Analysis of Twitter Activity and Sentiment on Starbucks

Zihuan Qiao Teammate: CJ Xiang

I. Introduaction

In this gerneration of social media, almost everyone has a social media account to interact with each other almost everyday. People express their ideas freely on any topics, from politics to what they eat today. There are also a series of functions enabling different kinds of interaction among users, like replying, forwarding, that enlarges the impact of social media on users. For its huge amount of users and highly interactive property, social media data plays an increasingly significant role in business analysis and financial analysis nowadays.

Twitter, one of the most popular social medias now, has its API available to the public which makes data analysis taking advatage of social media data easier than ever. By using twitter API, one can connect to twitter database to get data of a specific location during a specific time on a specific topic.

In this work, we try to use Starbucks twitter data to see the twitter activity and sentiment on Starbucks. Starbucks is an American coffee commpany and coffeehouse chain that has very high popularity. There are over 13,107 Starbucks in the United States. It is meaningful to see whether this popular and important coffeehouse chain company has good reputation on the social media. This can be helpful for many business purposes.

This project will be arranged as following: we will first look at the number of twitter on Starbucks compared with Dunkin' Donuts and their distribution in the US on a map to see which one is popular as a topic on social media. Then, we want to show what are people talking about Starbucks when mentioning it on Twitter by using word clouds. Further analysis is related to sentiment analysis. We use ordinal logistic regression to see the relationship between location and sentiment to Starbucks (positive, neutral or negative). Shiny app is used for visualization here. Taking diffrent level of influence of each tweets has on other users, we also conduct analysis on the variables that can reflect the influence level of a tweet. We use kernel density estimation to get plots of density of the variables and use bootstrap to estimate their means.

II. Method

1. Data

In order to get data from twitter, we first set up twitter API by linking URL and KEY to server.

In addition to Starbucks, we also collect Dunkin' Donuts twitter data for comparison. Dunkin' Donuts is another American coffeehouse chain. The company has grown to become one of the largest coffee and baked goods chains in the world. In order to compare their popularity on Twitter, we set all the other parameters to be the same except for the searching topic.

```
## Collect data from Twitter

### Corresponding roauth R code and stream R code can be found
###in the seperate R code files: roauth.R, stream_Starbucks.R, stream_Dunkin' Donuts.R
### Here starts from reading the saved RDS data

dunkinUS.df <- readRDS("Dunkin' Donuts US Data.RDS")
starbucksUS.df <- readRDS("Starbucks US Data.RDS")
dim(dunkinUS.df)
```

```
## [1]  6 42
```

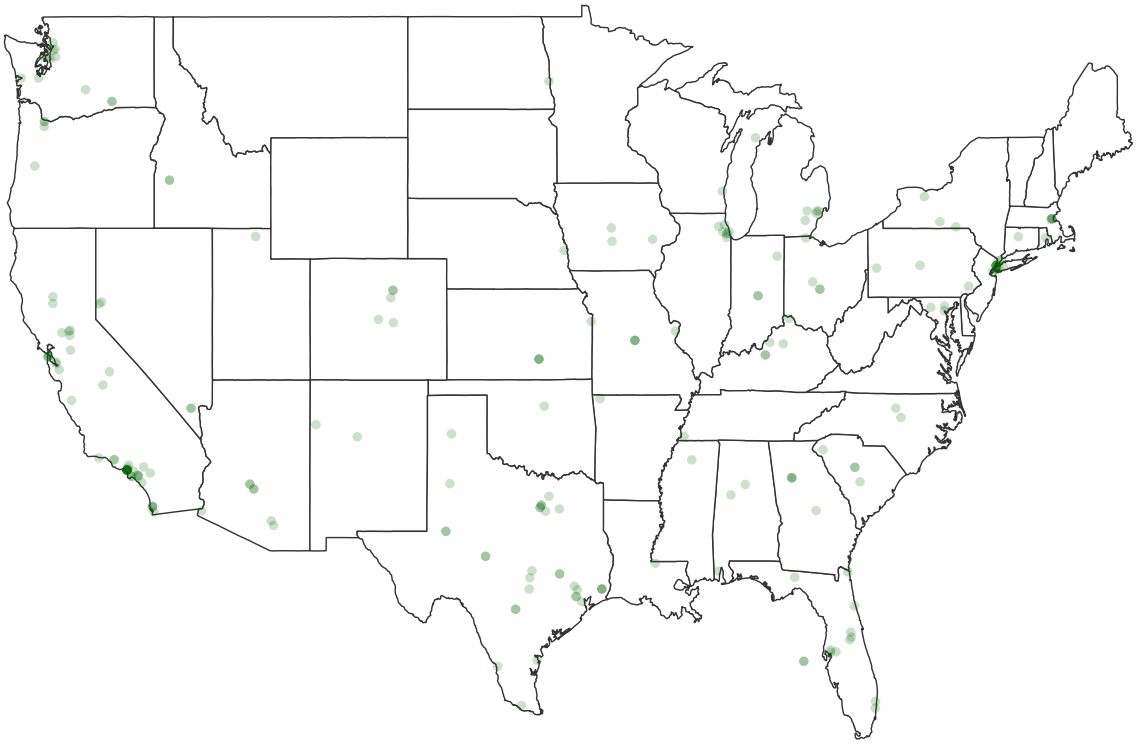
```
dim(starbucksUS.df)
```

```
## [1] 210  42
```

In the same amount of time, 6000 seconds, we collect 210 tweets about Starbucks in the US and only 6 tweets about Dunkin' Donuts in the US. Each dataset has 42 varibales. Huge amount of differece in number of twitter indicate that Starbucks is much more popular as a topic on twitter.

```
## draw maps of twitter activity distribution

library(ggplot2)

#draw map of Starbucks tweets in the USA domain
map.data <- map_data("state")
points <- data.frame(x=as.numeric(starbucksUS.df$place_lon),
                     y=as.numeric(starbucksUS.df$place_lat))

points <- points[points$y>25,]
ggplot(map.data)+
  geom_map(aes(map_id = region),
           map=map.data,
           fill="white",
           color="grey20",size=0.25)+
  expand_limits(x = map.data$long, y = map.data$lat)+
  theme(axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        panel.background = element_blank(),
        panel.border = element_blank(),
        panel.grid.major = element_blank(),
        plot.background = element_blank(),
        plot.margin = unit(0 * c(-1.5, -1.5, -1.5, -1.5), "lines"))+
  geom_point(data = points,
             aes(x = x, y = y), size = 1,
             alpha = 1/5, color = "darkgreen")+
  ggtitle("Tweets mentioning Starbucks in the U.S.")
```

Tweets mentioning Starbucks in the U.S.



```
#draw map of Dunkin' Donuts tweets in the USA domain
map.data <- map_data("state")
points <- data.frame(x=as.numeric(dunkinUS.df$place_lon),
                     y=as.numeric(dunkinUS.df$place_lat))


points <- points[points$y>25,]
ggplot(map.data)+
  geom_map(aes(map_id=region),
           map=map.data,
           fill="white",
           color="grey20",size=0.25)+
  expand_limits(x=map.data$long,y=map.data$lat)+
  theme(axis.line=element_blank(),
        axis.text=element_blank(),
        axis.ticks=element_blank(),
        axis.title=element_blank(),
        panel.background=element_blank(),
        panel.border=element_blank(),
        panel.grid.major=element_blank(),
        plot.background=element_blank(),
        plot.margin=unit(0*c(-1.5,-1.5,-1.5,-1.5),"lines"))+
        geom_point(data=points,
        aes(x=x,y=y),size=2,
        alpha=1/5,color="orange")+
  ggtitle("Tweets Mentioning Dunkin' Donuts in USA")
```

Two maps above show the twitter activity distribution in the country. Each point represents a tweet. We can see that tweets on Starbucks distribute almost evenly in ths US, except for the Midwest region. There are barely points in this region which indicates topic Starbucks is not hot there. While in terms of Dunkin' donuts, since there are only six data in the US, so the points on the map are also very sparse. Tweets on Dunkin' Donuts are far less active than Starbucks.

2. Variable

There are altogether 42 variables in the original data from Twitter, incluing text, followers count, favourites count, name, tweet time, country, place lontitude, place latitude and so on. But we are not going to use all of them. In this work, we are only going to focus on six of them, they are text, followers count, favourites count, full name, place latitude and place lontitude.

Table 1: Names and descriptions of Variables

Variable Name	Description and Variable Labels
text	tweets text
followers count	number of followers of user
favourites count	number of favourites of tweet
full name	user location full name, including city name and state abbreviation
place latitude	user location latitude
place lontitude	user location lontitude

3. Text Mining

a. Word Frequency

In order to find what twitter users are talking about when mentioning Starbucks, we use text mining technique to deal with text variable in the data which are tweets contents. In R, the main package to perform text mining is tm package. In addition to that, we also use wordcloud package along with RColorBrewer package to visualize the word frequency. Here we draw two word clouds, the first one corresponds to the Starbucks tweets content, the second one is about the hashtag frequency.

```
## text analysis: wordclouds

# Let's import necessary packages needed for generating a wordclound
library(tm)
```

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

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(RColorBrewer)
library(stringr)


# remove Emoji and weried characters
Star_text <- sapply(starbucksUS.df$text, function(row) iconv(row, "latin1", "ASCII", sub=""))
# create a corpus
Star_corpus = Corpus(VectorSource(Star_text))
# create document term matrix applying some transformations
tdm = TermDocumentMatrix(Star_corpus,
       control = list(removePunctuation = TRUE,
       stopwords = c("Starbucks", stopwords("english")),
       removeNumbers = TRUE, tolower = TRUE))

# define tdm as matrix
m = as.matrix(tdm)
# get word counts in decreasing order
word_freqs = sort(rowSums(m), decreasing=TRUE)
# create a data frame with words and their frequencies
dm = data.frame(word=names(word_freqs), freq=word_freqs)
# plot wordcloud
wordcloud(dm$word, dm$freq, random.order=FALSE, colors=brewer.pal(8, "Dark2"))
```


```
# wordclous on #
set.seed(146)
pal <- brewer.pal(9,"YlGnBu")
pal <- pal[-(1:4)]
hoo <- str_extract_all(Star_text, "#\\w+")
namesCorpus2 <- Corpus(VectorSource(hoo))
wordcloud(words = namesCorpus2, scale=c(3,0.5), max.words=40, random.order=FALSE,
          rot.per=0.10, use.r.layout=FALSE, colors=pal)
```

From the first word cloud, we can see that words including hiring, job, career have very high frequency. This indicates that many twitter user concern about something related to work at Starbucks. Other high frequency word include hospitality, happy, nice, beautiful which are related to quality of service of Starbucks, and all of them seem to be positive evaluation.

From the second word cloud, which is the word cloud of hashtags, it's even clearer that hashtags like jobs, career, hiring still are among the tops in frequency. It again indicates that jobs at Starbucks is a

hot topic in twitter to some extent.

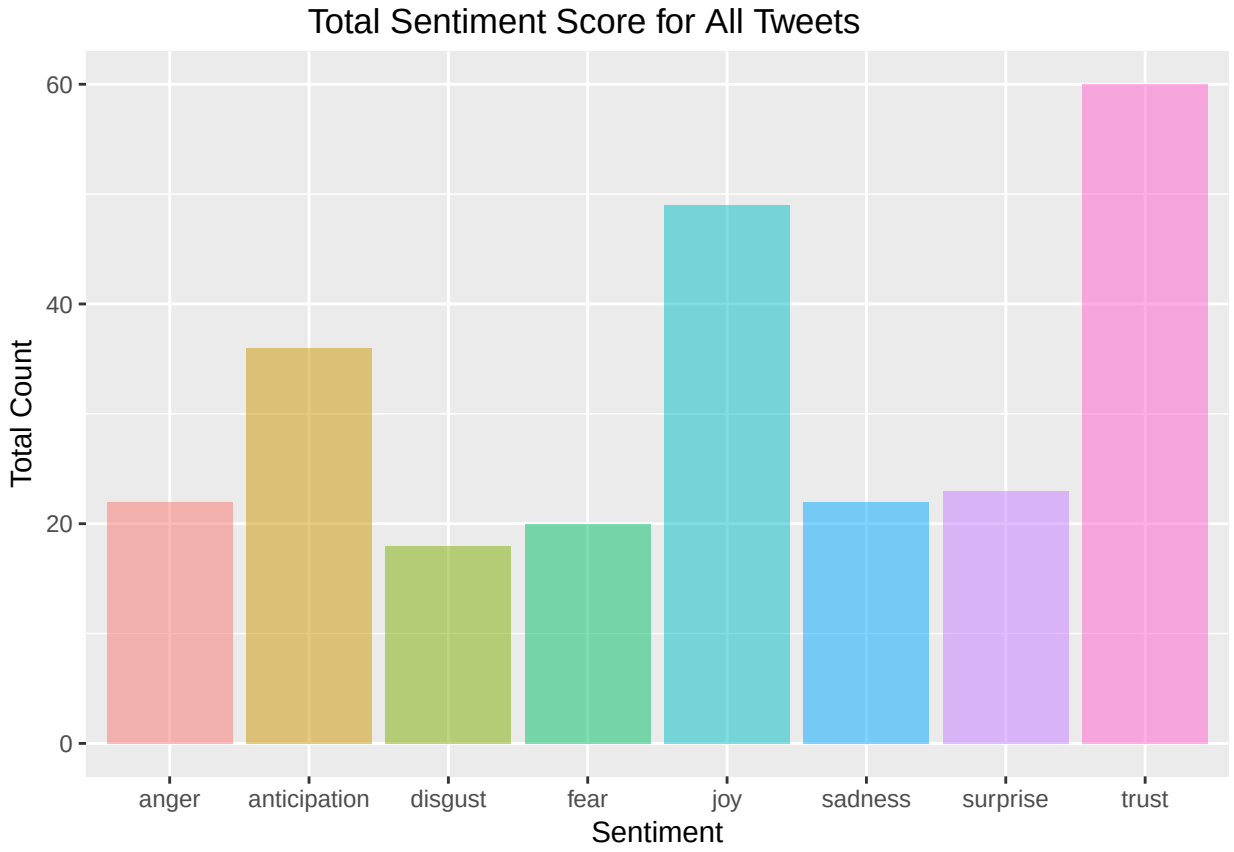
b. Sentiment Analysis

Sentiment analysis is another important part of conducting text mining. Inspired by the results from word frequency part that some words related to service quality are also frequently mentioned in tweets, and that many of these words are positive words, we further perform sentiment analysis on the text variable. In this part, we try to find twitter users' attitude towards Starbucks.

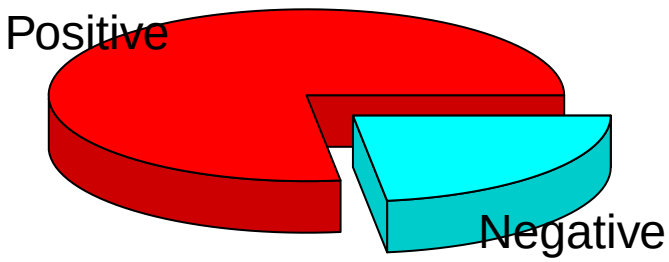
In this project, we perform sentiment analysis in R using syuzhet package. To get the sentiment of each tweet, we apply the get nrc sentiment command. The get nrc sentiment implements Saif Mohammad's NRC Emotion lexicon. According to Mohammad, "the NRC emotion lexicon is a list of words and their associations with eight emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive)".

```
## sentiment analysis

# required pakacges
library(dplyr)
library(ggplot2)
library(syuzhet)
library(plotrix)


# extract text
Star_text <- starbucksUS.df$text

# clean text
# We try to get rid of Emoji and weried characters
Star_text <- sapply(starbucksUS.df$text,
                    function(row) iconv(row, "latin1", "ASCII", sub=""))
# remove retweet entities
Star_text = gsub("(RT|via)((?:\\b\\W*@\\w+)+)", "", Star_text)
# remove at people
Star_text = gsub("@\\w+", "", Star_text)
# remove punctuation
Star_text = gsub("[[:punct:]]", "", Star_text)
# remove numbers
Star_text = gsub("[[:digit:]]", "", Star_text)
# remove html links
Star_text = gsub("http\\w+", "", Star_text)
# remove unnecessary spaces
Star_text = gsub("[ \t]{2,}", "", Star_text)
Star_text = gsub("^\\s+|\\s+$", "", Star_text)

# define "tolower error handling" function
try.error = function(x)
{
  # create missing value
  y = NA
  # tryCatch error
  try_error = tryCatch(tolower(x), error=function(e) e)
  # if 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
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")
```



```
# 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 Postive and Negative Tweets")
```




```
# Combine clean data with sentiment data
starfull <- cbind(starbucksUS.df,mySentiment[,9:10])
```

'Total Sentiment Score for all Tweets' shows difference in number of eight emotions based on all the tweets text in the US. Trust, joy and anticipation are three highest frequency emotions. Especially bars for trust and joy are significantly higher than the other bars. Top three emotions are all positive emotions. Level of other five emotions including anger, disgust, fear, sadness and surprise are close to each other. And four out of these five emotions are negative emotions. We also notice that the difference between sadness and anticipation is only 5, which not very big actually. To conclude, there are more tweets user that hold a positive attitude towards Starbucks.

'Pie Chart of Positive and Negative Tweets' summarize the proportion of positive tweets and negative tweets. This pie chart clearly shows that positive tweets take up more than three quatiles of the total tweets. Thus, we can draw the same conclusion as from the bar chart.

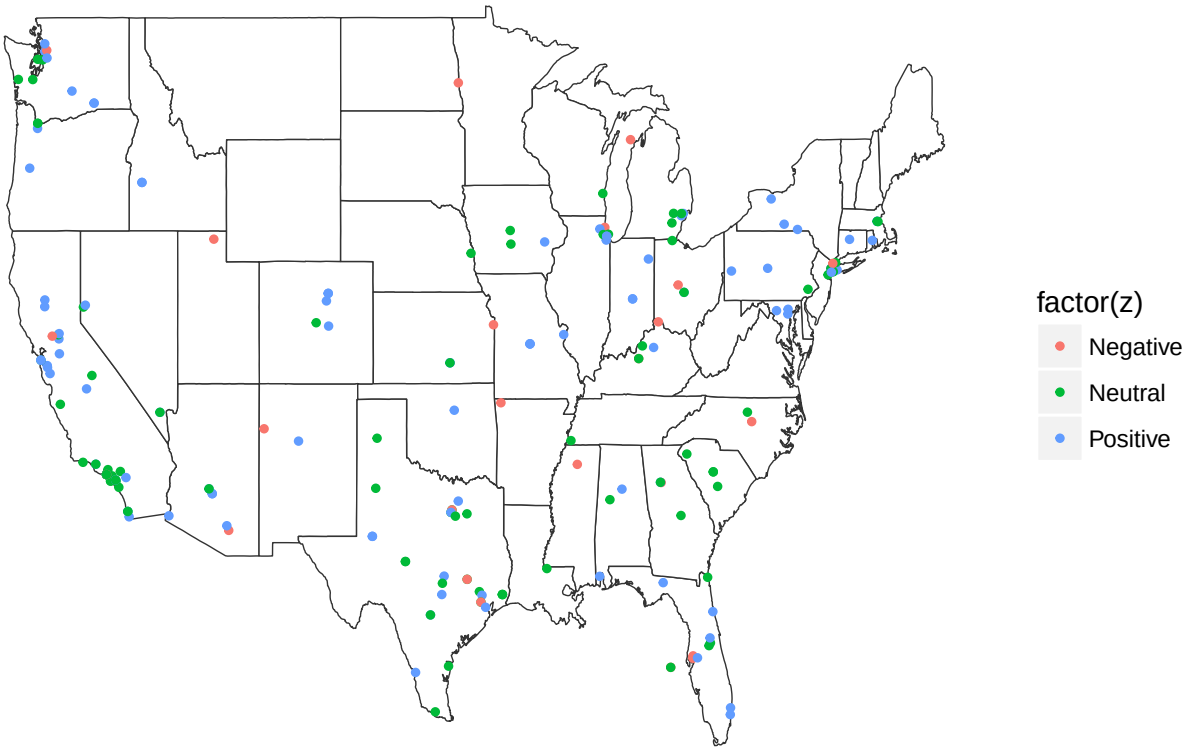
4. Ordinal Logistic Regression

Twitter users' attitude towards Starbucks may also vary between locations. Study on the sentiment diffrence between locations is of great importance because it can provide useful information for many business decisions, like target marketing based on location, sales analysis and so on.

```
# Relationship between Sentiment and Location: Ordinal Logistic Regression

# Visualize sentiment distribution
library(ggplot2)

starfull <- readRDS("StarbucksFullData.rds")
```

```
# sentiment classification
# sentiment classification function: '1' positive, '0' neutral, '-1' negative
sent.class <- function(neg,pos){
  if(neg<pos)
    return("Positive")
  else if (neg==pos)
    return("Neutral")
  else
    return("Negative")
}
sentclass <- mapply(sent.class,starfull$negative,starfull$positive)
starclass <- cbind(starfull,sentclass)

attach(starclass)
map.data <- map_data("state")
points <- data.frame(x=as.numeric(place_lon), y=as.numeric(place_lat), z=sentclass)

points <- points[points$y>25,]
ggplot(map.data)+
geom_map(aes(map_id = region),
       map=map.data,
       fill="white",
       color="grey20",size=0.25)+
expand_limits(x = map.data$long, y = map.data$lat)+
theme(axis.line = element_blank(),
       axis.text = element_blank(),
       axis.ticks = element_blank(),
       axis.title = element_blank(),
       panel.background = element_blank(),
       panel.border = element_blank(),
       panel.grid.major = element_blank(),
       plot.background = element_blank(),
       plot.margin = unit(0 * c(-1.5, -1.5, -1.5, -1.5), "lines"))+
geom_point(data = points,
           aes(x=x,y=y,colour=factor(z)), size = 1)+
ggtitle("Sentiment Distribution in the U.S.")
```

Sentiment Distribution in the U.S.



'Sentiment Distribution in the U.S.' map shows tweets with different sentiment in different color and how they spread in the country. Green points which stand for neutral tweets and blue points which stand for positive tweets all seem to spread almost evenly in the country except for the midwest region. But red points seem to be sparse in the west while dense in the east. To see the relationship between sentiment and location in a quantitative way, we use the Ordinal Logistic Regression Model.

Ordinal Logistic Regression is a regression for ordinal dependent variables. In our model, the response variable is sentiment which has three levels, positive, neutral and negative. The response variable is ordinal variable. The explanatory variable is regions. Here we divide the states into four regions, west, south, midwest and northeast. So the explanatory data is categorical data.

Table 2: Variables in the Ordinal Logistic Regression

Explanatory Variable	Response Variable
Resgions	Sentiment

Table 3: Regions Division

Region Name	States
West	Arizona, Colorado, Idaho, Montana, Nevada, New Mexico, Utah, Wyomingm, Alaska, California, Hawaii, Oregon, and Washington
South	Delaware, Florida, Georgia, Maryland, North Carolina, South Carolina, Virginia, District of Columbia, and West Virginia, Alabama, Kentucky, Mississippi, Tennessee, Arkansas, Louisiana, Oklahoma, and Texas
Mideast	Illinois, Indiana, Michigan, Ohio, Wisconsin, Iowa, Kansas, Minnesota, Missouri, Nebraska, North Dakota, and South Dakota
Norththeast	Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island, Vermont, New Jersey, New York, and Pennsylvania

In addition to the map above, we also visualize the the number of tweets that fall into different sentiment in every region by Shiny. Shiny is a web application framework that turns analysis into interactive web application.

This link leads to the Shiny app: https://sijiexiang.shinyapps.io/Final_Project/

```
# divide data into 4 Areas(West, Mid_west, South, North East)
# We load starclass data into EXCEl and make sure their full name column
# follow the same formart(City, States)
starclass <- read.csv("CorrectedLocationData.csv")
starclass <- starclass[,-1]

# extract state abbreviation
starclass$State_names <- gsub(".*,","",starclass$full_name)
starclass$State_names <- gsub(".* ","",starclass$State_names)

# Now we categorize 50 states into 4 regions: East, West, South, and Midwest
Divide.State <- function(input){
  Reg1 = c("WA", "MT", "OR", "ID", "WY", "CA", "NV", "UT", "CO", "AZ", "NM") # West
  Reg2 = c("OK", "TX", "AR", "LA", "MS", "AL", "TN", "KY", "WV", "MD", "DE", "DC",
           "VA", "NC", "SC", "GA", "FL") # South
  Reg3 = c("ND", "SD", "NE", "KS", "MO", "IA", "MN", "WI", "IL", "IN", "OH", "MI") # Midwest
  Reg4 = c("NY", "PA", "NJ", "CT", "RI", "MA", "VT", "NH", "ME") # Northeast
  if (input %in% Reg1)
    return("West")
  else if (input %in% Reg2)
    return("South")
  else if (input %in% Reg3)
    return("Midwest")
  else
    return("Northeast")
}

sen <- function(x){
  if(x==3)
    return("Positive")
  else if (x==2)
    return("Neutral")
  else
    return("Negative")
}
starclass$regions <- apply(data.frame(starclass$State_names), 1, Divide.State)
starclass$sentiment <- apply(data.frame(starclass$sentclass), 1, sen)


# Ordinal Logistic Regression
# extract data that will be used in the Ordinal Logistic Resgression Model
logdata <- starclass[,c("sentiment","regions")]
attach(logdata)

# load packages
library(foreign)
library(ggplot2)
library(MASS)
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 3.3.2
```

```
library(reshape2)

# description of data
# categorical data distribution
lapply(logdata[, c("sentiment","regions")], table)
```

```
## $sentiment
##
## Negative  Neutral Positive
##       26       89       95
##
## $regions
##
##   Midwest Northeast     South      West
##        34        27        67        82
```

```
ftable(xtabs(~ regions + sentiment, data = logdata))
```

```
##           sentiment Negative Neutral Positive
## regions
## Midwest                    5      12       17
## Northeast                  3      11       13
## South                     12      31       24
## West                       6      35       41
```

```
# Ordinal Logistic Regression
## fit ordered logit model and store results 'm'
logdata$sentiment <- as.factor(logdata$sentiment)
m <- polr(sentiment ~ regions, data = logdata, Hess=TRUE)
## view a summary of the model
summary(m)
```

```
## Call:
## polr(formula = sentiment ~ regions, data = logdata, Hess = TRUE)
##
## Coefficients:
##                      Value Std. Error  t value
## regionsNortheast  0.008381     0.4986  0.01681
## regionsSouth     -0.520995     0.4086 -1.27513
## regionsWest       0.138920     0.3960  0.35083
##
## Intercepts:
##                  Value   Std. Error t value
## Negative|Neutral -2.0964  0.3770    -5.5602
## Neutral|Positive  0.0883  0.3376     0.2615
##
## Residual Deviance: 407.4219
## AIC: 417.4219
```

```
## coefficient
coefficient <- coef(summary(m))
## calculate and store p values
p <- pnorm(abs(coefficient[, "t value"]), lower.tail = FALSE) * 2
## combined coefficient and p values
(mresult <- cbind(coefficient, "p value" = p))
```

```
##                          Value Std. Error     t value      p value
## regionsNortheast  0.008381112  0.4985735  0.01681018 9.865880e-01
## regionsSouth     -0.520995347  0.4085831 -1.27512695 2.022643e-01
## regionsWest       0.138919993  0.3959757  0.35082961 7.257162e-01
## Negative|Neutral -2.096360755  0.3770324 -5.56016011 2.695273e-08
## Neutral|Positive  0.088280405  0.3376490  0.26145611 7.937408e-01
```

```
## compute confidence interval
(ci <- confint(m))
```

```
##                        2.5 %    97.5 %
## regionsNortheast -0.9709307 0.9914298
## regionsSouth     -1.3317197 0.2753869
## regionsWest      -0.6446340 0.9136386
```

Since the confidence intervals for three regions all include 0, regions is not atatistically significant. Hence, there is not statistically significant relationship between regions and sentiment. In another word, there is not much difference in twitter users' attitude towards Starbucks among different regions in the U.S.. If given location of a specific tweet, we cannot predict if it is more likely to be positive, neutral or negative towards Starbucks.

5. Bootstrap

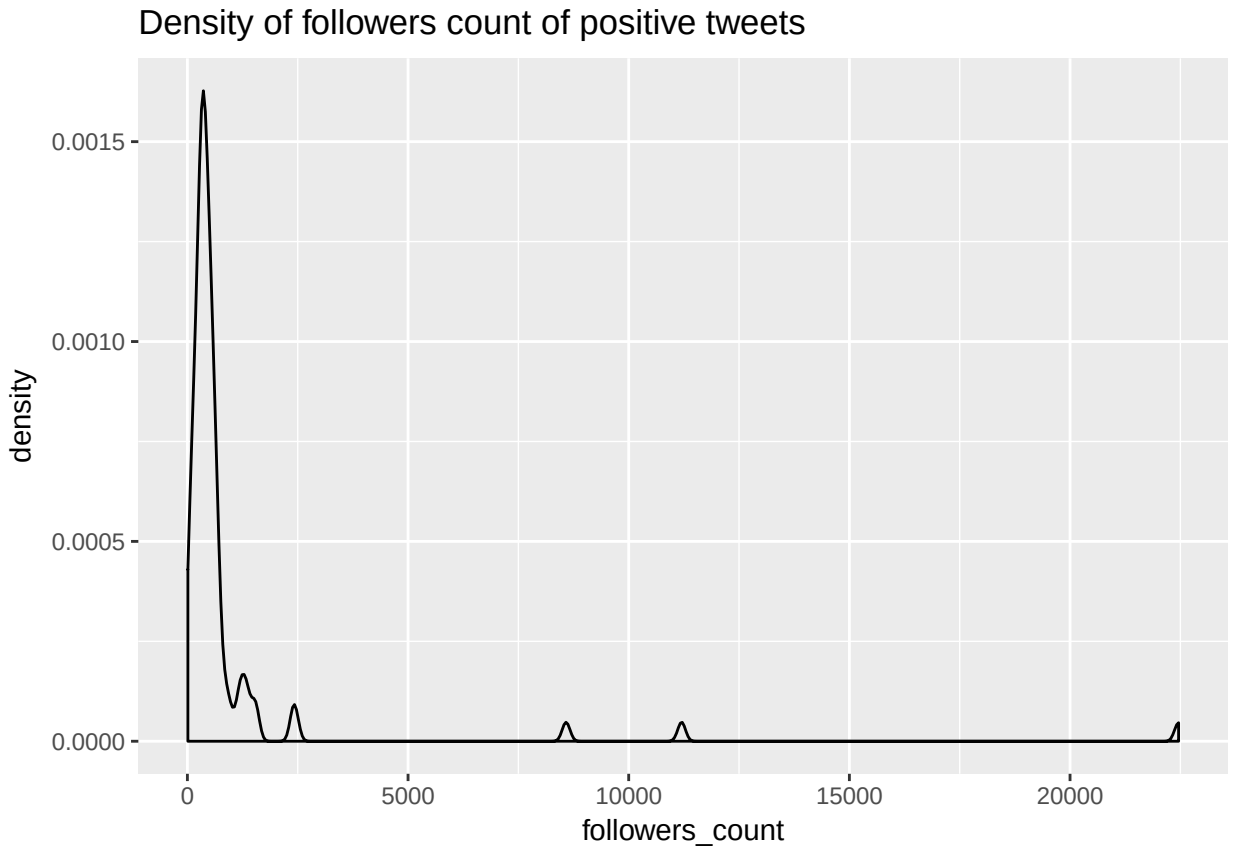
Above analysis based on sentiment analysis result mainly focus on the number of current tweets that falls into each sentiment category. However, different tweets have different influence on the social media. For example, Obama should have a bigger influence on other twitter users than us. If Obama recommends Starbucks coffee on twitter, it is not hard to imagine that there is a big possibility that Starbucks sales is going to increase soon. Some index are also dealing with calculating the influence of a user on the social media, like Klout. Klout ouput is a score ranging 0-100. It takes into account many parameters from different social media including twitter, facebook, Wikipedia and so on.

Measuring the influence of a social media user or a specific tweet is very important. Because it reflects an individual's potential to lead others to engage in a certain act. So measuring influence level is essential to prediction.

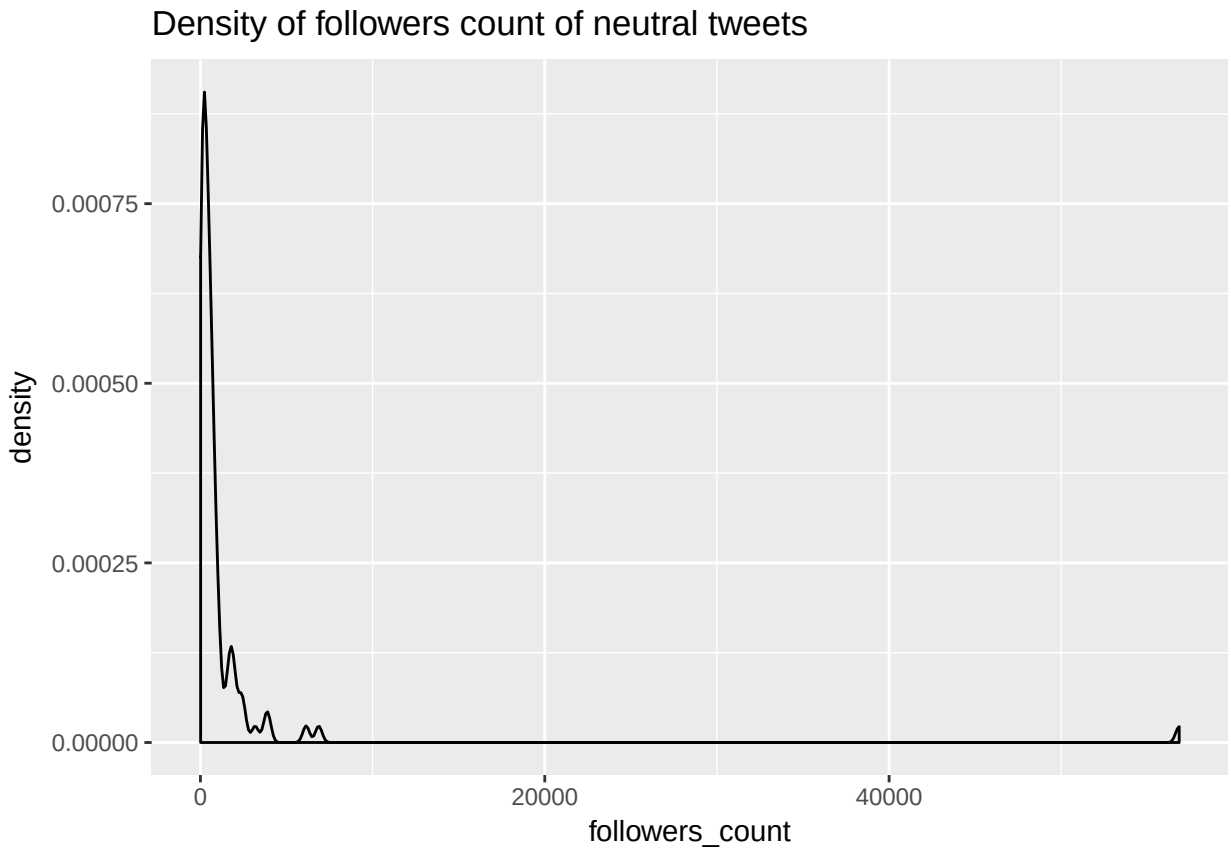
So how to measure a user's influence? Current study indicate parameters like retweet number, followers number, favourites number are important. So in this part, we look at the favourites count and followers count variable. We first plot their density function using the gaussian kernel. Then we estimate their smaple means using bootstrap.

```
## Statistical Model: Bootstrap

index.pos <- which(starclass$sentclass==3)
index.neu <- which(starclass$sentclass==2)
```

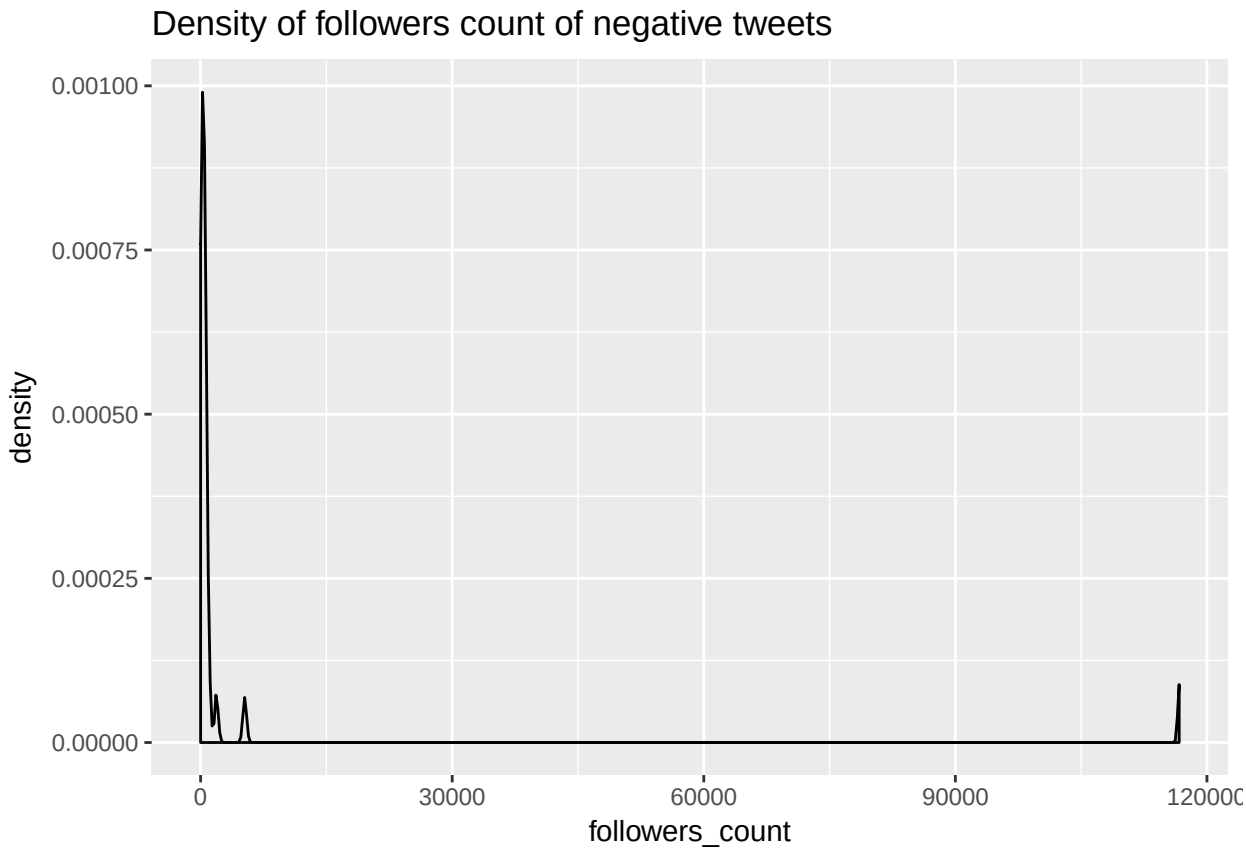
```
index.neg <- which(starclass$sentclass==1)
pos.fav <- starclass[index.pos,]$favourites_count
neu.fav <- starclass[index.neu,]$favourites_count
neg.fav <- starclass[index.neg,]$favourites_count
pos.fol <- starclass[index.pos,]$followers_count
neu.fol <- starclass[index.neu,]$followers_count
neg.fol <- starclass[index.neg,]$followers_count
pos <- starclass[index.pos,]
neu <- starclass[index.neu,]
neg <- starclass[index.neg,]
attach(pos)
attach(neu)
attach(neg)

# plot density of facourites_coun and followers_coun
# followers count
ggplot(data=pos, aes(followers_count))+geom_density(kernel="gaussian")+
  labs(title="Density of followers count of positive tweets")
```



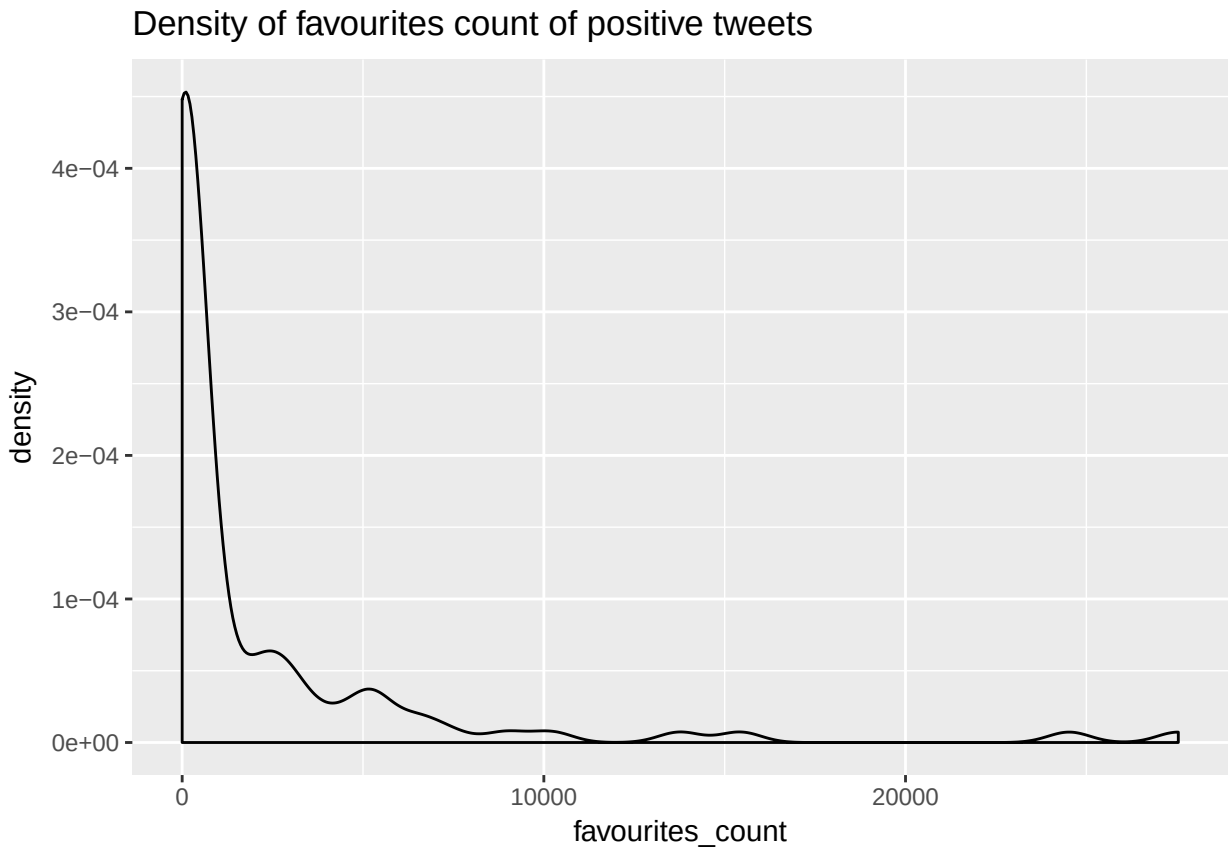

```
ggplot(data=neu, aes(followers_count))+geom_density(kernel="gaussian")+
  ggtitle("Density of followers count of neutral tweets")
```



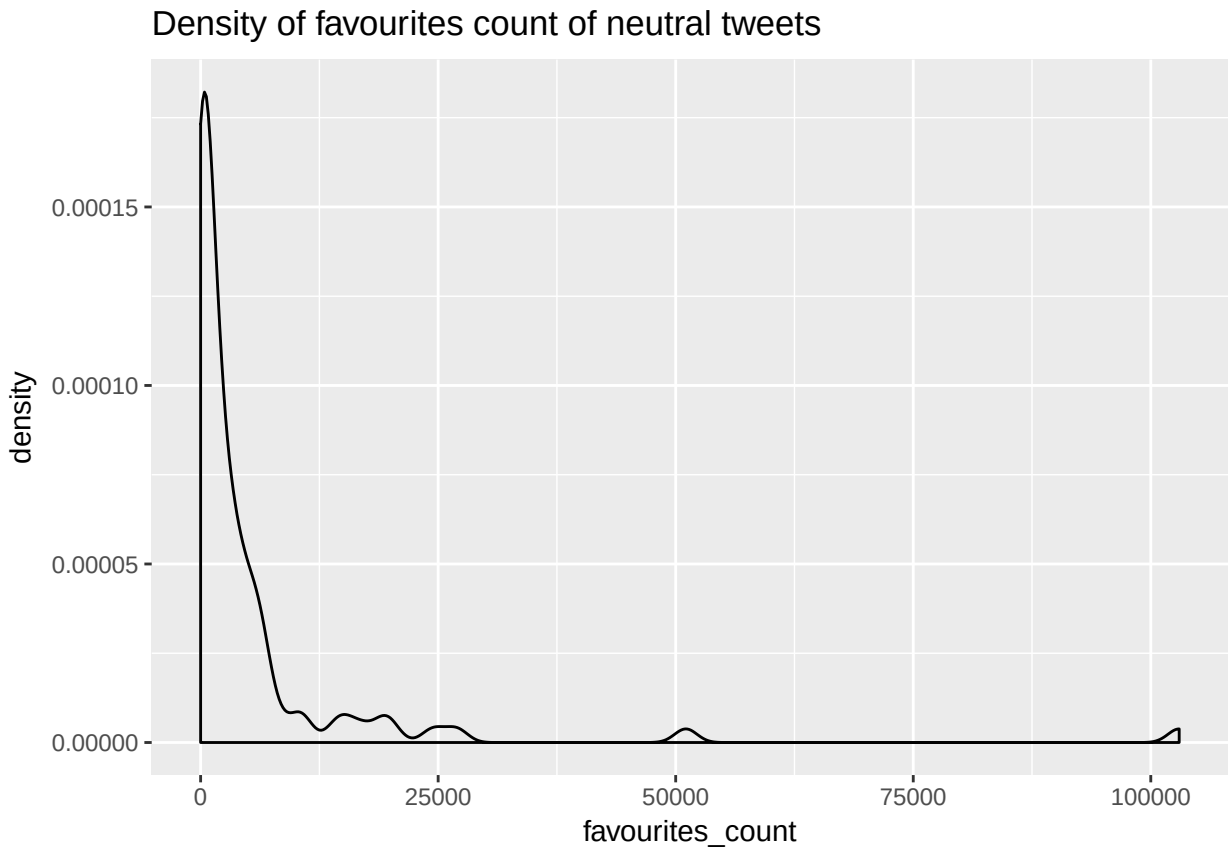

```
ggplot(data=neg, aes(followers_count))+geom_density(kernel="gaussian")+
  ggtitle("Density of followers count of negative tweets")
```



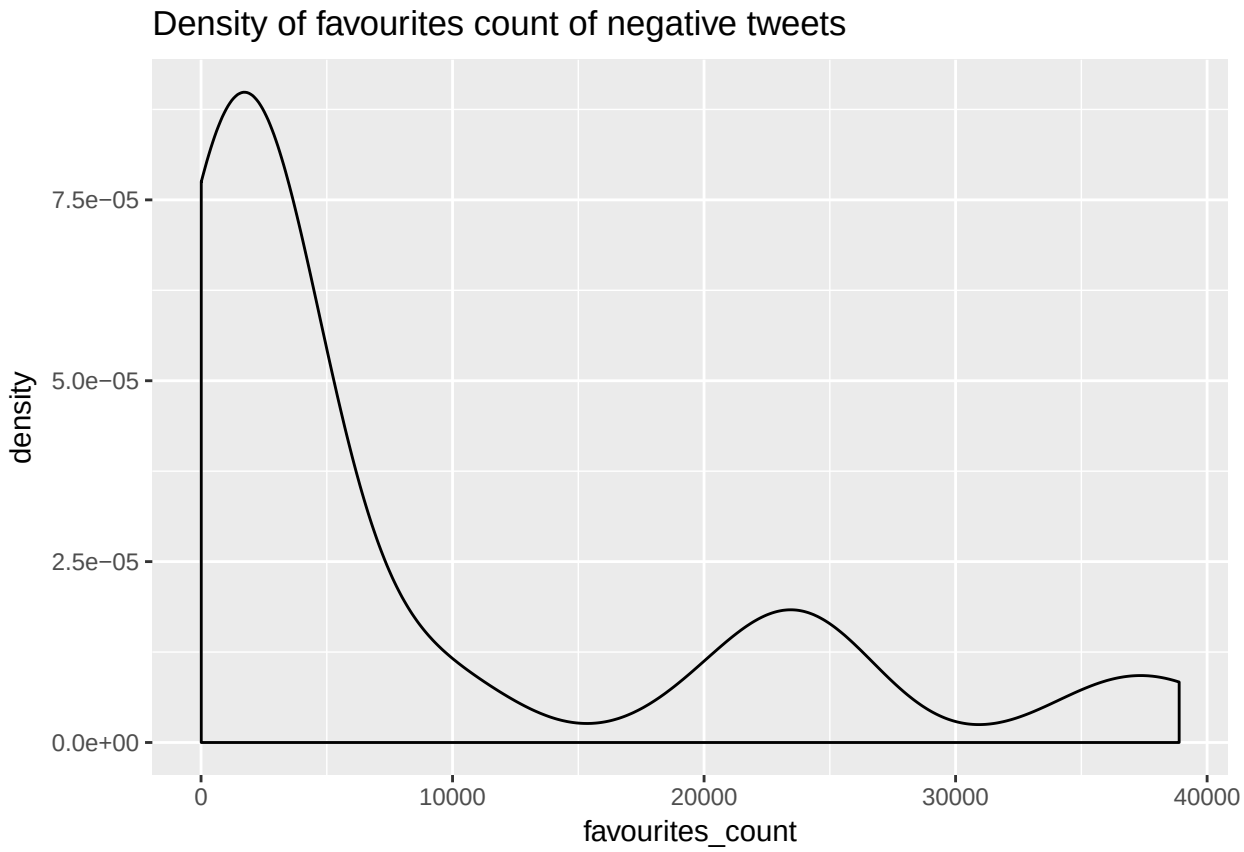

```
# favourites count
ggplot(data=pos, aes(favourites_count))+geom_density(kernel="gaussian")+
  ggtitle("Density of favourites count of positive tweets")
```




```
ggplot(data=neu, aes(favourites_count))+geom_density(kernel="gaussian")+
  ggtitle("Density of favourites count of neutral tweets")
```




```
ggplot(data=neg, aes(favourites_count))+geom_density(kernel="gaussian")+
  ggtitle("Density of favourites count of negative tweets")
```




```
# use bootstrap to find mean of facourites_count for nagative, neutral and positive tweets
library(boot)
```

```
##
## Attaching package: 'boot'

## The following object is masked from 'package:survival':
##
##     aml

## The following object is masked from 'package:lattice':
##
##     melanoma
```

```
funmean <- function(data, index)
{
  x <- data[index]
  return(mean(x))
}

# bootstrap for positive, neutral and negtive
bootout.posfav <- boot(pos.fav, funmean, R = 10000)
bootci.posfav <- boot.ci(bootout.posfav, conf = 0.95, type = "all")
```

```
## Warning in boot.ci(bootout.posfav, conf = 0.95, type = "all"): bootstrap
## variances needed for studentized intervals
```

```
bootout.posfav
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = pos.fav, statistic = funmean, R = 10000)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1* 2029.884 0.7557021    471.6803
```

```
bootci.posfav
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootout.posfav, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   (1105, 2954 )   (1041, 2871 )
##
## Level     Percentile            BCa
## 95%   (1189, 3018 )   (1309, 3299 )
## Calculations and Intervals on Original Scale
```

```
bootout.neufav <- boot(neu.fav, funmean, R = 10000)
bootci.neufav <- boot.ci(bootout.neufav, conf = 0.95, type = "all")
```

```
## Warning in boot.ci(bootout.neufav, conf = 0.95, type = "all"): bootstrap
## variances needed for studentized intervals
```

```
bootout.neufav
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = neu.fav, statistic = funmean, R = 10000)
##
##
## Bootstrap Statistics :
##     original   bias    std. error
## t1* 5094.888 13.77387    1370.653
```

```
bootci.neufav
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootout.neufav, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   (2395, 7768 )   (1976, 7319 )
##
## Level     Percentile            BCa
## 95%   (2871, 8214 )   (3269, 9574 )
## Calculations and Intervals on Original Scale
```

```
bootout.negfav <- boot(neg.fav, funmean, R = 10000)
bootci.negfav <- boot.ci(bootout.negfav, conf = 0.95, type = "all")
```

```
## Warning in boot.ci(bootout.negfav, conf = 0.95, type = "all"): bootstrap
## variances needed for studentized intervals
```

```
bootout.negfav
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = neg.fav, statistic = funmean, R = 10000)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1* 8288.808 -20.72669    2242.247
```

```
bootci.negfav
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootout.negfav, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 3915, 12704 )   ( 3707, 12416 )
##
## Level     Percentile            BCa
## 95%   ( 4162, 12871 )   ( 4694, 13906 )
## Calculations and Intervals on Original Scale
```

```r
# use bootstrap to find mean of followers_count for nagative, neutral and positive tweets
bootout.posfol <- boot(pos.fol, funmean, R = 10000)
bootci.posfol <- boot.ci(bootout.posfol, conf = 0.95, type = "all")
```

```
## Warning in boot.ci(bootout.posfol, conf = 0.95, type = "all"): bootstrap
## variances needed for studentized intervals
```

```r
bootout.posfol
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = pos.fol, statistic = funmean, R = 10000)
##
##
## Bootstrap Statistics :
##     original   bias    std. error
## t1* 933.6947 8.564057    273.6942
```

```r
bootci.posfol
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootout.posfol, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 388.7, 1461.6 )   ( 303.2, 1364.2 )
##
## Level     Percentile            BCa
## 95%   ( 503.2, 1564.1 )   ( 574.4, 1839.5 )
## Calculations and Intervals on Original Scale
```

```r
bootout.neufol <- boot(neu.fav, funmean, R = 10000)
bootci.neufol <- boot.ci(bootout.neufol, conf = 0.95, type = "all")
```

```
## Warning in boot.ci(bootout.neufol, conf = 0.95, type = "all"): bootstrap
## variances needed for studentized intervals
```

```r
bootout.neufol
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
```

```
## boot(data = neu.fav, statistic = funmean, R = 10000)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1* 5094.888 -12.51332    1322.892
```

```
bootci.neufol
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootout.neufol, conf = 0.95, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   (2515, 7700 )   (2181, 7314 )
##
## Level     Percentile            BCa
## 95%   (2876, 8009 )   (3316, 9462 )
## Calculations and Intervals on Original Scale
```

```
bootout.negfol <- boot(neg.fav, funmean, R = 10000)
bootci.negfol <- boot.ci(bootout.negfol, conf = 0.95, type = "all")
```

```
## Warning in boot.ci(bootout.negfol, conf = 0.95, type = "all"): bootstrap
## variances needed for studentized intervals
```

```
bootout.negfol
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = neg.fav, statistic = funmean, R = 10000)
##
##
## Bootstrap Statistics :
##     original   bias    std. error
## t1* 8288.808 25.34923    2243.349
```

```
bootci.negfol
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bootout.negfol, conf = 0.95, type = "all")
##
```

```
## Intervals :
## Level      Normal              Basic
## 95%   ( 3867, 12660 )   ( 3568, 12412 )
##
## Level     Percentile            BCa
## 95%   ( 4166, 13010 )   ( 4571, 13692 )
## Calculations and Intervals on Original Scale
```

Table 4: Bootstrap results for favourites count

Sentiment	Estimate Mean	Percentile Confidence Interval
Positive	2029.8842105	$1188.5065811, 3018.2718391$
Neutral	5094.8876404	$2870.6577774, 8213.5656634$
Negative	8288.8076923	$4162.0646539, 1.2871109 \times 10^4$

Table 5: Bootstrap results for followers count

Sentiment	Estimate Mean	Percentile Confidence Interval
Positive	933.6947368	$503.1607942, 1564.1457353$
Neutral	5094.8876404	$2876.1829436, 8008.902479$
Negative	8288.8076923	$4165.8906453, 1.3009703 \times 10^4$

From the bootstrap results, we can see that for both followers count and favourites count, estimate mean under the negative category is much larger than that under the neutral category and postive category. This result suggests that although negative tweets number is not big, even much smaller than the neutral tweets and positive tweets right now according to the above relative analysis, per negative tweet has larger influence on other users on twitter than a neutral tweet or a positive tweet. This might influence the attitude of public towards Starbucks to convert to the worse in the future.

III. Contribution

Main contribution of this work is that we propose to use ordinal logistic regression to test the if we can use regions to predict sentiment. The result shows that there isn't significant difference in the number of tweets that falls into various sentiment categories among regions.

We also propose to involve a new predictor, influence level, to help prediction. In this project, we perform bootstrap to get estimates for favourites count mean and followers count mean that reflect the influence level. Larger favourites count mean and followers count mean of the negative tweets suggests that the attitude of public towards Starbucks may convert to the worse in the future.

IV. Future Work

Here are some ideas for future work:
a) Optimize the algorithm of getting an influence level
b) Predict the overall public sentiment using predictors like influence level, tweets number in different categories and so on