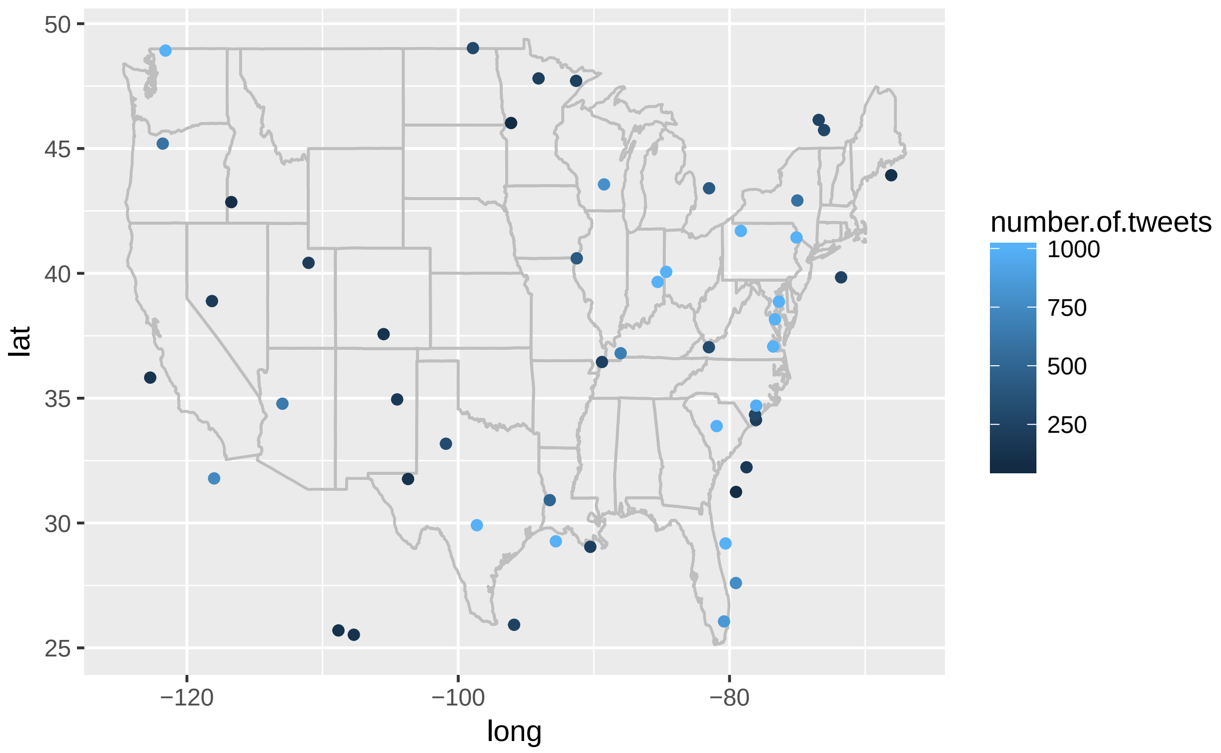
Table 1: Variables and Corresponding Descriptions

Variable Name	Description
text	tweets text
created	published time of the tweet
url	url of each tweet
latitude	user location latitude
longtitude	user location longtitude

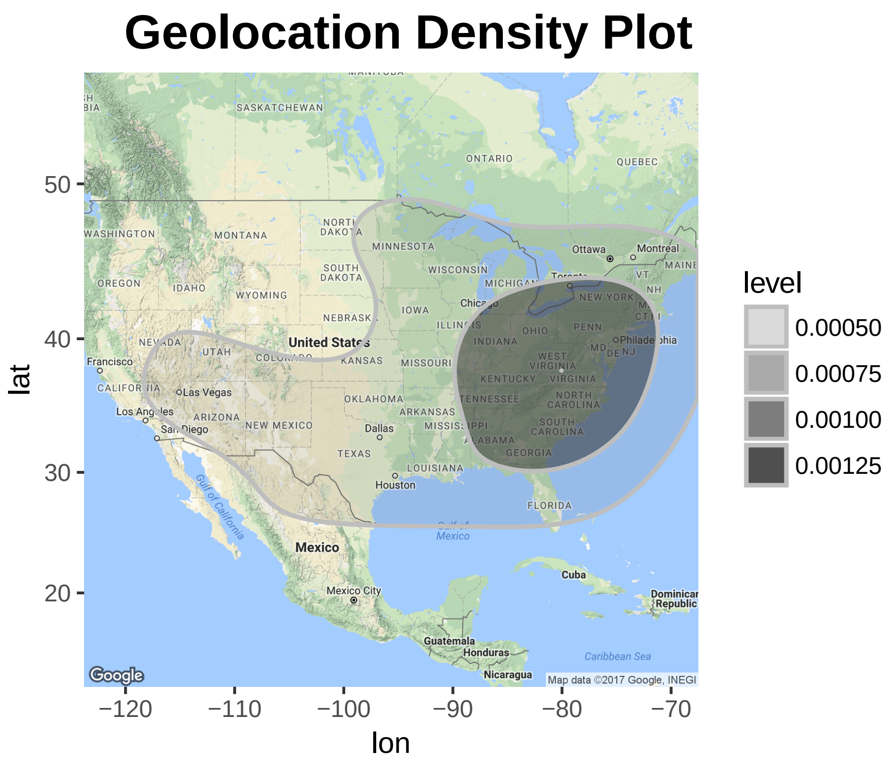
2. Map

In this part, I draw two plots to show the twitter activity distribution in the US. In the first plot, each point represents the number of tweets in the circle of radius equals 100 miles that centered in that point. The lighter the point is, the more tweets in the area. According to the plot we can see that most of the light points ditributed in east coast, Texas and Carlifonia, and there are mainly dark blue points in the middle area. This plot coincides with the fact that most of the big cities are distibuted in east coast (e.g. Boston, NYC, Orlando etc) and west coast where have more Starbucks chain stores. In the geolocation density plot, it shows the tweets density in the US. The darker the circle, the higher tweets density in the corresponding area. And if the density is too low then there will not be such grey circle. Accoording to the density plot, we can get the similar conclusion as in the previous plot. The tweets density in east coast is much higher than other area since there are more big cities in this area.

```r
#plot 1: number of tweets plot
p <- ggplot()
p <- p + geom_polygon( data=all_states, aes(x=long, y=lat, group = group),colour="grey", fill=NA )
p<- p + geom_point( data=coordinates1, aes(x=long, y=lat,color=number.of.tweets
  )) + scale_size(name="# of tweets")
p
```



```
#plot 2: Density plot based on Google map
map<-get_map(location = 'united states',zoom = 4,maptype = 'terrain',source = 'google',color = 'color')
ggmap(map)+ggtitle("   Geolocation Density Plot")+
  theme(plot.title = element_text(size = 18,face = "bold"))+stat_density2d(aes(x=long,y=lat,alpha=..leve
```



3. Emoji Analysis

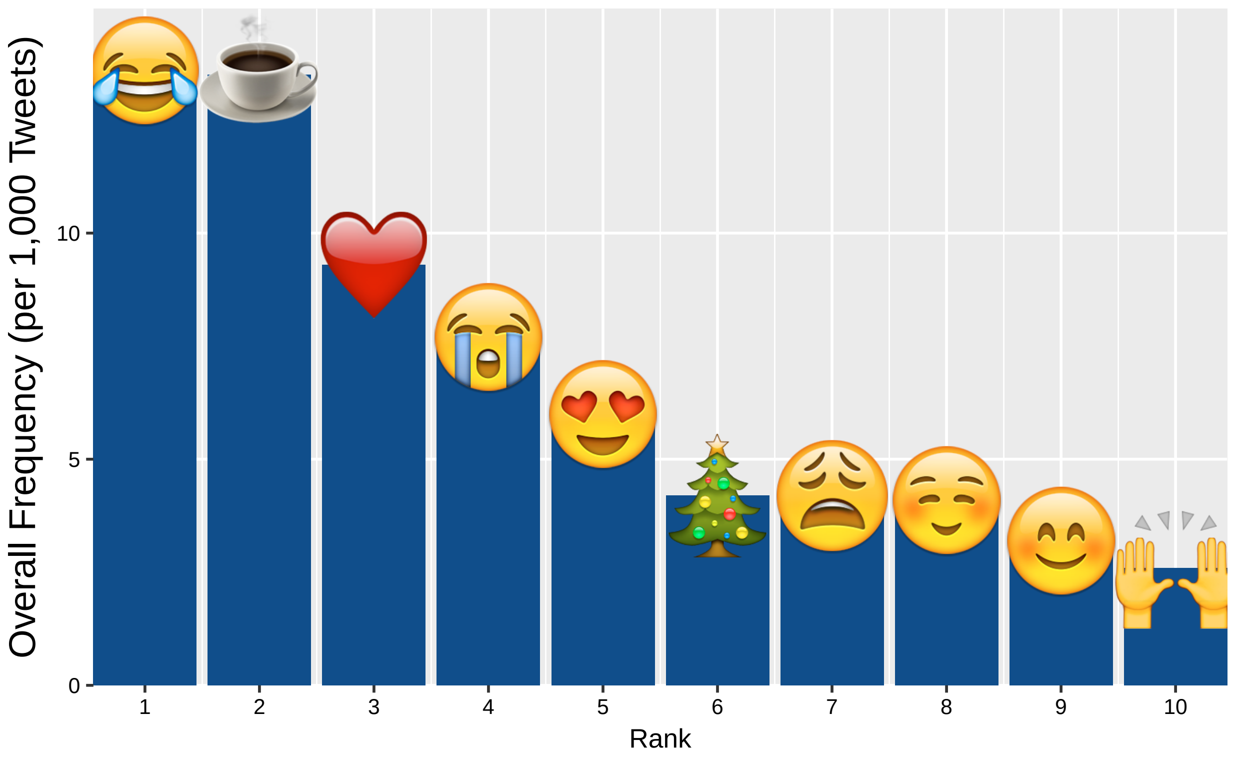
People frequently use emojis in Twitter to express their feelings that can not be expressed in words which make the tweets more interesting. And thus we can explore what type of emojis people prefer to use about one certain topic. In this part, I conduct an emojis analysis to find out the most frequently used emojis when people talk about Starbucks and draw two plots to show the results in a clear way.

```
fnames <- c(
  'tweets.cleaned3_starbucks'
);
fnames <- paste0(fnames, '.csv');
df <- do.call(rbind.fill, lapply(fnames, read.csv));
df$username <- substr(substr(df$url, 21, nchar(as.character(df$url))), 1, nchar(substr(df$url, 21, ncha
tweets.full <- df;
tweets.full$X <- NULL;
tweets.full$z <- 1;
#### sanity checking
tweets.full$created <- as.POSIXct(tweets.full$created);
## dedupe dataset by url
tweets.dupes <- tweets.full[duplicated(tweets.full$url), ];
# test <- subset(tweets.full, url %in% tweets.dupes$url); test <- test[with(test, order(url)), ];
tweets <- tweets.full[!duplicated(tweets.full$url), ];
tweets <- arrange(tweets, url); row.names(tweets) <- NULL;
tweets$tweetid <- as.numeric(row.names(tweets));
tweets.final <- tweets;
```

```
emdict.la <- read.csv('emoticon_conversion_noGraphic.csv', header = F)
emdict.la <- emdict.la[-1, ];
row.names(emdict.la) <- NULL;
names(emdict.la) <- c('unicode', 'bytes', 'name');
emdict.la$emojiid <- row.names(emdict.la);
emdict.jpb <- read.csv('emDict.csv', header = F)
emdict.jpb <- emdict.jpb[-1, ];
row.names(emdict.jpb) <- NULL;
names(emdict.jpb) <- c('name', 'bytes', 'rencoding');
emdict.jpb$name <- tolower(emdict.jpb$name);
emdict.jpb$bytes <- NULL;
## merge dictionaries
emojis <- merge(emdict.la, emdict.jpb, by = 'name');
emojis$emojiid <- as.numeric(emojis$emojiid);
emojis <- arrange(emojis, emojiid);
###### FIND TOP EMOJIS FOR A GIVEN SUBSET OF THE DATA
tweets <- tweets.final;
# tweets <- subset(tweets.final, hashtag %in% c('#womensmarch'));
## create full tweets by emojis matrix
df.s <- matrix(NA, nrow = nrow(tweets), ncol = ncol(emojis));
system.time(df.s <- sapply(emojis$rencoding, regexpr, tweets$text, ignore.case = T, useBytes = T));
rownames(df.s) <- 1:nrow(df.s); colnames(df.s) <- 1:ncol(df.s); df.t <- data.frame(df.s); df.t$tweetid
# merge in hashtag data from original tweets dataset
df.a <- subset(tweets, select = c(tweetid, hashtag));
df.u <- merge(df.t, df.a, by = 'tweetid');
df.u$z <- 1;
df.u <- arrange(df.u, tweetid);
tweets.emojis.matrix <- df.u;
## create emoji count dataset
```

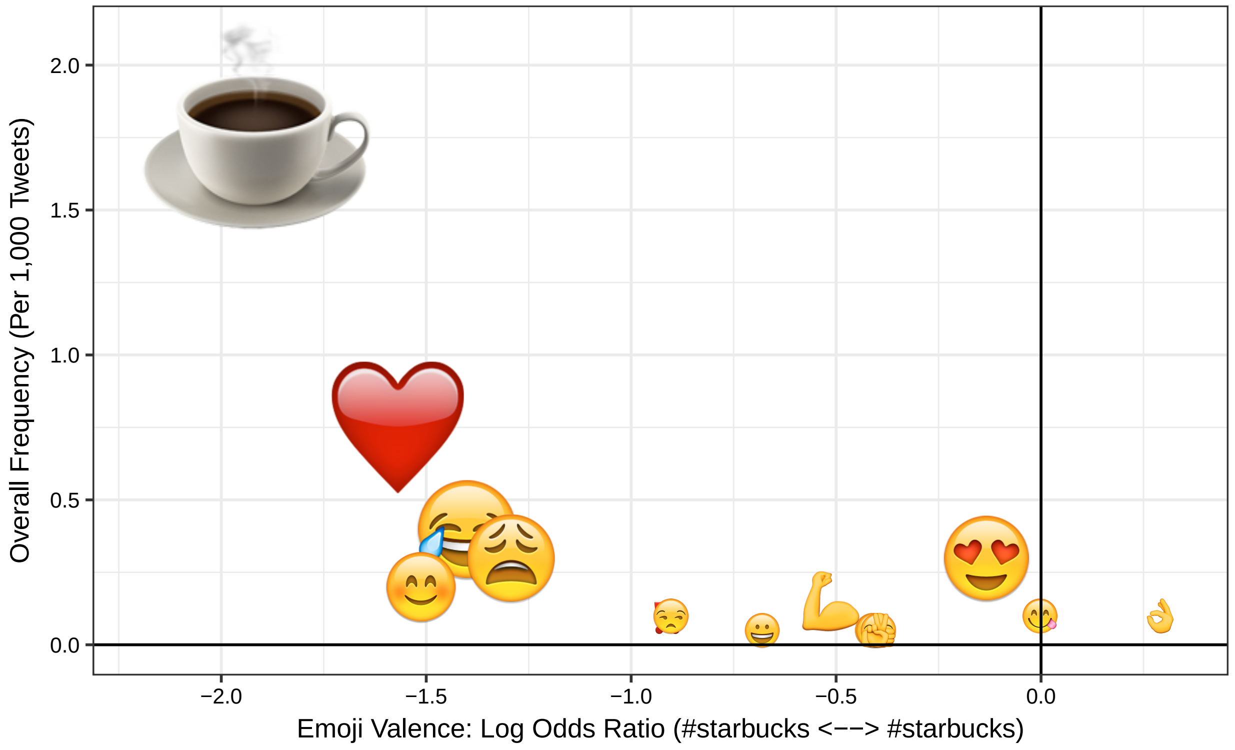
```r
df <- subset(tweets.emojis.matrix)[, c(2:843)]
count <- colSums(df > -1)
emojis.m <- cbind(count, emojis)
emojis.m <- arrange(emojis.m, desc(count))
emojis.count <- subset(emojis.m, count > 1)
emojis.count$dens <- round(1000 * (emojis.count$count / nrow(tweets)), 1)
emojis.count$dens.sm <- (emojis.count$count + 1) / (nrow(tweets) + 1)
emojis.count$rank <- as.numeric(row.names(emojis.count))
emojis.count.p <- subset(emojis.count, select = c(name, dens, count, rank))
# print summary stats
subset(emojis.count.p, rank <= 10);
num.tweets <- nrow(tweets);
df.t <- rowSums(tweets.emojis.matrix[, c(2:843)] > -1);
num.tweets.with.emojis <- length(df.t[df.t > 0]);
num.emojis <- sum(emojis.count$count);
round(100 * (num.tweets.with.emojis / num.tweets), 1);
##### MAKE BAR CHART OF TOP EMOJIS IN NEW DATASET
df.plot <- subset(emojis.count.p, rank <= 10);
xlab <- 'Rank';
ylab <- 'Overall Frequency (per 1,000 Tweets)';
```

```r
df.plot <- arrange(df.plot, name)
imgs <- lapply(paste0('emojis-master/2017.0206 emoji data science tutorial/ios_9_3_emoji_files/',df.plo
g <- lapply(imgs, grid::rasterGrob);

k <- 0.20 * (10/nrow(df.plot)) * max(df.plot$dens);
df.plot$xsize <- k; df.plot$ysize <- k; #df.plot$xsize <- k * (df.plot$dens / max(df.plot$dens)); df.pl
df.plot <- arrange(df.plot, name);
g1 <- ggplot(data = df.plot, aes(x = rank, y = dens)) +
  geom_bar(stat = 'identity', fill = 'dodgerblue4') +
  xlab(xlab) + ylab(ylab) +
  mapply(function(x, y, i) {
    annotation_custom(g[[i]], xmin = x-0.5*df.plot$xsize[i], xmax = x+0.5*df.plot$xsize[i],
                      ymin = y-0.5*df.plot$ysize[i], ymax = y+0.5*df.plot$ysize[i])},
    df.plot$rank, df.plot$dens, seq_len(nrow(df.plot))) +
  scale_x_continuous(expand = c(0, 0), breaks = seq(1, nrow(df.plot), 1), labels = seq(1, nrow(df.plot)
  scale_y_continuous(expand = c(0, 0), limits = c(0, 1.10 * max(df.plot$dens))) +
  theme(panel.grid.minor.y = element_blank(),
        axis.title.x = element_text(size = 10), axis.title.y = element_text(size = 14),
        axis.text.x  = element_text(size = 8, colour = 'black'), axis.text.y  = element_text(size = 8, 
g1
```



```
##### CREATE MASTER DATASET OF ORIGINAL TWEETS appended with array of emojis
## EMOJIS: create reduced tweets+emojis matrix
df.s <- data.frame(matrix(NA, nrow = nrow(tweets), ncol = 2));
names(df.s) <- c('tweetid', 'emoji.ids');
df.s$tweetid <- 1:nrow(tweets);
system.time(df.s$emoji.ids <- apply(tweets.emojis.matrix[, c(2:843)], 1, function(x) paste(which(x > -1
system.time(df.s$num.emojis <- sapply(df.s$emoji.ids, function(x) length(unlist(strsplit(x, ', ')))));
df.s.emojis <- subset(df.s, num.emojis > 0);
df.s.nonemojis <- subset(df.s, num.emojis == 0);
df.s.nonemojis$emoji.names <- '';

# convert to long, only for nonzero entries
df.l <- cSplit(df.s.emojis, splitCols = 'emoji.ids', sep = ', ', direction = 'long')
map <- subset(emojis, select = c(emojiid, name));
map$emojiid <- as.numeric(map$emojiid);
df.m <- merge(df.l, map, by.x = 'emoji.ids', by.y = 'emojiid');
df.m <- arrange(df.m, tweetid);
df.m <- rename(df.m, c(name = 'emoji.name'))
tweets.emojis.long <- subset(df.m, select = c(tweetid, emoji.name));
df.n <- aggregate(emoji.name ~ tweetid, paste, collapse = ', ', data = df.m);

## merge back with original tweets dataset
df.f <- merge(df.s.emojis, df.n, by = 'tweetid');
df.f <- rename(df.f, c(emoji.name = 'emoji.names'));
df.g <- rbind(df.f, df.s.nonemojis);
df.g <- arrange(df.g, tweetid);
df.h <- merge(tweets, df.g, by = 'tweetid', all.x = TRUE);
df.h$emoji.ids <- NULL;
df.h$tweetid <- as.numeric(df.h$tweetid);
df.h <- arrange(df.h, tweetid);
```

```r
tweets.emojis <- df.h;

#### MAKE TWO WAY PLOT FOR A SET OF MUTUALLY EXCLUSIVE SUBSETS OF THE DATA
df.1 <- subset(tweets.emojis, grepl(paste(c('latte'), collapse = '|'), tolower(tweets.emojis$text)));
df.2 <- subset(tweets.emojis, grepl(paste(c('coffee'), collapse = '|'), tolower(tweets.emojis$text)));
nrow(df.1);
nrow(df.2);
# dataset 1
df.a <- subset(subset(df.1, emoji.names != ''), select = c(tweetid, emoji.names));
df.a$emoji.names <- as.character(df.a$emoji.names);
df.b <- data.frame(table(unlist(strsplit(df.a$emoji.names, ','))));
names(df.b) <- c('var', 'freq');
df.b$var <- trimws(df.b$var, 'both');
df.b <- subset(df.b, var != '');
df.c <- aggregate(freq ~ var, data = df.b, function(x) sum(x));
df.c <- df.c[with(df.c, order(-freq)), ];
row.names(df.c) <- NULL;
df.d <- subset(df.c, freq > 0);
df.d$dens <- round(1000 * (df.d$freq / nrow(df)), 1);
df.d$dens.sm <- (df.d$freq + 1) / (nrow(df) + 1);
df.d$rank <- as.numeric(row.names(df.d)); df.d <- rename(df.d, c(var = 'name'));
df.e <- subset(df.d, select = c(name, dens, dens.sm, freq, rank));
df.e$ht <- as.character(arrange(data.frame(table(tolower(unlist(str_extract_all(df.1$text, '#\\w+')))))
df.e[1:10, ]; emojis.count.1 <- df.e;
# dataset 2
df.a <- subset(subset(df.2, emoji.names != ''), select = c(tweetid, emoji.names)); df.a$emoji.names <- a
df.b <- data.frame(table(unlist(strsplit(df.a$emoji.names, ','))));
names(df.b) <- c('var', 'freq');
df.b$var <- trimws(df.b$var, 'both');
df.b <- subset(df.b, var != '');
df.c <- aggregate(freq ~ var, data = df.b, function(x) sum(x));
df.c <- df.c[with(df.c, order(-freq)), ]; row.names(df.c) <- NULL;
df.d <- subset(df.c, freq > 1);
df.d$dens <- round(1000 * (df.d$freq / nrow(df)), 1);
df.d$dens.sm <- (df.d$freq + 1) / (nrow(df) + 1);
df.d$rank <- as.numeric(row.names(df.d));
df.d <- rename(df.d, c(var = 'name'));
df.e <- subset(df.d, select = c(name, dens, dens.sm, freq, rank));
df.e$ht <- as.character(arrange(data.frame(table(tolower(unlist(str_extract_all(df.2$text, '#\\w+')))))
df.e[1:10, ];
emojis.count.2 <- df.e;
# combine datasets and create final dataset
names(emojis.count.1)[-1] <- paste0(names(emojis.count.1)[-1], '.1');
names(emojis.count.2)[-1] <- paste0(names(emojis.count.2)[-1], '.2');
df.a <- merge(emojis.count.1, emojis.count.2, by = 'name', all.x = TRUE, all.y = TRUE);
df.a[, c(2:4, 6:8)][is.na(df.a[, c(2:4, 6:8)])] <- 0;
df.a <- df.a[with (df.a, order(-dens.1)), ];
df.a$index <- ifelse(df.a$dens.1 > 0 & df.a$dens.2 > 0 & (df.a$dens.1 > df.a$dens.2), round(100 * ((df.a
                     ifelse(df.a$dens.1 > 0 & df.a$dens.2 > 0 & (df.a$dens.2 > df.a$dens.1), -1 * round
df.a$logor <- log(df.a$dens.sm.1 / df.a$dens.sm.2);
df.a$dens.mean <- 0.5 * (df.a$dens.1 + df.a$dens.2);
k <- 50;
df.b <- subset(df.a, (rank.1 <= k | rank.2 <= k) &
                              (freq.1 >= 1 | freq.2 >= 1) &
                              (freq.1 > 0 & freq.2 > 0) & dens.mean > 0); nrow(df.b);
df.c <- subset(df.b, select = c(name, dens.1, dens.2, freq.1, freq.2, dens.mean, round(logor, 2)));
df.c <- df.c[with(df.c, order(-logor)), ]; row.names(df.c) <- NULL; nrow(df.c); df.c;
emojis.comp.p <- df.c;
rbind(head(emojis.comp.p), tail(emojis.comp.p))
```



First of all, the emoji bar chart shows the top ten most frequently used emojis when people talk about Starbucks and the corresponding frequency. The top five emojis are "face with tears of joy", "hot beverage", "heavy black heart", "loudlycrying face" and "smiling face with heart-shaped eyes". After that I choose "coffee" and "lattee" as two keywords (or topics) to get two subsets and explore the emojis used within each subset. The comparison result is interesing and seem intuitive. Hot beverage, joy face and red heart emojis over index on #coffee. The yummy face appeara around y=0 means the odds ratio close to 0 and which indicate this emoji is used equally frequently with both hashtags.

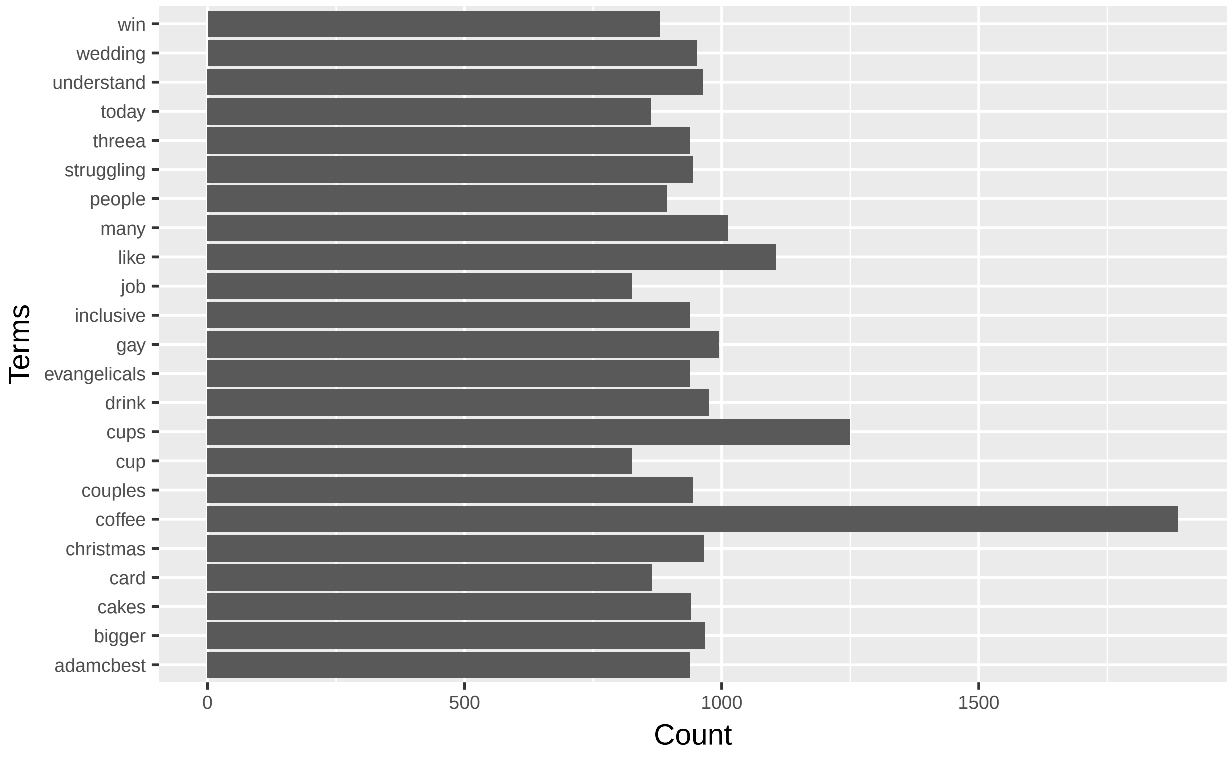
4. Text Mining

a. Word Frequency

After doing the emojis analysis, I also want to explore the most frequently used words when peolple talk about Starbucks. Here I use the textmining technique to deal with the text variable in the dataset. The main package I use here is the tm package. To show the result in a clear way, I draw a bar chart of the words that appear more than 800 times and then draw a wordcloud. The mainly package I use to draw the wordcloud is the wordcloud package.

```
#build a corpus, and specify the source to be character vectors
myCorpus<-Corpus(VectorSource(tweets$text))
#only keep English
removeNumPunct<-function(x) gsub("[^[:alpha:][:space:space]]*","",x)
```

```r
myCorpus<-tm_map(myCorpus,content_transformer(removeNumPunct))
#convert to lower case
myCorpus<-tm_map(myCorpus,content_transformer(tolower))
#remove url's
removeURL<-function(x) gsub("http[^[:space:]]*","",x)
myCorpus<-tm_map(myCorpus,content_transformer(removeURL))
#remove stopwords
myStopwords<-c(stopwords('english'),"use","see","used","via","amp","im","will","ear","rt","starbucks","
myCorpus<-tm_map(myCorpus,removeWords,myStopwords)
#remove blank
myCorpus<-tm_map(myCorpus,stripWhitespace)
#remove punctuation
myCorpus<-tm_map(myCorpus,removePunctuation)

#build term document marix
tdm<-TermDocumentMatrix(myCorpus,control = list(wordLengths=c(1,Inf)))
term.freq<-rowSums(as.matrix(tdm))
term.freq2<-subset(term.freq,term.freq>=10)
term.df<-data.frame(term=names(term.freq2),freq=term.freq2)
term.df1<-term.df[which(term.df$freq>800),]
bad_word1 <- c('NA')
term.df <- term.df[!(term.df[,'term'] %in% bad_word1),]
par(mfrow=c(1,1))
ggplot(term.df1,aes(x=term,y=freq))+geom_bar(stat = "identity")+
  xlab("Terms")+ylab("Count")+coord_flip()+
  theme(axis.text = element_text(size = 7))
```

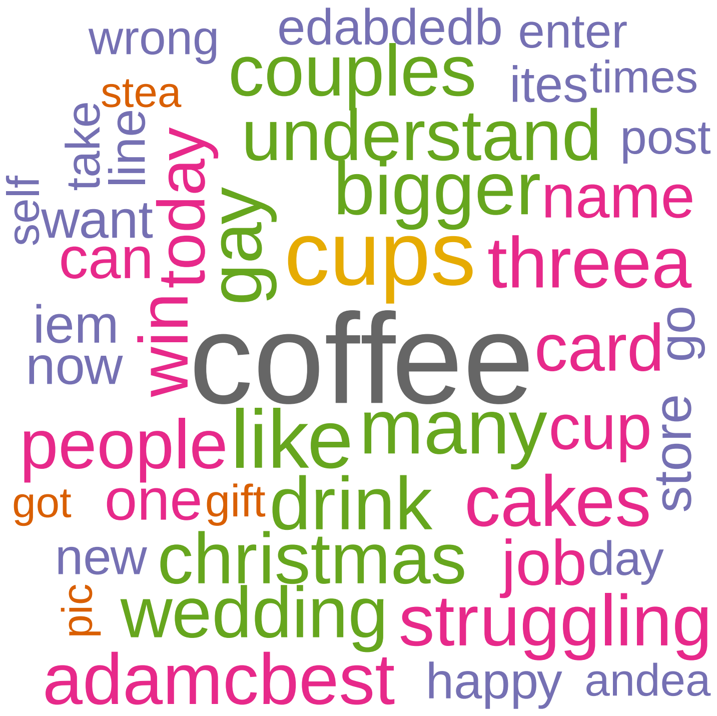


```r
kable(term.df %>%
        filter(freq>950),caption="Top 10 used words")
```

Table 1: Top 10 used words

term	freq
bigger	968
cups	1249
gay	995
many	1012
understand	963
wedding	952
like	1105
drink	976
coffee	1888
christmas	966

```r
m<-as.matrix(tdm)
#calculate the frequency of words ans sort is by frequency
word.freq<-sort(rowSums(m),decreasing = T)
#set up colors
pal<-brewer.pal(8,"Dark2")
#plot word cloud
wordcloud(words = names(word.freq),freq = word.freq,min.freq = 10,
          random.order = F,colors = pal,max.words = 60)
```



The table above shows the words that appear 950 times when people talk about Starbucks in Twitter. According to the bar chart and wordcloud, we can see that words including coffee, cup,cakes have higher frequency than other words. This result indicates that people also talk about the things that are relevant to coffee when they talk about Starbucks. Other words like struggling, like and win may indicate their emotions when they talk about Starbucks.

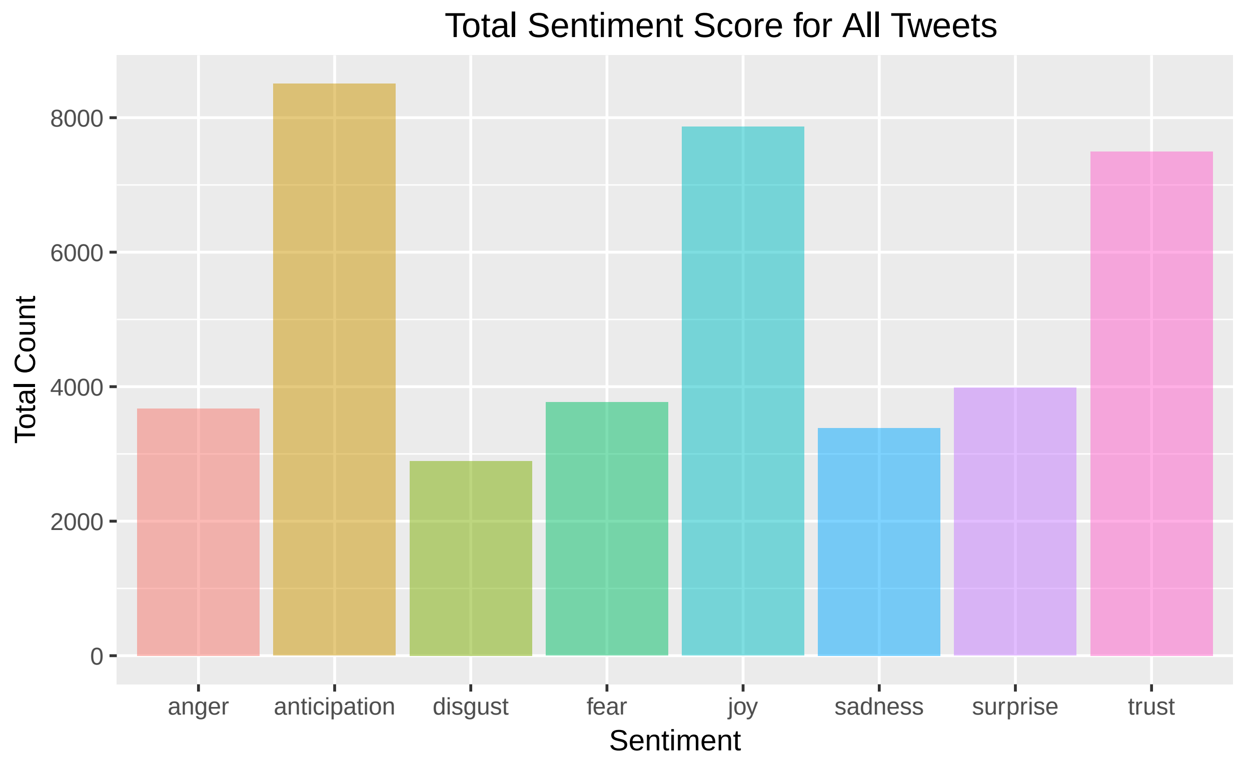
b. Sentiment Analysis

Sentiment analysis is another important part in text mining. Inspird by the results from word frequency part that some words related to service quality also frequently used in tweets, and many of these words can be considered positive, we further perform sentiment analysis on the text variable. In this part, we will focus on attitude of those twitter users' toward Starbucks.

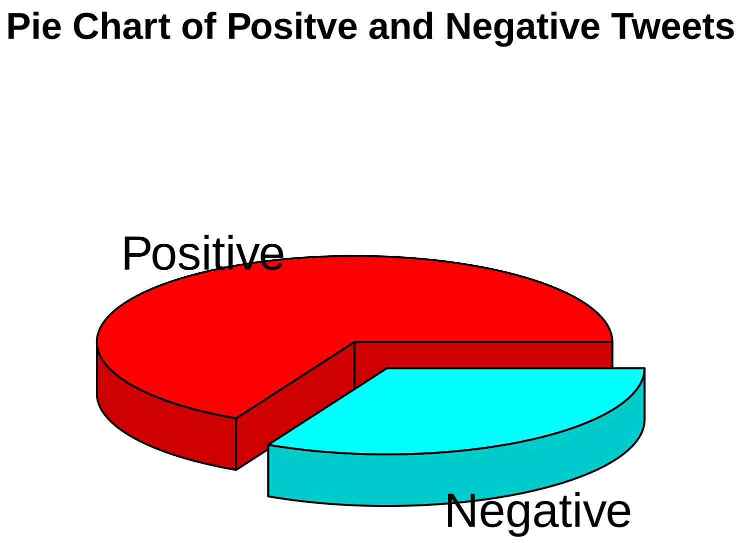
In this part, we conduct sentiment analysis in R using syuzhet package. To get the sentiment of each tweet, we use the get_nrc_sentiment function which implements Saif Mohammand's NRC Emotion lexion. Acoording to Mohammad, "the NRC emotion lexicon" is a list of words and their associations with eight emotions (anger, fear,anticipation,trust,surprise,sadness,joy,and disgust) and two sentiment which are negative and positive".

```r
#extract text
star<-read.csv("tweets.cleaned3_starbucks.csv")
star.text<-star$text

#clean text, get rid of emoji and other irrelavant characters
star.text<-sapply(star.text,function(row) iconv(row,"latin1","ASCII",sub = ""))
#remove retweet entities
star.text<-gsub("(RT|via)((?:\\b\\W*@\\w+)+)","",star.text)
#remove at someone
star.text<-gsub("@\\w+","",star.text)
#remove punctuation
star.text<-gsub("[[:punct:]]","",star.text)
#remove numbers
star.text<-gsub("[[:digit:]]","",star.text)
#remove html link
star.text<-gsub("http\\w+","",star.text)
#remove unnecessary spaces
star.text<-gsub("[ \t]{2,}","",star.text)
star.text<-gsub("^\\s+|\\s+$","",star.text)

#define "tolower error handling" function
try.error<-function(x){
  #create missing values
  y=NA
  #tryCatch error
  try_error=tryCatch(tolower(x),error=function(e) e)
  #if it is not an error
  if(!inherits(try_error,"error"))
    y=tolower(x)
  #result
  return(y)
}
#lower case using try.error with sapply
star.text<-sapply(star.text,try.error)
#extract sentiment
mySentiment<-get_nrc_sentiment(star.text)
#plot sentiment
sentimentTotals<-data.frame(colSums(mySentiment[,c(1:8)]))
names(sentimentTotals)<-"count"
sentimentTotals<-cbind("sentiment"=rownames(sentimentTotals),sentimentTotals)
rownames(sentimentTotals)<-NULL
```

```r
ggplot(data = sentimentTotals,aes(x=sentiment,y=count))+
  geom_bar(aes(fill=sentiment),stat = "identity",alpha=0.5)+
  theme(legend.position = "none")+
  xlab("Sentiment")+ylab("Total Count")+
  ggtitle("                                  Total Sentiment Score for All Tweets")
```



```r
#pie chart with percentages
pos<-sum(mySentiment$positive)
neg<-sum(mySentiment$negative)
slices<-c(pos,neg)
lbls<-c("Positive","Negative")
pie3D(slices,labels = lbls,explode = 0.12,main="Pie Chart of Positve and Negative Tweets")
```



```r
#combine cleaned data with sentiment data
star.sentiment<-cbind(star,mySentiment[,9:10])
```

```
## Warning in data.row.names(row.names, rowsi, i): some row.names duplicated:
## 13,30,33,38,56,75,76,79,80,119,121,123,134,145,149,154,163,164,166,167,173,178,180,184,186,190,191,1
```

"Total Sentiment Score of all Tweets" shows the difference in number of eight emotions based on all the tweets twxt in the US. Trust, joy and anticipation are three highest frequency emotions. Especially the bar of anticipation is significantly higher than other bars. We also notice that all of the top three emotions are positive emotions. The height of other five emotions which are anger, disgust, sadness,fear and surprise are close to each other but surprise is a little bit higher than other four negative emotions. Therefore, we can conclude that there are more Twitter users that hold a postive attitude towards Starbucks.

The "Pie Chart of Positive and Negative Tweets" summarizes the proportion of positive and negative tweets. This pie chart clearly indicates that positive tweets take up more than three quatiles of the total tweets. Therefore, we can get the same conclusion as from the bar chart.

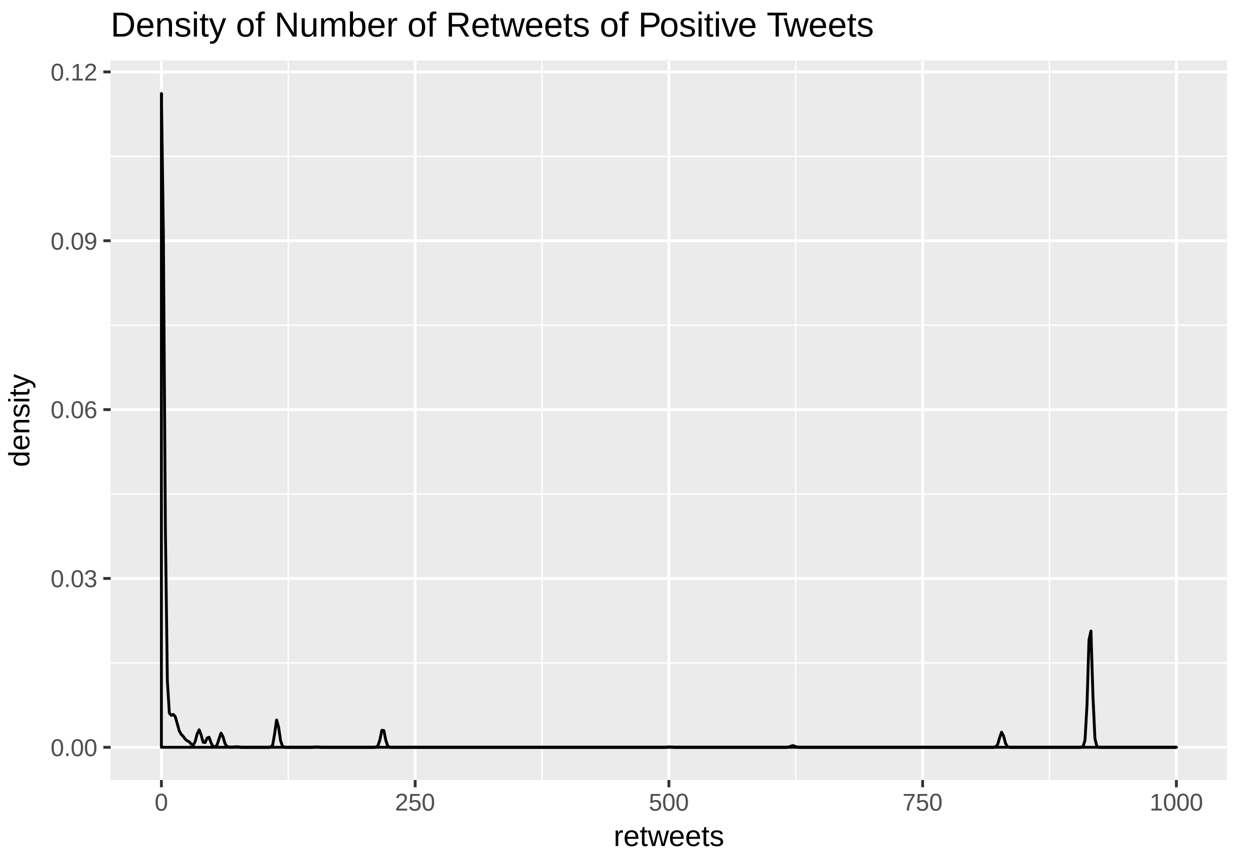
5. Bootstrap

After analyzing the number of tweets of each sentiment group and their proportion, I also want to explore the influence of one's sentiment on Twitter. For example, if Taylor Swift recommends a new Christmas frappuccino on Twitter, it is very likely that the sales of this frappuccino will increase rapidly. Also, if she says really hates the taste of this frappucino then the sales will be likely decreased.

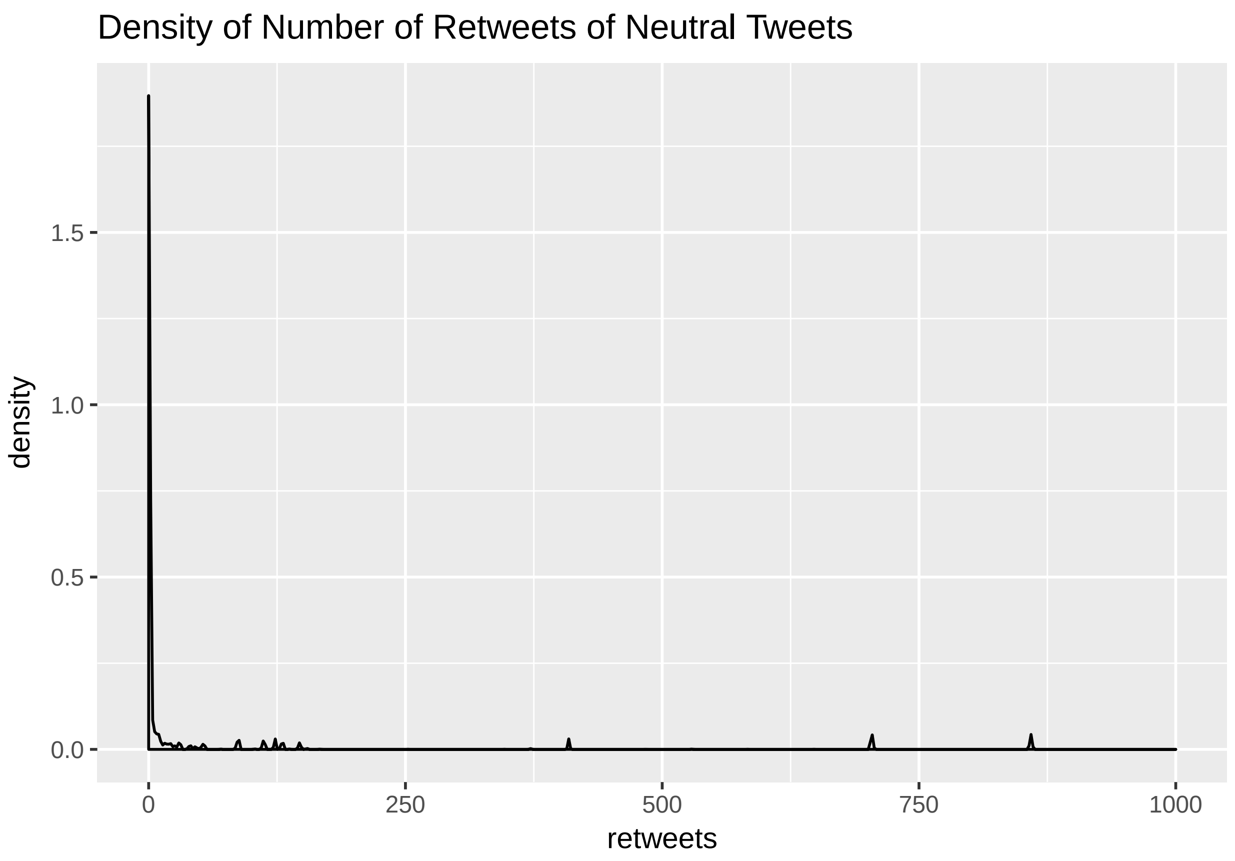
One's emotion is pretty contagious in social media and the effect of one's emotion on others is much larger than we imagine. Indexes like number of retweets, followers and favourites may indicates one's influence on other users. In this part, I use the number of retweets to measure a user's influence on others. I fisrt plot the density function of each group with Gaussian Kernel and then estimate the mean and 95% confidence interval by bootstrap method.

```r
#classification function in which 1:positive,0:neutral,-1:negative
sentiment.class<-function(neg,pos){
  if(neg<pos)
    return("Positive")
  else if(neg>pos)
    return("Negative")
  else
    return("Neutral")
}
sentclass<-mapply(sentiment.class,star.sentiment$positive,star.sentiment$negative)
starclass<-cbind(star,sentclass)

index.pos<-which(starclass$sentclass=="Positive")
index.neu<-which(starclass$sentclass=="Neutral")
index.neg<-which(starclass$sentclass=="Negative")
pos.retweet<-starclass[index.pos,]$retweets
neg.retweet<-starclass[index.neg,]$retweets
neu.retweet<-starclass[index.neu,]$retweets
pos<-starclass[index.pos,]
neg<-starclass[index.neg,]
neu<-starclass[index.neu,]
attach(pos)
attach(neu)
attach(neg)
```

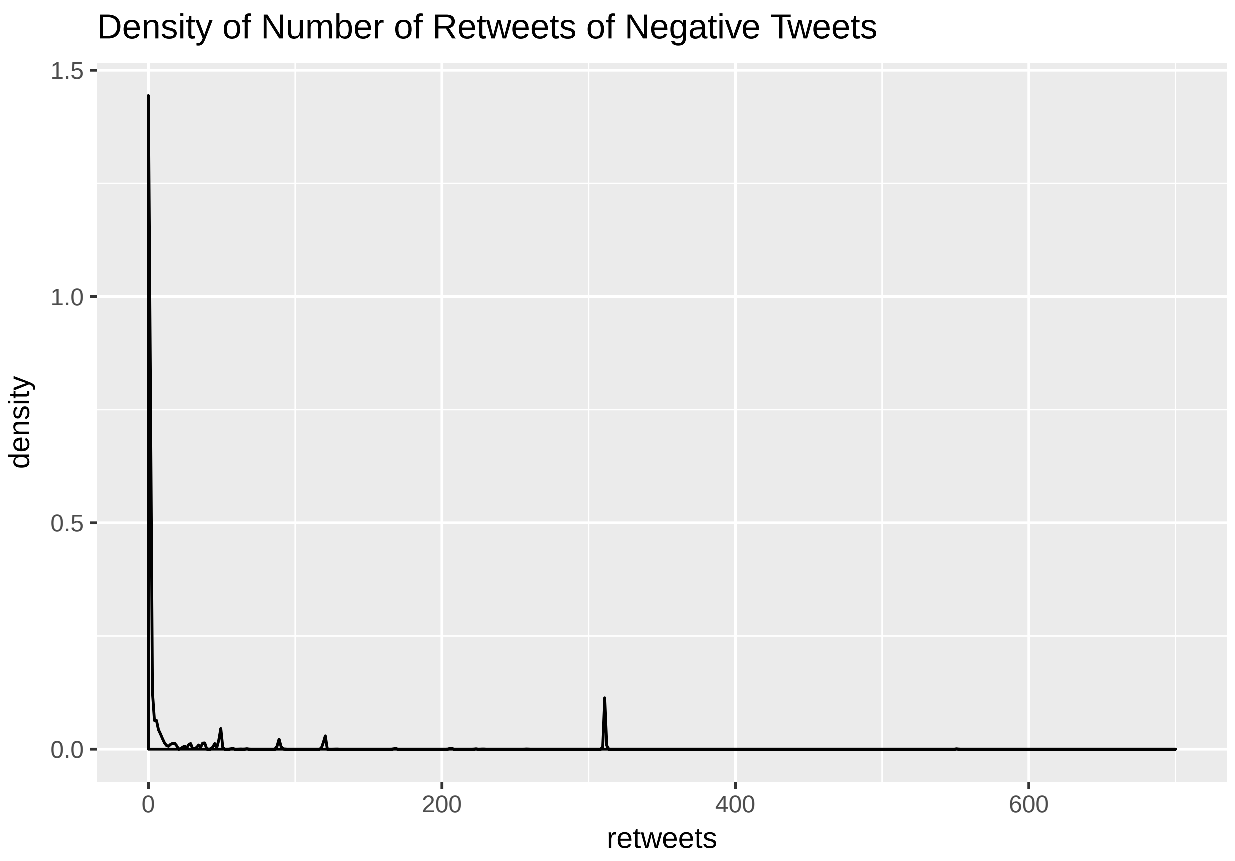
```r
#plot density of retweets
ggplot(data = pos,aes(retweets))+geom_density(kernel="gaussian")+
  labs(title="Density of Number of Retweets of Positive Tweets ")+
   xlim(0,1000)
```



```r
ggplot(data = neu,aes(retweets))+geom_density(kernel="gaussian")+
  labs(title="Density of Number of Retweets of Neutral Tweets ")+
   xlim(0,1000)
```



```
ggplot(data = neg,aes(retweets))+geom_density(kernel="gaussian")+
  labs(title="Density of Number of Retweets of Negative Tweets ")+
   xlim(0,700)
```



```r
library(boot)
#use bootstap to find mean of retweets for negative, positive and neutral tweets
boot.mean<-function(data,index){
  x<-data[index]
  return(mean(x))
}
#bootstrap for positive, negative and neutral tweets
#positive
boot.pos<-boot(pos.retweet,boot.mean,R=5000)
bootci.pos<-boot.ci(boot.pos,conf = 0.95,type = "all")
boot.pos
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = pos.retweet, statistic = boot.mean, R = 5000)
##
##
## Bootstrap Statistics :
##     original     bias    std. error
## t1* 158.8634 0.1123414    6.918483
```

```r
bootci.pos
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 5000 bootstrap replicates
```

```
##
## CALL :
## boot.ci(boot.out = boot.pos, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   (145.2, 172.3 )   (144.4, 171.6 )
##
## Level     Percentile            BCa
## 95%   (146.2, 173.3 )   (146.7, 174.1 )
## Calculations and Intervals on Original Scale
```

```
#negative
boot.neg<-boot(neg.retweet,boot.mean,R=10000)
bootci.neg<-boot.ci(boot.neg,conf = 0.95,type = "all")
boot.neg
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = neg.retweet, statistic = boot.mean, R = 10000)
##
##
## Bootstrap Statistics :
##     original     bias    std. error
## t1* 349.3531 0.05604726    10.76173
```

```
bootci.neg
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.neg, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   (328.2, 370.4 )   (326.9, 369.3 )
##
## Level     Percentile            BCa
## 95%   (329.4, 371.8 )   (331.1, 374.4 )
## Calculations and Intervals on Original Scale
```

```
#netural
#boot.neu<-boot(neu.retweet,boot.mean,R=20000)
#bootci.neu<-boot.ci(boot.neu,conf = 0.95,type = "all")
#boot.neu
#bootci.neu
```

According to the output above, we can see that the estimated mean for the number of retweets of the negative group is smaller than the positive group. This result indicates that although the number of retweets in negative group is much smaller than that of the neutral and positive group, per nergative tweet influence much on other twitter users than a positive tweet. Therefore, it might have an larger effect on the overall attitude toward Starbucks.

6. Clustering Analysis

In this part, I use two distance-based clustering methods which are K-Means clustering and Hierarchical clustering. For the K-Means clustering, clustering splits the total sum of squares between the sum of squares within clusters and the sum of squares between clusters. And the sum of squares within clusters decreases as the number of clusters increases. For the hierarchical clustering, it begins by building a full tree that begins with each observation, called leavees, as sigleton clusters and agglomerates the leaves into clusters based on distance calculations recorded in a single matrix.

a. K-Means Clustering

The basic idea behind k-means clustering consists of defining clusters so that the total within-cluster variation is minimized. The standard algorithm defines the total within-cluster variation as the sum of squared distances Euclidean distances between items and the corresponding centroid:

$$W(C_k) = \sum_{x_i \in C_k} (x_i - \mu_k)^2.$$

where x_k is a data point belonging to the cluster C_k and μ_k is the mean value of the points assigned cluster centers (μ_k) is minimized. Furthermore, we define the total within-cluster variation as follows:

$$tot.withiness = \sum_{k=1}^{k} W(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2.$$

The total within-cluster sum of square measures the compactness (i.e goodness) of the clustering and we want it to be as small as possible

```r
#remove sparse terms
tdm2<-removeSparseTerms(tdm,sparse = 0.98)
#showing the words that are left
m2<-as.matrix(tdm2)
m3<-t(m2)
m4 <- m3[,-c(6,24)]
#colnames(m3)
set.seed(2017)
k<-5
kmeansResult<-kmeans(m4,k)
#round(kmeansResult$centers,digits = 3)

for(i in 1:k){
  cat(paste("cluster",i,": ",sep = ""))
  s<-sort(kmeansResult$centers[i,],decreasing = T)
  cat(names(s)[1:5],"\n")
}
```

```
## cluster1: now coffee can christmas new
## cluster2: coffee cups like many gay
## cluster3: coffee day go drink cup
## cluster4: want work job go now
## cluster5: win card enter starbuckscanada gift
```

In this part, I determine to classify the words into five groups and the mainly function used here is the kmeans function in R. From the clustering result we can see that the words in each group are somewhat similar to each other

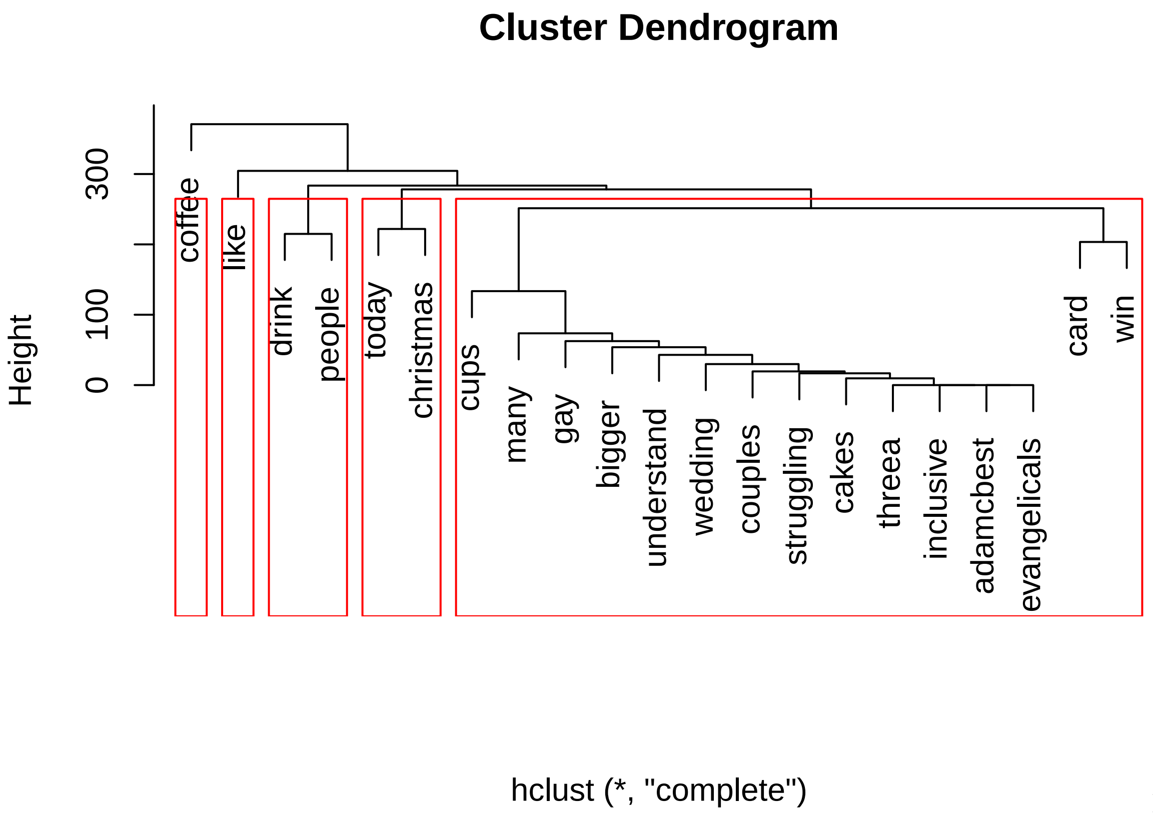
b. Hierarchical Clustering

In this part, I use hclust function and adopt the complete method to do the hierarchical clustering. In hierarchical clustering, the linkage criterion determines the distance between sets of observations as a function of the pairwise distances between observations. Here the linkage criteria of complete-linkage clustering is

$$max\{d(a,b) : a \in A, b \in B\}.$$

where d is the chosen matric.

```r
#remove sparse terms
tdm2.1<-removeSparseTerms(tdm,sparse = 0.96)
#showing the words that are left
m2.1<-as.matrix(tdm2.1)
#cluster terms
distMatrix<-dist(scale(m2.1))
fit<-hclust(distMatrix,method = "complete")
#show cluster dendrogram
p<-plot(fit,xlab="")
p<-rect.hclust(fit,k=5)
```



In the hierarchical clustering, I also divide those words into five froups. The result is different compared to the K-Means clustering but it still gives us a reasonable classification.

III. Conclusion

In this project, I select Starbucks as the keyword (or topic) and analyze the Twitter activity of Starbucks. First of all, I draw two maps that one of which shows the distribution of the number of tweets in the United States and the other is a geolocation density plot. In the emojis analysis, I draw a bar chart that sumarises the most frequently used words that appear more than 800 times and then a plot to show the rank. Considering one's sentiment is important to a brand and pretty contagious in social media, I perform a sentiment analysis and then use bootstrap to get the influence of one's tweets on others. The results show that most of the users hold positive attitude toward Starbucks. Howerver, the negative attitude has the greatest effect on Twitter. In the last part, I use two kinds of distance-based clustering methods to classify the words into five group. The clustering results are not exactly the same but both of them are reasonable in real life.