CSDA 1040: Assignment 2 - Group 4

Jose German, Anjana Pradeep Kumar, Anupama Radhakrishnan Kowsalya, and Xenel Nazar

04/07/2020

1.0 Abstract

Tweets by U.S. airlines passengers can help provide useful real-time customer sentiment information to carriers. Information pulled from passengers' tweets give airlines insight on customer sentiment on current airline operations. Sentiment analysis done on the tweets by text classification help categorize various tweets as either positive, negative, or neutral. Airlines can focus certain measures to counteract any negative sentiments based on tweet information, thus improving overall customer service and customer satisfaction.

2.0 Introduction

The U.S. airline industry is an important driver of the U.S. economy, generating $1.7 trillion in economic activity and more than 10 million jobs (Airlines for America, 2020). The average industry growth from 2015-2020 was 0.3% (IBIS World, 2020). In addition, around 17 U.S. airports are listed in the top 60 busiest airports in the world (The Port Authority of New York and New Jersey, 2019).

Various factors affect the competitive nature between the leading U.S. airlines, one way to differentiate is to provide excellent customer service in their operations. Airlines can use various resources like Twitter, to get real time indicators of customer sentiment. Twitter is a global social platform for public self-expression and conversation in real time (United States Securities and Exchange Commission, 2013).

Reviewing customer sentiment through passengers' twitter posts can help airlines quickly act on any issues that the passengers face and implement any measures to address declining or negative sentiments.

3.0 Objective

The objective is to analyze U.S. airline customer tweets and visualizing keywords that will help stakeholders quickly determine customer sentiment, positive or negative.

4.0 Data Understanding

4.1 About the Data

The data includes twitter posts that were scrapped from February 2015, detailing the problems of each major U.S. Airline, and hosted on Kaggle (Figure Eight, 2019). The data was originally collected by Crowdflower, previously known as Figure Eight Inc., and subsequently acquired by Appen an AI data company (Figure Eight, 2019; Appen Limited, 2020). Contributors helped classify the tweets as either positive, negative, or neutral, and then categorizing negative tweets under various negative reasons (e.g. "late flight", "rude service") (Crowdflower, 2016). Prior to the initial load into Kaggle, certain transformations were done by Ben Hammer, the Co-Founder and CTO at Kaggle (Hamner, 2016).

4.2 Import Data

```
getwd()
```

```
## [1] "/Users/xen/Documents/CSDA 1040/ASSIGNMENT2"
```

```
# Import Data
tweets=read.csv("Tweets.csv",na.strings=c("", "NA"),stringsAsFactors = FALSE)
df=read.csv("Tweets.csv")
```

4.3 Import Packages

```
# Load Libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidytext)
library(ggplot2)
library(stringr)
library(RColorBrewer)
library(wordcloud)
library(tm)
```

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

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

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

```
library(NLP)
library(SentimentAnalysis)
```

```
##
## Attaching package: 'SentimentAnalysis'
```

```
## The following object is masked from 'package:base':
##
##     write
```

```
library(e1071)
library(gmodels)
library(tidyverse)
```

```
## ── Attaching packages ───────────────────────────────────────────────────────────────
────────────────────────────────────────────────────────────────── tidyverse 1.3.0 ──
```

```
## ✓ tibble  3.0.1     ✓ purrr   0.3.4
## ✓ tidyr   1.1.0     ✓ forcats 0.5.0
## ✓ readr   1.3.1
```

```
## ── Conflicts ────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────── tidyverse_conflicts() ──
## x NLP::annotate() masks ggplot2::annotate()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ggthemes)
library(tidyr)
library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following object is masked from 'package:e1071':
##
##     impute
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
packageVersion("Hmisc")
```

```
## [1] '4.4.0'
```

5.0 Data Exploration and Preparation

Overview of Data

```
# Summary of Data
str(tweets)
```

```
## 'data.frame':    14640 obs. of  15 variables:
##  $ tweet_id                    : num  5.7e+17 5.7e+17 5.7e+17 5.7e+17 5.7e+17 ...
##  $ airline_sentiment           : chr  "neutral" "positive" "neutral" "negative" ...
##  $ airline_sentiment_confidence: num  1 0.349 0.684 1 1 ...
##  $ negativereason              : chr  NA NA NA "Bad Flight" ...
##  $ negativereason_confidence   : num  NA 0 NA 0.703 1 ...
##  $ airline                     : chr  "Virgin America" "Virgin America" "Virgin Ameri
ca" "Virgin America" ...
##  $ airline_sentiment_gold      : chr  NA NA NA NA ...
##  $ name                        : chr  "cairdin" "jnardino" "yvonnalynn" "jnardino"
...
##  $ negativereason_gold         : chr  NA NA NA NA ...
##  $ retweet_count               : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ text                        : chr  "@VirginAmerica What @dhepburn said." "@VirginA
merica plus you've added commercials to the experience... tacky." "@VirginAmerica I did
n't today... Must mean I need to take another trip!" "@VirginAmerica it's really aggress
ive to blast obnoxious \"entertainment\" in your guests' faces &amp; they hav"| __trunca
ted__ ...
##  $ tweet_coord                 : chr  NA NA NA NA ...
##  $ tweet_created               : chr  "2015-02-24 11:35:52 -0800" "2015-02-24 11:15:5
9 -0800" "2015-02-24 11:15:48 -0800" "2015-02-24 11:15:36 -0800" ...
##  $ tweet_location              : chr  NA NA "Lets Play" NA ...
##  $ user_timezone               : chr  "Eastern Time (US & Canada)" "Pacific Time (US
& Canada)" "Central Time (US & Canada)" "Pacific Time (US & Canada)" ...
```

```
# Details of Data
summary(tweets)
```

```
##     tweet_id         airline_sentiment  airline_sentiment_confidence
##  Min.   :5.676e+17   Length:14640       Min.   :0.3350
##  1st Qu.:5.686e+17   Class :character   1st Qu.:0.6923
##  Median :5.695e+17   Mode  :character   Median :1.0000
##  Mean   :5.692e+17                      Mean   :0.9002
##  3rd Qu.:5.699e+17                      3rd Qu.:1.0000
##  Max.   :5.703e+17                      Max.   :1.0000
##
##  negativereason     negativereason_confidence   airline
##  Length:14640       Min.   :0.000            Length:14640
##  Class :character   1st Qu.:0.361            Class :character
##  Mode  :character   Median :0.671            Mode  :character
##                     Mean   :0.638
##                     3rd Qu.:1.000
##                     Max.   :1.000
##                     NA's   :4118
##  airline_sentiment_gold     name           negativereason_gold
##  Length:14640           Length:14640       Length:14640
##  Class :character       Class :character   Class :character
##  Mode  :character       Mode  :character   Mode  :character
##
##
##
##
##  retweet_count          text           tweet_coord        tweet_created
##  Min.   : 0.00000   Length:14640       Length:14640       Length:14640
##  1st Qu.: 0.00000   Class :character   Class :character   Class :character
##  Median : 0.00000   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 0.08265
##  3rd Qu.: 0.00000
##  Max.   :44.00000
##
##  tweet_location     user_timezone
##  Length:14640       Length:14640
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
##
```

```
summary(df)
```

```
##     tweet_id         airline_sentiment  airline_sentiment_confidence
##  Min.   :5.676e+17   Length:14640       Min.   :0.3350
##  1st Qu.:5.686e+17   Class :character   1st Qu.:0.6923
##  Median :5.695e+17   Mode  :character   Median :1.0000
##  Mean   :5.692e+17                      Mean   :0.9002
##  3rd Qu.:5.699e+17                      3rd Qu.:1.0000
##  Max.   :5.703e+17                      Max.   :1.0000
##
##  negativereason     negativereason_confidence   airline
##  Length:14640       Min.   :0.000             Length:14640
##  Class :character   1st Qu.:0.361             Class :character
##  Mode  :character   Median :0.671             Mode  :character
##                     Mean   :0.638
##                     3rd Qu.:1.000
##                     Max.   :1.000
##                     NA's   :4118
##  airline_sentiment_gold     name           negativereason_gold
##  Length:14640           Length:14640       Length:14640
##  Class :character       Class :character   Class :character
##  Mode  :character       Mode  :character   Mode  :character
##
##
##
##
##  retweet_count          text           tweet_coord        tweet_created
##  Min.   : 0.00000   Length:14640       Length:14640       Length:14640
##  1st Qu.: 0.00000   Class :character   Class :character   Class :character
##  Median : 0.00000   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 0.08265
##  3rd Qu.: 0.00000
##  Max.   :44.00000
##
##  tweet_location     user_timezone
##  Length:14640       Length:14640
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
##
```

The data contains 14,640 tweets.

5.1 Initial Data cleanup

Remove columns that are not required for analysis

```
# Drop unused columns
tweets<- subset(tweets, select = -c(tweet_id,airline_sentiment_gold,negativereason_gold,
tweet_coord) )
```

Rename columns to simplify

```
# Rename columns
 new_colname<-c("sentiment","confidence","reason","negconfidence","airline","user","retw
eet","text","created","location","timezone")
colnames(tweets)<-new_colname
# View columns
colnames(tweets)
```

```
##  [1] "sentiment"     "confidence"    "reason"        "negconfidence"
##  [5] "airline"       "user"          "retweet"       "text"
##  [9] "created"       "location"      "timezone"
```

Convert the Created Date data from String to Datetime format

```
# Convert created date from string to datetime
tweets$created<-as.Date(tweets$created)
```

```
# Verify column data types
sapply(tweets,class)
```

```
##     sentiment    confidence        reason negconfidence       airline
##   "character"     "numeric"   "character"     "numeric"   "character"
##          user       retweet          text       created      location
##   "character"     "integer"   "character"        "Date"   "character"
##      timezone
##   "character"
```

Replace NAs from Reason column

```
# Replace NA with Unknown
tweets$reason[is.na(tweets$reason)]<-"Unknown"
```

Replace NAs under the Negative Reason Confidence with Mean

```
# Replace NA with mean
confMean=trunc(mean(tweets$negconfidence, na.rm = TRUE))
tweets$negconfidence[is.na(tweets$negconfidence)]<-confMean
```

Replace NAs under Location, with Not Available

```
# Replace NA with Not Available
tweets$location[is.na(tweets$location)]<-"Not Available"
```

Replace NAs under Timezone with Not Available

```
# Replace NA with not Available
tweets$timezone[is.na(tweets$timezone)]<-"Not Available"
```

Remove any other NAs in dataframe

```
# Remove NA's if any
tweets<-na.omit(tweets)
```

Check data cleanup

```
# Checking data cleanup
dim(tweets)
```

```
## [1] 14640    11
```

```
table(is.na(tweets))
```

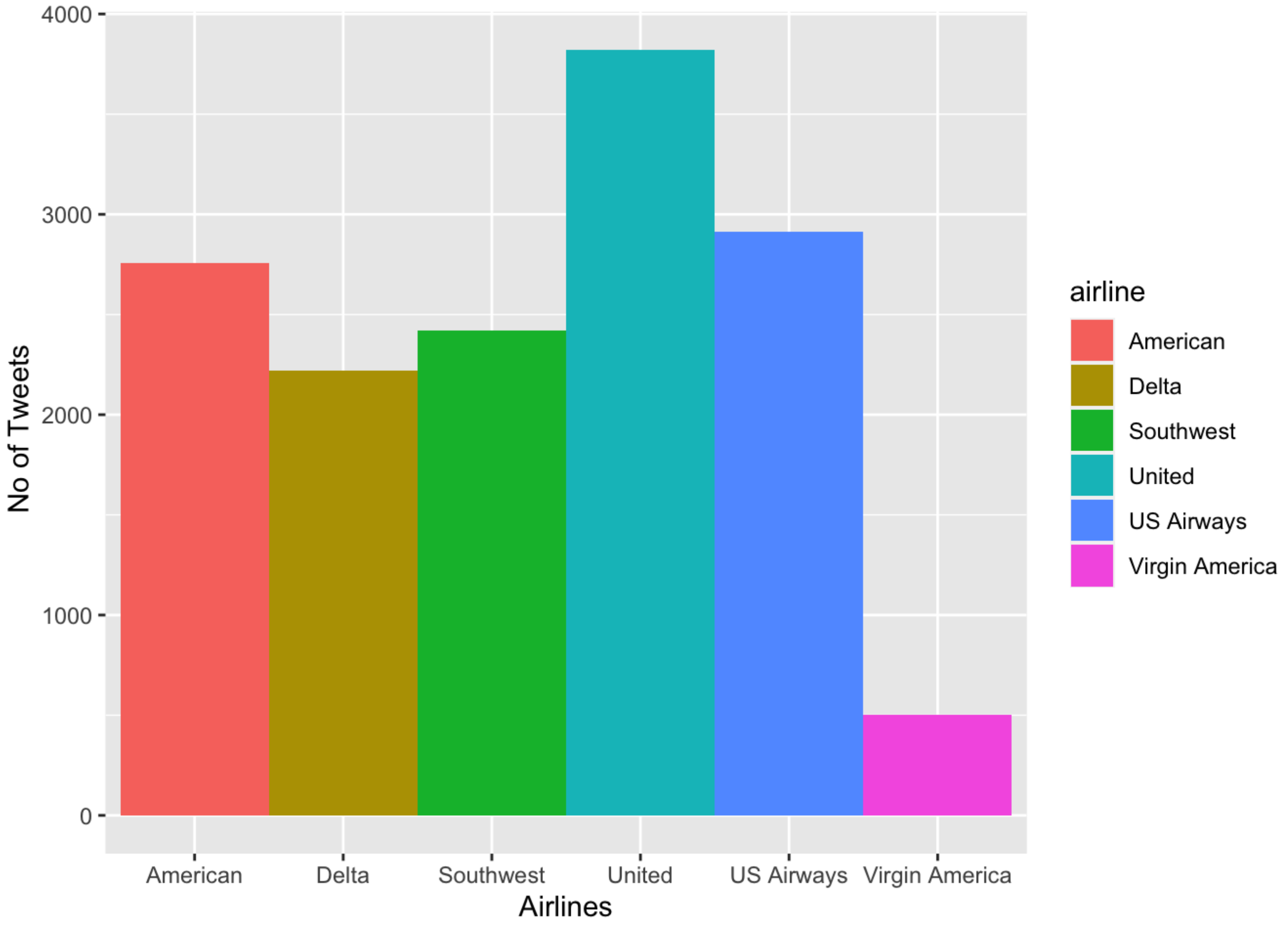
```
##
##  FALSE
## 161040
```

```
#create clean version of all tweets
cleanTweets <- tweets %>% select(9,8)

#rename columns
names(cleanTweets)[1] <- "Created"
names(cleanTweets)[2] <- "Tweet"
```

5.2 Visualization of Data

Plot number of tweets per Airline

```
# of tweets per Airline
count<-tweets %>%
  group_by(airline) %>%
  summarise(tcount1=n(),.groups = 'drop')

#Plotting the number of tweets each airline has received

bg<- ggplot(count) + aes(x= airline,y = tcount1,fill=airline)+
  geom_bar(width = 1, stat = "identity")+
  #geom_histogram()+
  ylab(" No of Tweets") + xlab("Airlines")
bg
```



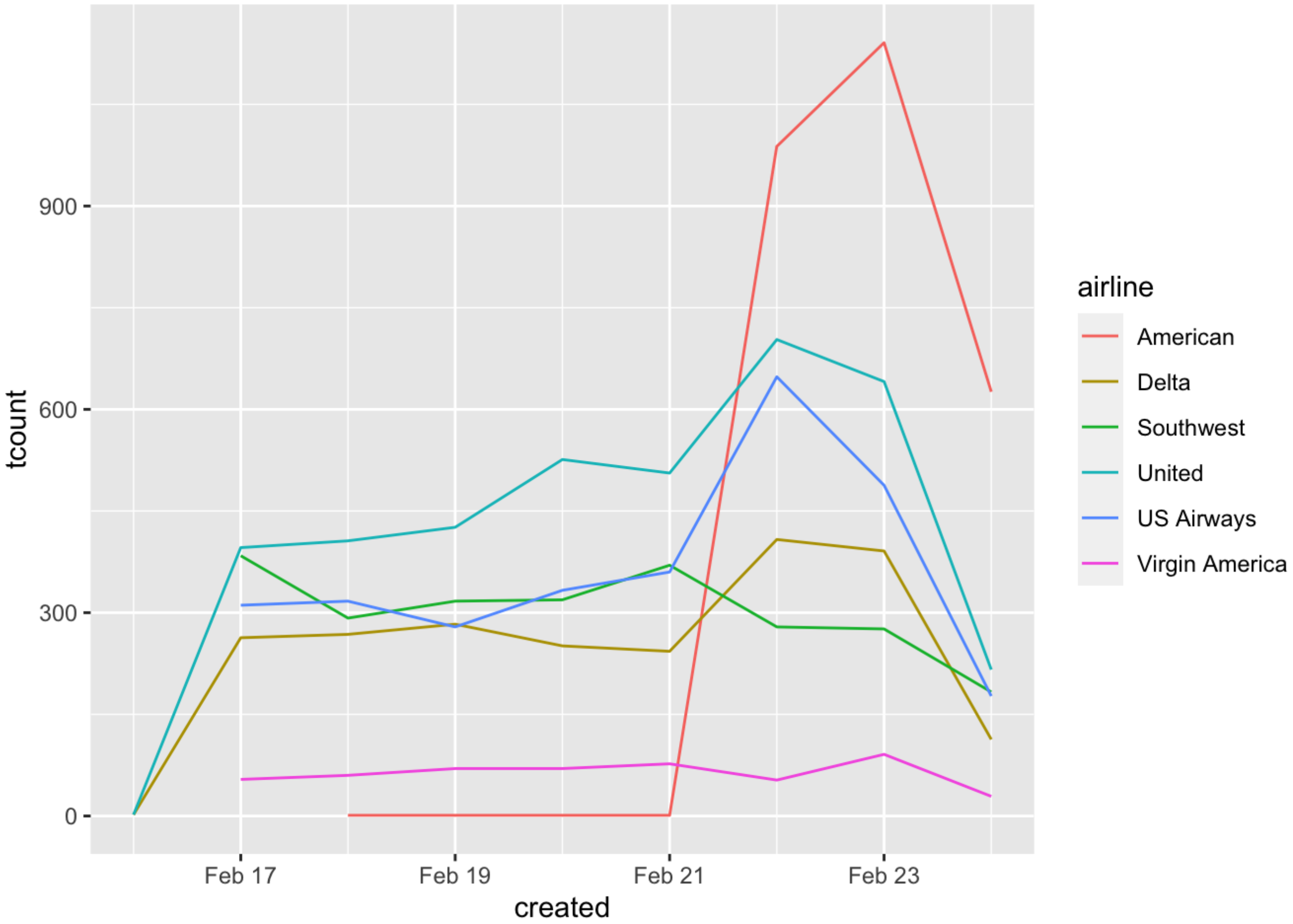
The number of tweets per U.S. airline provides a quick indicator of market share or size of each airline, mirroring actual figures (Mazareanu, 2020). For perspective, US Airways and American began the process of a merger in 2013 (Isidore, 2013). Virgin America, a smaller player in the industry, was then acquired by Alaska airlines in 2016 (Alaska Airlines, Inc., 2016).

Plot tweets per Airline with time period

```
# Plot tweets by Airline
tweetsbyAirline<-tweets%>%
  group_by(airline,created)%>%
  summarise(tcount=n())
```

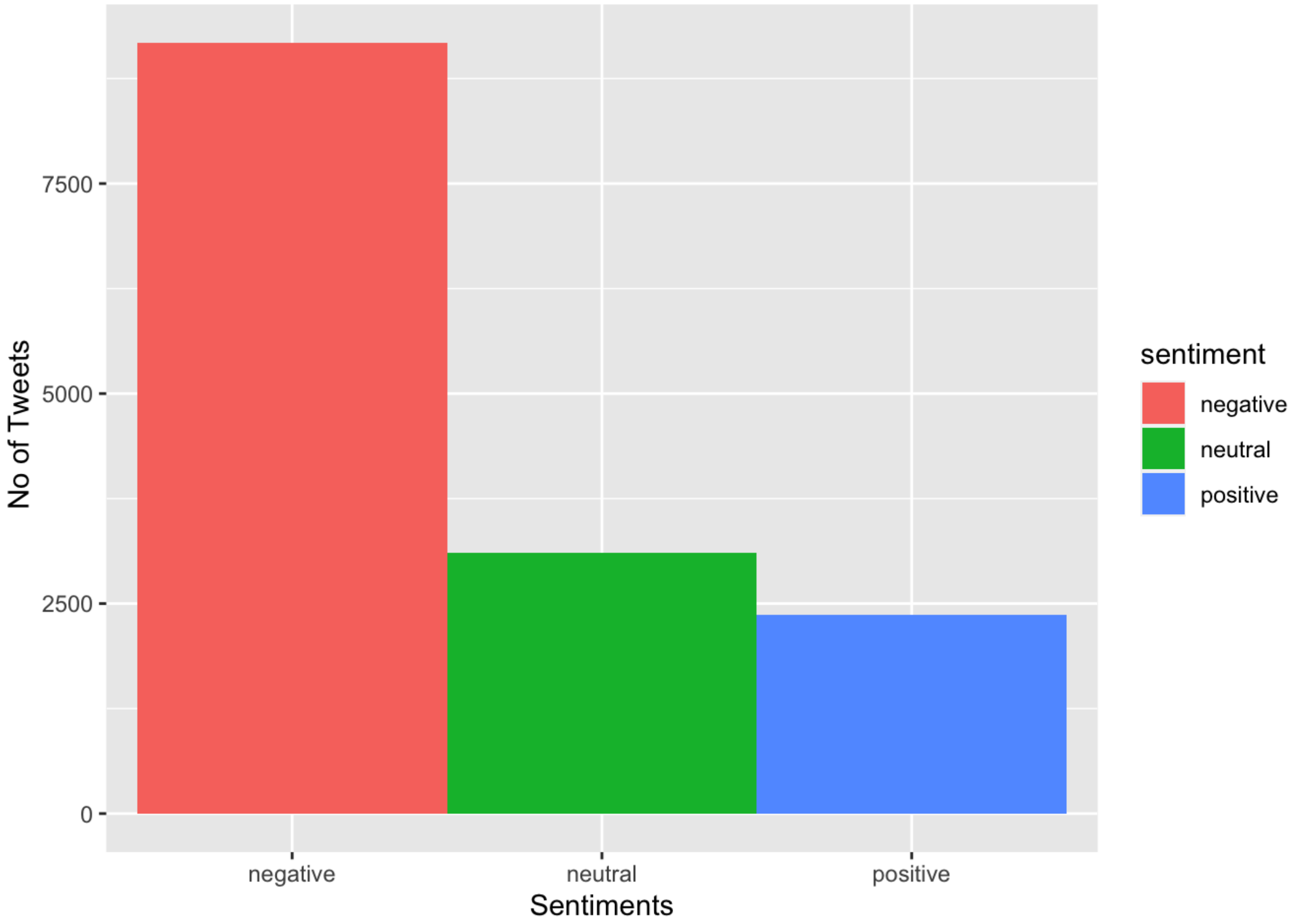
```
## `summarise()` regrouping output by 'airline' (override with `.groups` argument)
```

```
tweetsbyAirlinePlot=ggplot()+geom_line(data=tweetsbyAirline,aes(x=created,y=tcount,group
=airline,color=airline))
tweetsbyAirlinePlot
```



Plot tweets by Sentiment (Positive, Neutral, or Negative)

```
#counting the number of each type of sentiments
count_senti<-tweets %>%
  group_by(sentiment) %>%
  summarise(tc=n(),.groups = 'drop')

#Plotting the number of each type of sentiments
bg2<- ggplot(count_senti) + aes(x= sentiment ,y = tc,fill=sentiment)+
  geom_bar(width = 1, stat = "identity")+
  #geom_histogram()+
  ylab(" No of Tweets") + xlab("Sentiments")

bg2
```

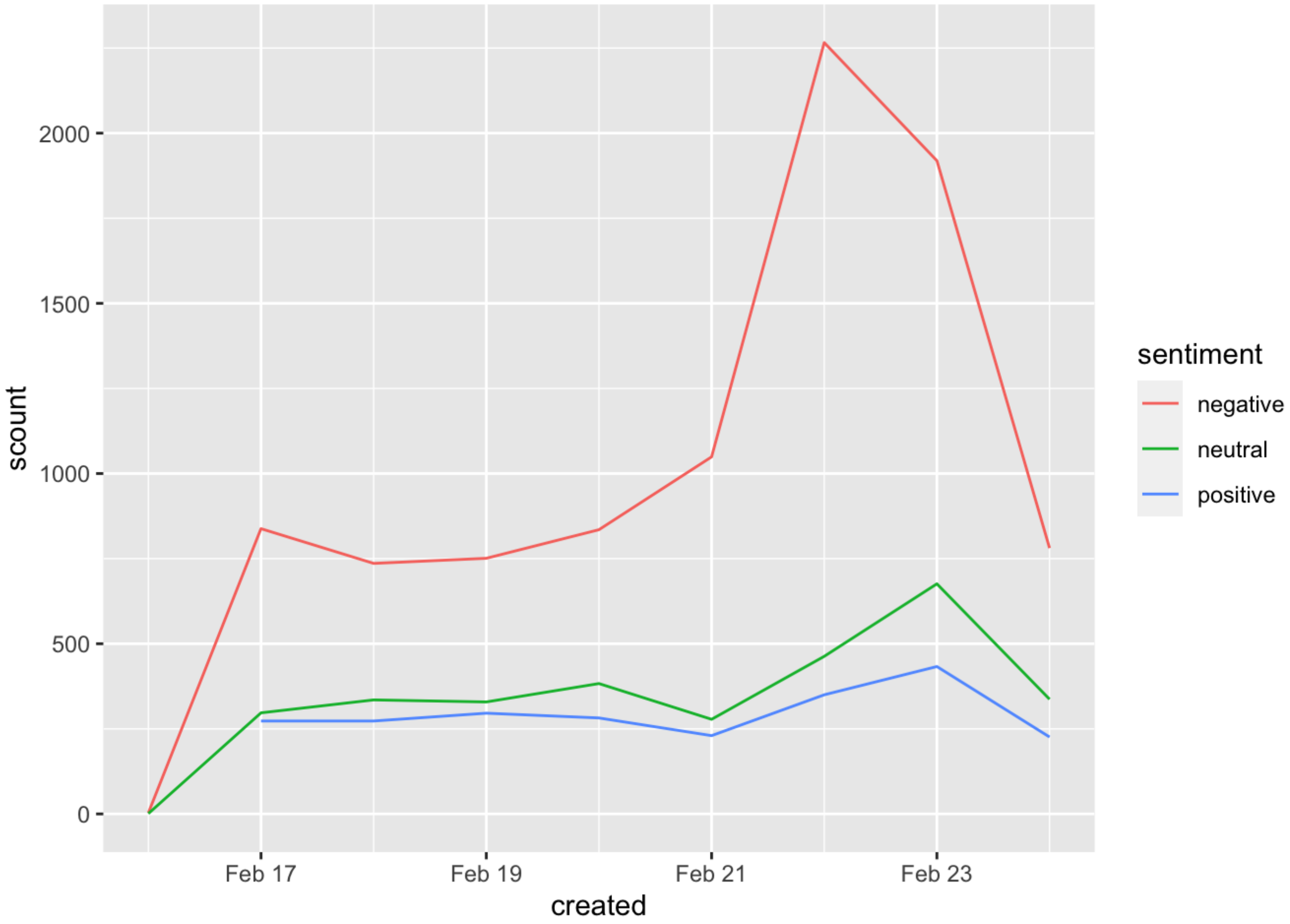


Plot tweets by Sentiment with time period

```
# Plot tweets by Sentiment from when tweet was posted
tweetsbySentiment<-tweets%>%
  group_by(sentiment,created)%>%
  summarise(scount=n())
```

```
## `summarise()` regrouping output by 'sentiment' (override with `.groups` argument)
```

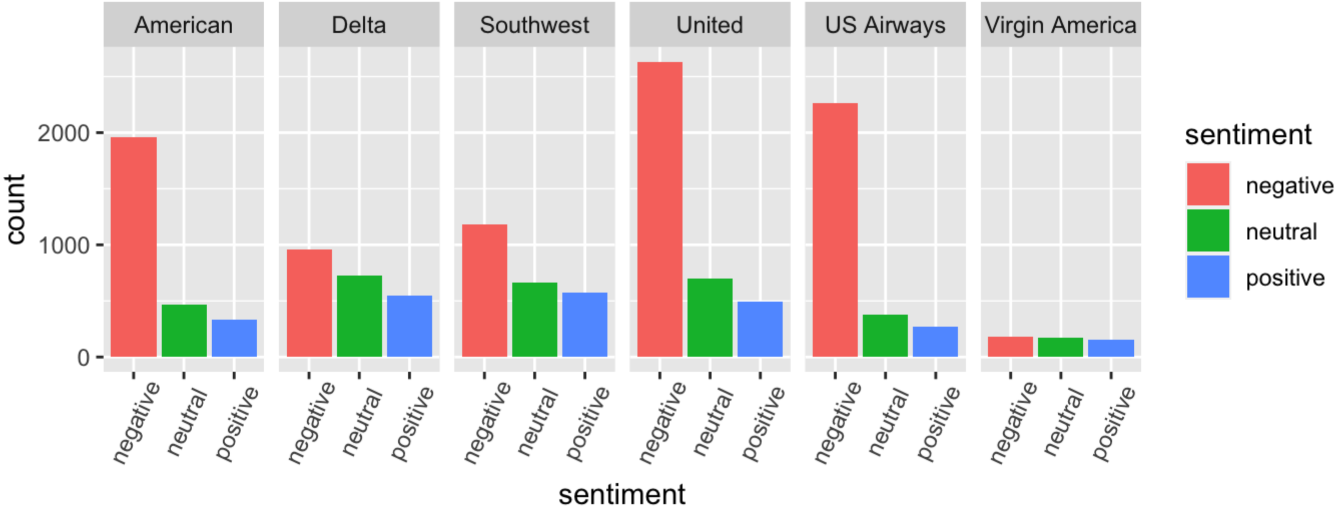
```
tweetsbySentimentPlot=ggplot()+geom_line(data=tweetsbySentiment,aes(x=created,y=scount,g
roup=sentiment,color=sentiment))
tweetsbySentimentPlot
```



Reviewing the tweets, we can see a lot more volatility in negative sentiments versus positive or neutral tweets.

Sentiment breakdown by airline

```
# Sentiment count by airline
ggplot(tweets, aes(x = sentiment, fill = sentiment)) +
  geom_bar() +
  facet_grid(. ~ airline) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6),
        plot.margin = unit(c(3,0,3,0), "cm"))
```



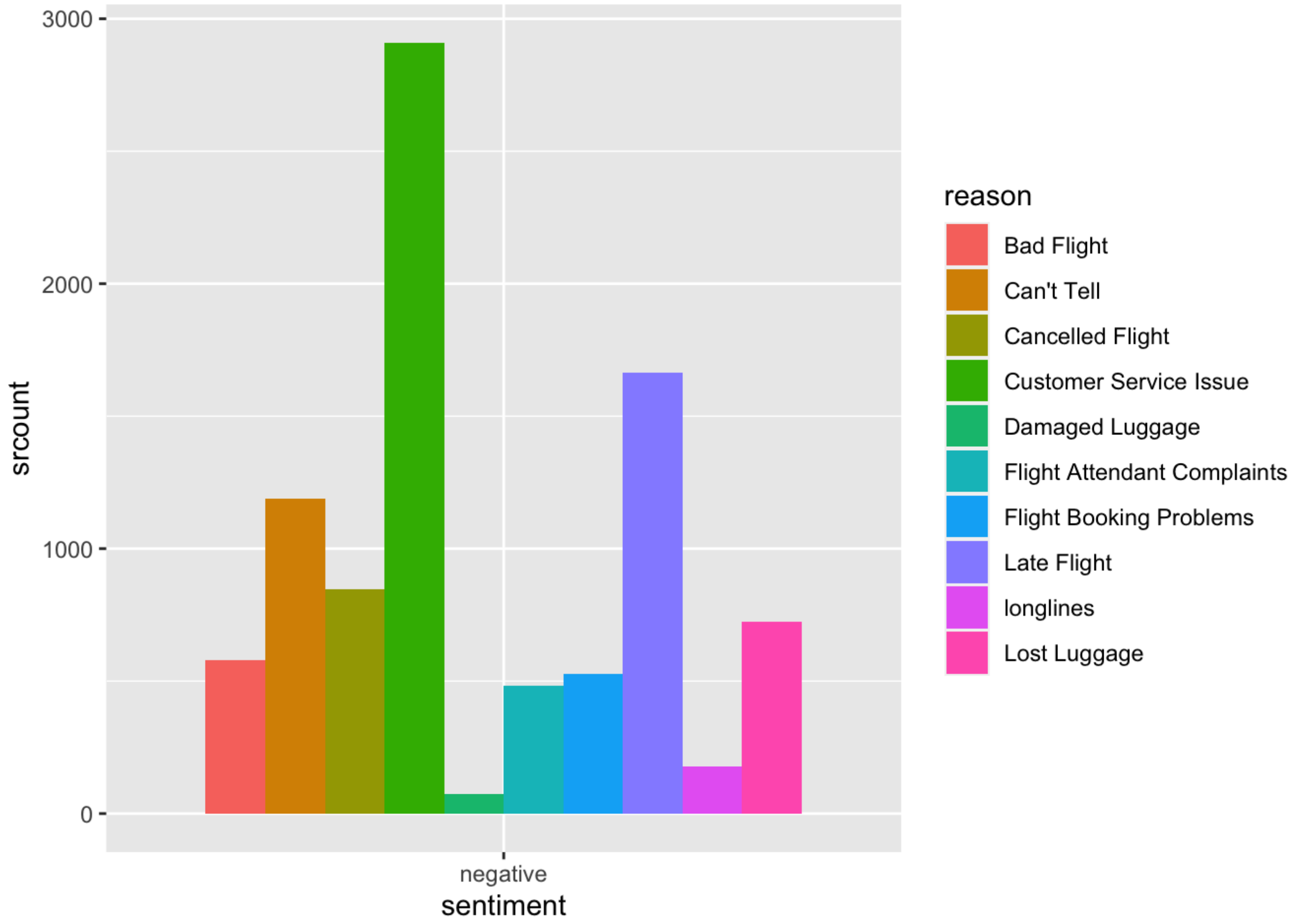
The breakdown of the sentiment by airline, shows that no airline was immune to negative sentiment. However, certain airlines like United, US Airways, and American were more likely to receive tweets with negative sentiments compared to others.

Plot tweets by Negative Sentiment Reason

```
# Plot tweets by Sentiment Reason
tweetsbySentimentreason<-tweets%>%
  filter(sentiment=="negative")%>%
  group_by(sentiment,reason)%>%
  summarise(srcount=n())
```

```
## `summarise()` regrouping output by 'sentiment' (override with `.groups` argument)
```

```
tweetsbySentimentreasonPlot<-ggplot(tweetsbySentimentreason) +
  geom_col(
    mapping = aes(x = sentiment, y = srcount, fill = reason), position = "dodge"
  )
tweetsbySentimentreasonPlot
```



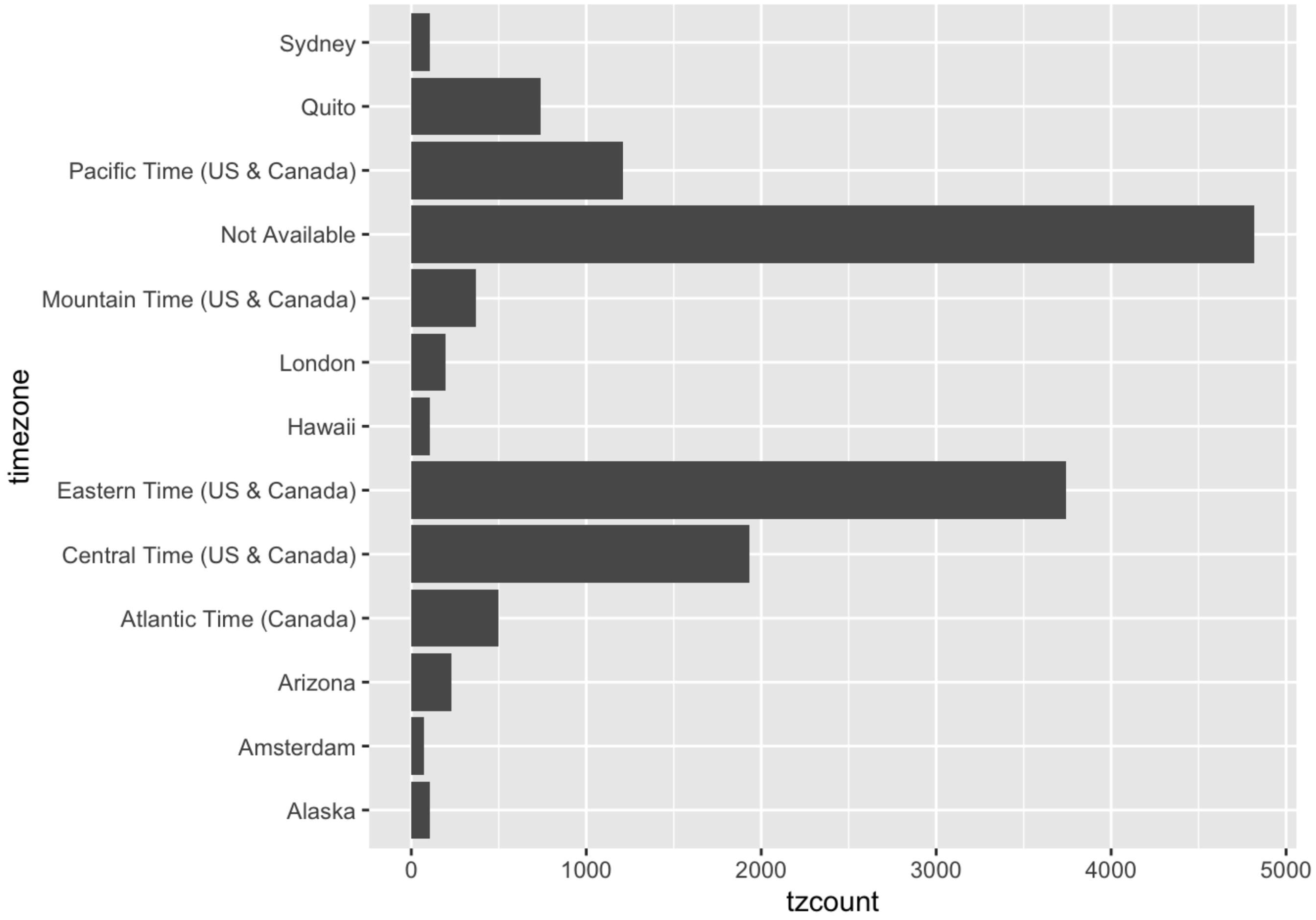
Customer service issues was listed as the main reason of negative sentiment. The airlines can address this by having measures in place to counteract negative sentiment, such as interacting with passengers directly on twitter to resolve their issues and/or provide compensation.

Plot tweets by Timezone

```
# Plot tweets by Timezone
tweetsbyTimezone<-tweets%>%
  group_by(timezone)%>%
  summarise(tzcount=n())%>%
  filter(tzcount>50)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
tweetsbyTimezonePlot<-ggplot(data = tweetsbyTimezone) +
  geom_col(mapping = aes(x = tzcount, y = timezone))
tweetsbyTimezonePlot
```



Timezone details of the tweets can help airlines determine which locations regional teams and customer support need to focus on.

```
# Get airlines listed in tweets
allAirlines <- distinct(tweets, airline)
allAirlines <- lapply(allAirlines, as.character)
print(allAirlines)
```

```
## $airline
## [1] "Virgin America" "United"         "Southwest"      "Delta"
## [5] "US Airways"     "American"
```

```
sentimentCount<-df %>%
  group_by(airline_sentiment) %>%
  summarise(tc=n(),.groups = 'drop')
```

```
#Get sentiment counts for each airlines
americanSentimentCount<-df%>%
  group_by(airline_sentiment)%>%
  filter(airline == 'American')%>%
  summarise(tc1=n(),.groups = 'drop')

deltaSentimentCount<-df%>%
  group_by(airline_sentiment)%>%
  filter(airline == 'Delta')%>%
  summarise(tc2=n(),.groups = 'drop')

southwestSentimentCount<-df%>%
  group_by(airline_sentiment)%>%
  filter(airline == 'Southwest')%>%
  summarise(tc3=n(),.groups = 'drop')

unitedSentimentCount<-df%>%
  group_by(airline_sentiment)%>%
  filter(airline == 'United')%>%
  summarise(tc4=n(),.groups = 'drop')

usairwaysSentimentCount<-df%>%
  group_by(airline_sentiment)%>%
  filter(airline == 'US Airways')%>%
  summarise(tc5=n(),.groups = 'drop')

virginSentimentCount<-df%>%
  group_by(airline_sentiment)%>%
  filter(airline == 'American')%>%
  summarise(tc6=n(),.groups = 'drop')
```

5.3 Tweets text cleanup

```
# Cleaning tweet
tweets$text <- str_replace_all(tweets$text,"@[a-z,A-Z]*","")
tweets$text <- gsub("&amp", "", tweets$text)
tweets$text <- gsub("(RT|via)((?:\\b\\W*@\\w+)+)", "", tweets$text)
tweets$text <- gsub("@\\w+", "", tweets$text)
tweets$text <- gsub("[[:punct:]]", "",tweets$text)
tweets$text <- gsub("[[:digit:]]", "", tweets$text)
tweets$text <- gsub("http\\w+", "",tweets$text)
tweets$text <- gsub("[ \t]{2,}", "",tweets$text)
tweets$text <-gsub("^\\s+|\\s+$", "", tweets$text)
```

5.4 Corpus Setup

```
# Corpus Setup
tweetCorpus<-SimpleCorpus(VectorSource(tweets$text))
print(tweetCorpus)
```

```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 14640
```

5.5 Corpus tweet cleanup

```
# Remove punctuation
tweetCorpus<-tm_map(tweetCorpus,removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, removePunctuation): transformation
## drops documents
```

```
# Remove numbers
 tweetCorpus<-tm_map(tweetCorpus,removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, removeNumbers): transformation drops
## documents
```

```
# To lower case
 tweetCorpus<-tm_map(tweetCorpus,content_transformer(tolower))
```

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

```
# Remove white space
tweetCorpus<-tm_map(tweetCorpus,stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, stripWhitespace): transformation
## drops documents
```

```
# Remove stopwords
tweetCorpus<-tm_map(tweetCorpus,removeWords,stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, removeWords, stopwords("english")):
## transformation drops documents
```

```
# Remove stop word flight
customstopwords <- c("flight","airline","get","got","dont","will","ive","","told","day",
"still","can","cant")
tweetCorpus<-tm_map(tweetCorpus,removeWords,customstopwords)
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, removeWords, customstopwords):
## transformation drops documents
```

5.6 Remove Stopwords

```
# Remove stopwords
tweetCorpus<-tm_map(tweetCorpus,removeWords,stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, removeWords, stopwords("english")):
## transformation drops documents
```

```
# Additional stopwords to remove
customstopwords <- c("flight","airline","get","got","dont","will","ive","","told","day",
"still","can","cant")
tweetCorpus<-tm_map(tweetCorpus,removeWords,customstopwords)
```

```
## Warning in tm_map.SimpleCorpus(tweetCorpus, removeWords, customstopwords):
## transformation drops documents
```

Verify Corpus

```
# Verify corpus
tweetCorpus[[8]]$content
```

```
## [1] "really missed  prime opportunity  men without hats parody "
```

5.7 Create Term Document Matrix

```
# Term Document Matrix
tdmtweetair<-TermDocumentMatrix(tweetCorpus)
inspect(tdmtweetair)
```

```
## <<TermDocumentMatrix (terms: 17925, documents: 14640)>>
## Non-/sparse entries: 112798/262309202
## Sparsity           : 100%
## Maximal term length: 51
## Weighting          : term frequency (tf)
## Sample             :
##            Docs
## Terms       10374 11187 12416 1539 2945 3684 3698 3995 7248 8783
##   cancelled     0     0     0    1    1    0    0    0    0    1
##   customer      0     0     0    0    0    0    0    0    0    0
##   flights       0     0     0    0    0    1    0    0    0    0
##   help          0     0     0    0    0    0    0    0    0    0
##   hold          0     0     0    0    0    0    0    0    0    0
##   just          0     1     0    0    0    2    0    0    0    0
##   now           0     0     0    0    0    1    0    0    0    1
##   service       0     0     1    0    0    0    0    0    0    0
##   thanks        0     0     0    0    0    0    0    0    0    0
##   time          2     0     0    0    0    0    0    0    0    1
```

Convert to a Matrix

```
# Convert to Matrix
mtweet<-as.matrix(tdmtweetair)
wordcount<-sort(rowSums(mtweet),decreasing = TRUE)
```

Check Word Frequency

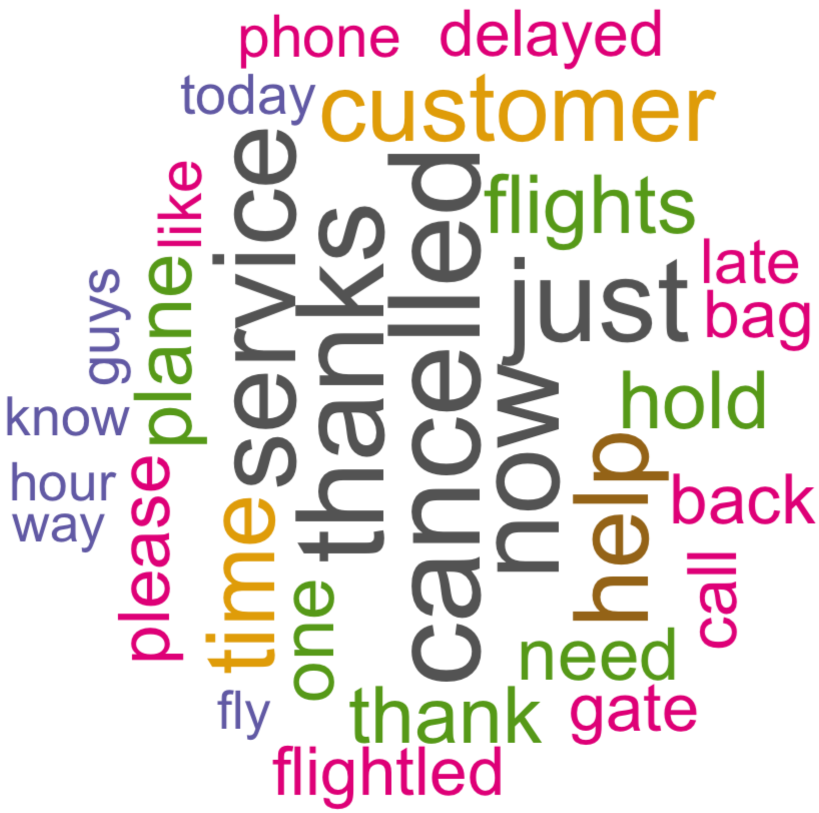
```
# Word Frequency
wordfrequency<-data.frame(text=names(wordcount),freq=wordcount)
head(wordfrequency)
```

```
##                text freq
## cancelled cancelled 1019
## thanks       thanks  970
## now             now  930
## just           just  922
## service     service  905
## help           help  805
```

5.8 Additional Visualizations

Plot Word Cloud

```
# Word Cloud
wordfrequency<-data.frame(text=names(wordcount),freq=wordcount)
wordcloud(words =wordfrequency$text,freq=wordfrequency$freq,min.freq = 1,
          max.words=30,random.order = FALSE,rot.per=0.35,colors = brewer.pal(8,"Dark2"))
```

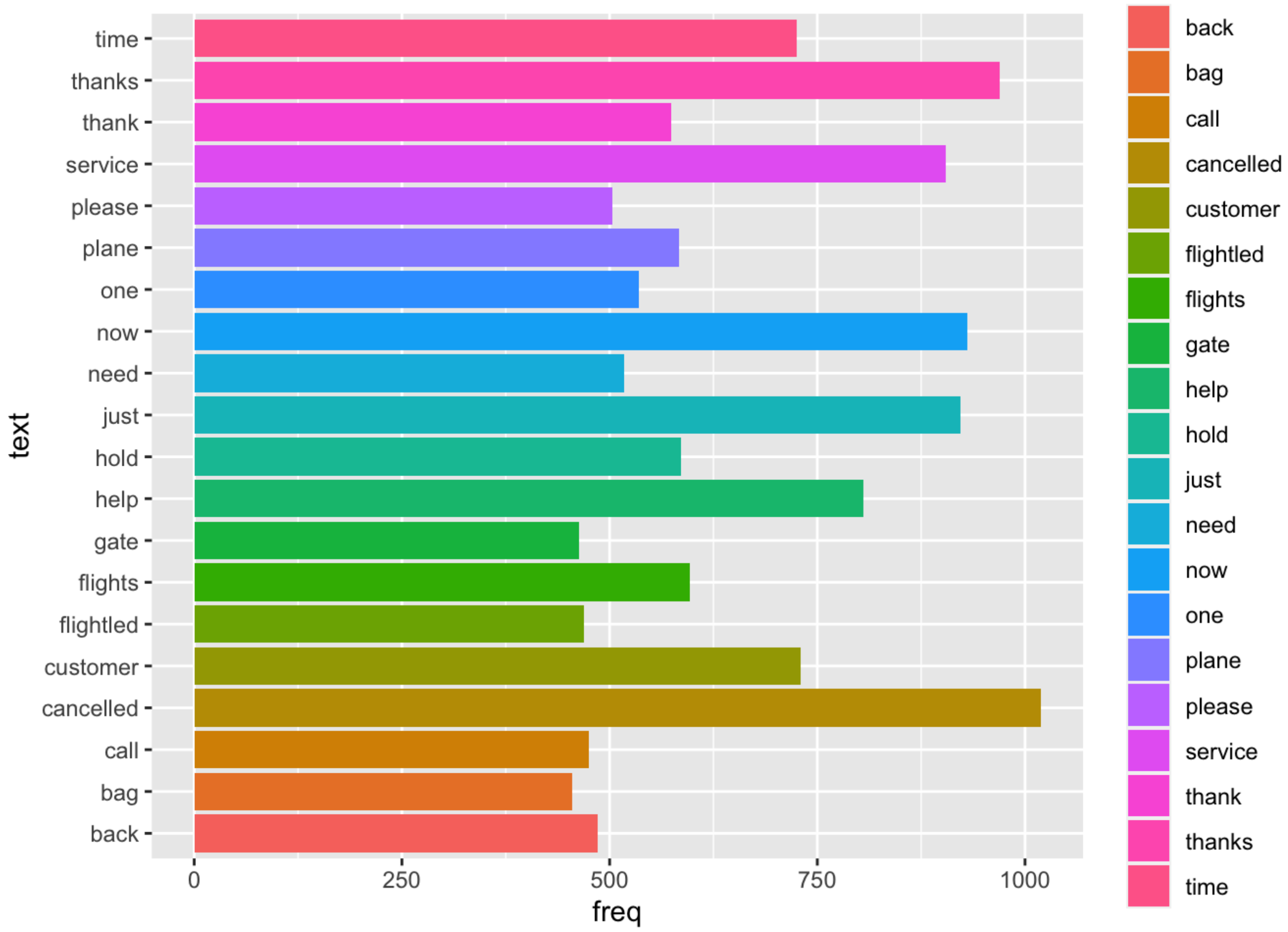


Plot Top 20 words from tweets

```
# Top 20 words from the tweets
uniquewords<-wordfrequency%>%
  arrange(-freq)%>%
  top_n(20)
```

```
## Selecting by freq
```

```
uniqueWordsPlot<-ggplot(uniquewords) +
  geom_col(
    mapping = aes(x = freq, y = text, fill = text), position = "dodge"
  )

uniqueWordsPlot
```



Find terms that appear at least a 100 times in the Term Document Matrix

```
# Find terms appearing at least 100 times from TDM
findFreqTerms(tdmtweetair,100)
```

```
##   [1] "experience"  "another"     "didnt"       "need"        "take"
##   [6] "today"       "trip"        "really"      "bad"         "flying"
##  [11] "seats"       "every"       "fly"         "time"        "wont"
##  [16] "yes"         "missed"      "without"     "now"         "well"
##  [21] "good"        "hour"        "youre"       "know"        "better"
##  [26] "much"        "already"     "even"        "great"       "havent"
##  [31] "yet"         "travel"      "thanks"      "first"       "lax"
##  [36] "nothing"     "couldnt"     "due"         "help"        "last"
##  [41] "seat"        "two"         "week"        "awesome"     "please"
##  [46] "want"        "available"   "times"       "love"        "making"
##  [51] "free"        "guys"        "response"    "status"      "anything"
##  [56] "say"         "miss"        "air"         "booking"     "just"
##  [61] "problems"    "hours"       "online"      "left"        "number"
##  [66] "one"         "call"        "phone"       "use"         "flights"
##  [71] "nice"        "best"        "ever"        "way"         "done"
##  [76] "book"        "working"     "getting"     "next"        "new"
##  [81] "ticket"      "night"       "ago"         "website"     "never"
##  [86] "since"       "think"       "called"      "check"       "tried"
##  [91] "someone"     "trying"      "hold"        "tonight"     "bag"
##  [96] "lost"        "change"      "credit"      "reservation" "customer"
## [101] "let"         "service"     "baggage"     "booked"      "airlines"
## [106] "crew"        "like"        "plane"       "morning"     "bags"
## [111] "going"       "thank"       "delayed"     "late"        "cancelled"
## [116] "business"    "whats"       "home"        "back"        "flightled"
## [121] "jfk"         "class"       "passengers"  "find"        "anyone"
## [126] "people"      "line"        "info"        "weather"     "come"
## [131] "phl"         "right"       "gate"        "doesnt"      "problem"
## [136] "delay"       "long"        "thats"       "wait"        "customers"
## [141] "hotel"       "said"        "give"        "always"      "luggage"
## [146] "work"        "answer"      "understand"  "boarding"    "tickets"
## [151] "dfw"         "email"       "flighted"    "stuck"       "agent"
## [156] "airport"     "waiting"     "follow"      "many"        "rebook"
## [161] "tomorrow"    "worst"       "issue"       "missing"     "sitting"
## [166] "sent"        "keep"        "refund"      "looking"     "checked"
## [171] "see"         "tell"        "delays"      "says"        "hope"
## [176] "board"       "made"        "care"        "ill"         "also"
## [181] "make"        "pay"         "app"         "system"      "finally"
## [186] "sure"        "wifi"        "person"      "update"      "able"
## [191] "united"      "issues"      "agents"      "rude"        "appreciate"
## [196] "put"         "connection"  "flightr"     "miles"       "staff"
## [201] "rebooked"    "jetblue"     "fleek"       "fleets"
```

5.9 Create Document Term Matrix

```
# Create Document Term Matrix
dtmtweetairline<-DocumentTermMatrix(tweetCorpus)
dtmtweetairline
```

```
## <<DocumentTermMatrix (documents: 14640, terms: 17925)>>
## Non-/sparse entries: 112798/262309202
## Sparsity           : 100%
## Maximal term length: 51
## Weighting          : term frequency (tf)
```

Convert Sentiment to a Factor

```
# Convert Sentiment to Factors
tweets$sentiment <- as.factor(tweets$sentiment)
```

6.0 Modeling

6.1 Train-Test Partitioning

The data was partitioned with a 80-20 split

```
# Partition index data 80-20
train_index <- sample(1:nrow(tweets), 0.8 * nrow(tweets))
test_index <- setdiff(1:nrow(tweets), train_index)
```

6.2 Set Train and Test Split

```
# Train and Test set for document matrix,corpus and original dataframe
doc.train <- dtmtweetairline[train_index,]
doc.test <- dtmtweetairline[test_index,]

corpus.train <-tweetCorpus[train_index]
corpus.test <- tweetCorpus[test_index]

tweets.train<-tweets[train_index,]
tweets.test<-tweets[test_index,]
```

6.3 Term Frequency

```
# Get terms at least 5 times in document matrix
fivefreq <- findFreqTerms(doc.train, 5)
```

```
# Update the document matrix for frequent terms
dtm.train<- DocumentTermMatrix(corpus.train, control=list(dictionary = fivefreq))
dtm.test <- DocumentTermMatrix(corpus.test, control=list(dictionary = fivefreq))
```

6.4 Boolean Term Freqency Conversion

```
# Convert term frequency to boolean
convert_count <- function(x) {
  y <- ifelse(x > 0, 1,0)
  y <- factor(y, levels=c(0,1), labels=c("No", "Yes"))
  y
}

train<-apply(dtm.train,2,convert_count)
test<-apply(dtm.test,2,convert_count)
```

6.5 Naives Bayes Algorithm

The Naive Bayes text classification algorithm will be applied, specifically the multinomial Naive Bayes algorithm. This method determines the absence or presence of features in a Boolean format, replacing term frequencies. In sentiment classification, the presence/absence of a word is more important than the frequency of a word (Jurafsky, 2019).

```
# Naives Bayes Classification Setup
senticlassifier <- naiveBayes(train, tweets.train$sentiment, laplace = 1)
pred<-predict(senticlassifier,test)
```

7.0 Evaluation

7.1 Model Evaluation

```
# Evaluating the model
CrossTable(pred, tweets.test$sentiment,
           prop.chisq = FALSE, prop.t = FALSE, prop.r = FALSE,
           dnn = c('predicted', 'actual'))
```

```
## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Col Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  2928 
## 
## 
##              | actual 
##    predicted |  negative |   neutral |  positive | Row Total | 
## -------------|-----------|-----------|-----------|-----------|
##     negative |      1253 |        85 |        23 |      1361 | 
##              |     0.675 |     0.136 |     0.052 |           | 
## -------------|-----------|-----------|-----------|-----------|
##      neutral |       343 |       408 |        57 |       808 | 
##              |     0.185 |     0.651 |     0.128 |           | 
## -------------|-----------|-----------|-----------|-----------|
##     positive |       260 |       134 |       365 |       759 | 
##              |     0.140 |     0.214 |     0.820 |           | 
## -------------|-----------|-----------|-----------|-----------|
## Column Total |      1856 |       627 |       445 |      2928 | 
##              |     0.634 |     0.214 |     0.152 |           | 
## -------------|-----------|-----------|-----------|-----------|
## 
## 
```

7.2 Polarity Over Time

```
# Actual Polarity over time
cc <- c("#E69F00", "#56B4E9", "#009E73", "#CC79A7", "#D55E00", "#D65E00")
tweet_polarity_date_actual <- tweets.test %>%
 count(sentiment, created) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(polarity = positive - negative,
    percent_positive = positive / (positive + negative) * 100)

polarity_over_time_actual <- tweet_polarity_date_actual %>%
  ggplot(aes(created, polarity)) +
  geom_col() +
  geom_smooth(method = "loess", se = FALSE) +
  geom_smooth(method = "lm", se = FALSE,aes(color = cc[1])) +
  theme_fivethirtyeight()+ theme(plot.title = element_text(size = 11)) +
  xlab(NULL) + ylab(NULL) +
  ggtitle("Polarity Over Time-Actual")

relative_polarity_over_time_actual <- tweet_polarity_date_actual %>%
  ggplot(aes(created, percent_positive )) +
  geom_col() +
  geom_smooth(method = "loess", se = FALSE) +
  geom_smooth(method = "lm", se = FALSE, aes(color = cc[1])) +
  theme_fivethirtyeight() + theme(plot.title = element_text(size = 11)) +
  xlab(NULL) + ylab(NULL) +
  ggtitle("Percent Positive Over Time-Actual")
```

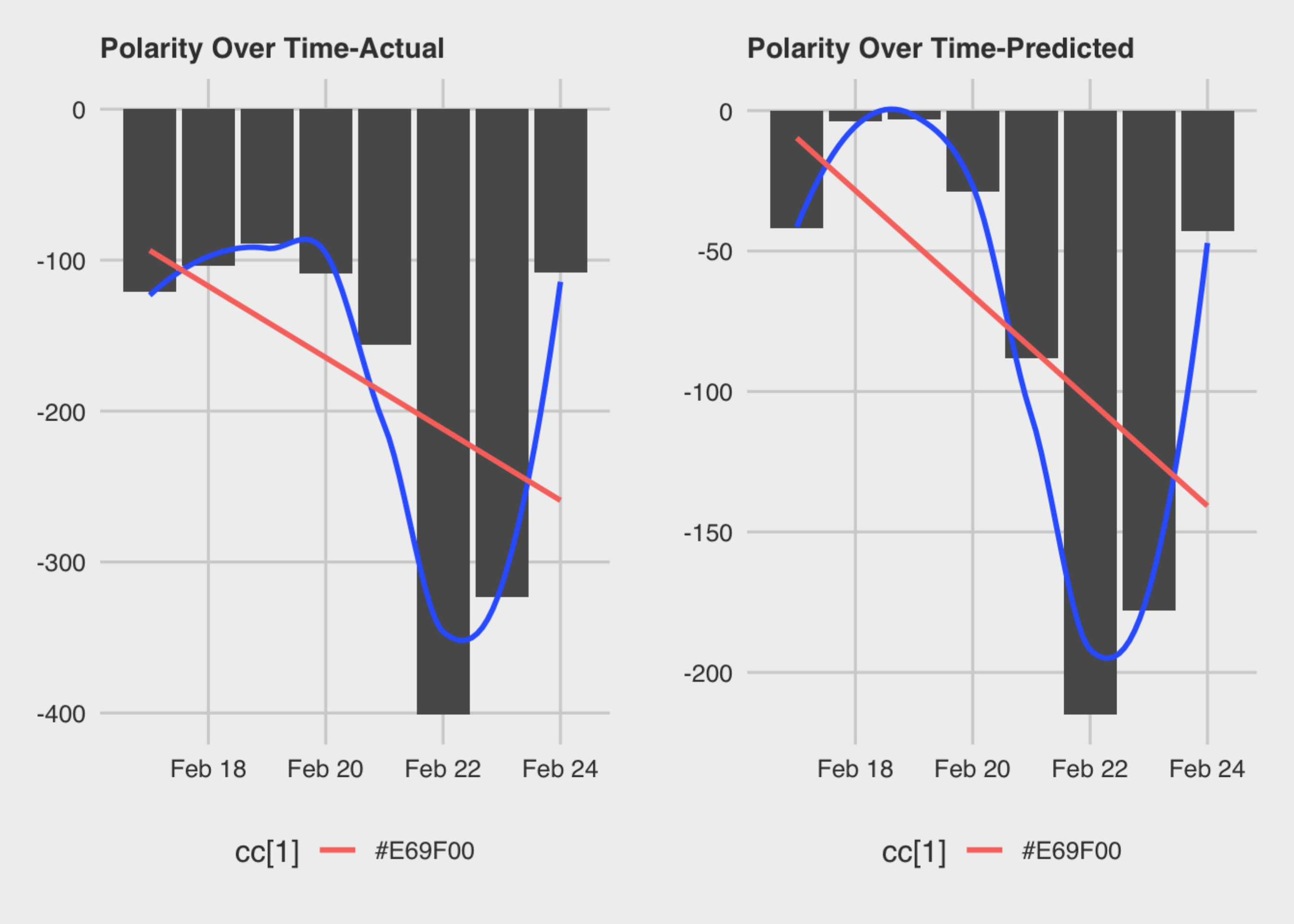
```
# Append the prediction to test
tweets.test$prediction<-pred
```

```
# Prediction Polarity Over time
tweet_polarity_date_predicted <- tweets.test %>%
 count(prediction, created) %>%
  spread(prediction, n, fill = 0) %>%
  mutate(polarity = positive - negative,
    percent_positive = positive / (positive + negative) * 100)

polarity_over_time_predicted <- tweet_polarity_date_predicted %>%
  ggplot(aes(created, polarity)) +
  geom_col() +
  geom_smooth(method = "loess", se = FALSE) +
  geom_smooth(method = "lm", se = FALSE,aes(color = cc[1])) +
  theme_fivethirtyeight()+ theme(plot.title = element_text(size = 11)) +
  xlab(NULL) + ylab(NULL) +
  ggtitle("Polarity Over Time-Predicted")

relative_polarity_over_time_predicted <- tweet_polarity_date_predicted %>%
  ggplot(aes(created, percent_positive )) +
  geom_col() +
  geom_smooth(method = "loess", se = FALSE) +
  geom_smooth(method = "lm", se = FALSE, aes(color = cc[1])) +
  theme_fivethirtyeight() + theme(plot.title = element_text(size = 11)) +
  xlab(NULL) + ylab(NULL) +
  ggtitle("Percent Positive Over Time-Predicted")
```

7.3 Polarity Visualizations

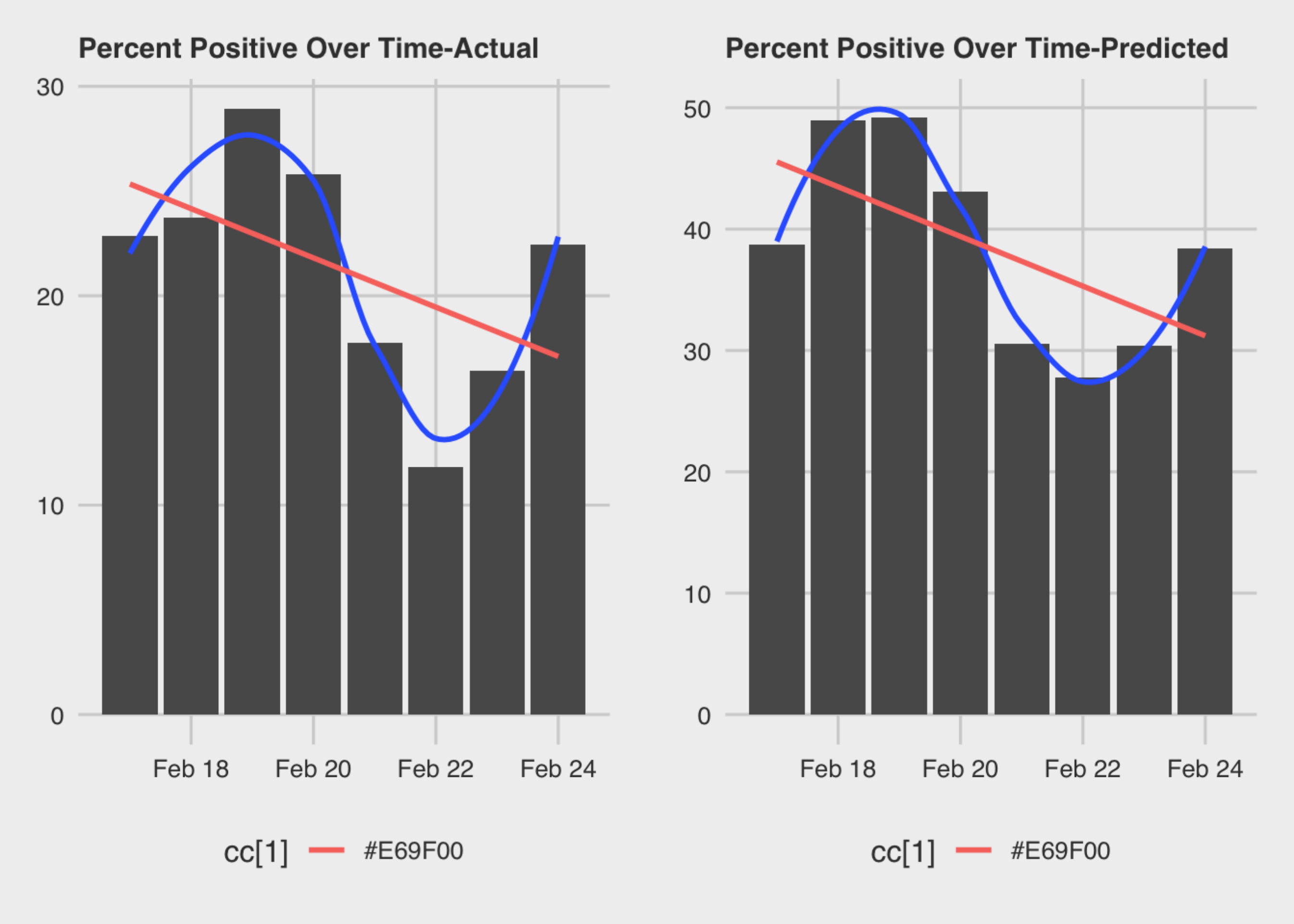
```
grid.arrange(polarity_over_time_actual,polarity_over_time_predicted , ncol = 2)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



```
grid.arrange(relative_polarity_over_time_actual, relative_polarity_over_time_predicted,
ncol = 2)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



The plots show polar sentiment of the tweets from 16-Feb-2015 to 24-Feb-2015. Polarity is Positive less Negative, Percent Positive is calculated by (Positive/Positive+Negative)*100").

In both plots, sentiment is mostly negative over time.

8.0 RDS Files for Shiny App

```
# Generate RDS Files for Shiny Application
saveRDS(cleanTweets, "cleanTweets.rds")
saveRDS(sentimentCount, "sentimentCount.rds")
saveRDS(americanSentimentCount, "americanSentimentCount.rds")
saveRDS(deltaSentimentCount, "deltaSentimentCount.rds")
saveRDS(southwestSentimentCount, "southwestSentimentCount.rds")
saveRDS(unitedSentimentCount, "unitedSentimentCount.rds")
saveRDS(usairwaysSentimentCount, "usairwaysSentimentCount.rds")
saveRDS(virginSentimentCount, "virginSentimentCount.rds")
saveRDS(wordfrequency, "wordfrequency.rds")
saveRDS(polarity_over_time_actual, "polarity_over_time_actual.rds")
saveRDS(tweet_polarity_date_actual, "tweet_polarity_date_actual.rds")
saveRDS(relative_polarity_over_time_actual, "relative_polarity_over_time_actual.rds")
saveRDS(tweet_polarity_date_actual, "tweet_polarity_date_actual.rds")
saveRDS(polarity_over_time_predicted, "polarity_over_time_predicted.rds")
saveRDS(tweet_polarity_date_predicted, "tweet_polarity_date_predicted.rds")
saveRDS(relative_polarity_over_time_predicted, "relative_polarity_over_time_predicted.rds")
saveRDS(tweet_polarity_date_predicted, "tweet_polarity_date_predicted.rds")
saveRDS(cc, "cc.rds")
saveRDS(uniquewords, "uniquewords.rds")
saveRDS(uniqueWordsPlot, "uniqueWordsPlot.rds")
saveRDS(tweets, "tweets.rds")
saveRDS(tweetsbySentimentreasonPlot, "tweetsbySentimentreasonPlot.rds")
```

9.0 Conclusions and Recommendations

It is evident that sentiment from tweets for the U.S. airlines is predominantly negative, due to customer service issues. Certain airlines receive more negative tweets in part not just because of issues passengers face, but also due to the size of their operations; planes they operate versus other carriers. Airlines can address the negative sentiment by proactively monitoring their twitter feeds from disgruntled passengers and working on solutions with them, before they loose the customer to other carriers down the line.

Further analysis of the twitter feed could help understand if the negative sentiment was just for this short period in time, or an ongoing issue. Increasing the sample size to months or years will help provide more insight. In addition, detailing sentiment over certain time periods like certain seasons or peak flying periods like summer or Christmas holidays can provide airlines better indications to properly prepare customer support staff prior on where to focus their efforts, and keep passengers satisfied and retain future business.

Additional information like local weather data and aircraft details, that airlines already have can also provide the model more insight on customer sentiment. Canceled flights due to local weather delays or mechanical failures on aging aircraft, can help pinpoint certain actions the airlines prior to the passenger receiving the bad news. Airlines can remove aircraft that results in more negative sentiment than other aircraft in the fleet, and Airlines can offer revised routes to passengers’ destinations prior to a flight being canceled.

Proactive monitoring of the customers sentiment overtime, will help provide a quick indicator if actions taken by the airlines are effective or would need further refinement.

10.0 Deployment

The underlying code of this markdown report, can be found on Github (https://github.com/xnazar/CSDA1040Assignment2) and on the Shiny web application About page as listed on shinyapps.io (https://jose-g.shinyapps.io/AirlineSentiment)

The Shiny app provides a quick visual overview of the customer sentiment from the analysis done on the tweets overall and on each airline.

11.0 Bibliography

Airlines for America (A4A). (2020). The Airline Industry. Retrieved July 01, 2020, from https://www.airlines.org/industry/ (https://www.airlines.org/industry/)

Alaska Airlines, Inc. (2016, April 04). Alaska Air Group to Acquire Virgin America, Creating West Coast's Premier Carrier. Retrieved July 02, 2020, from https://investor.alaskaair.com/news-releases/news-release-details/alaska-air-group-acquire-virgin-america-creating-west-coasts (https://investor.alaskaair.com/news-releases/news-release-details/alaska-air-group-acquire-virgin-america-creating-west-coasts)

Appen Limited. (2020). Confidence to Deploy AI with World-Class Training Data. Retrieved June 28, 2020, from https://appen.com/ (https://appen.com/)

Crowdflower. (2016, November 21). Airline Twitter Sentiment - dataset by crowdflower. Retrieved June 27, 2020, from https://data.world/crowdflower/airline-twitter-sentimentThe (https://data.world/crowdflower/airline-twitter-sentimentThe) Port Authority of New York and New Jersey. (2019). 2019 Airport Traffic Report. Retrieved July 02, 2020, from https://www.panynj.gov/content/dam/airports/statistics/statistics-general-info/annual-atr/ATR2019.pdf (https://www.panynj.gov/content/dam/airports/statistics/statistics-general-info/annual-atr/ATR2019.pdf)

Figure Eight. (2019, October 16). Twitter US Airline Sentiment. Retrieved June 28, 2020, from https://www.kaggle.com/crowdflower/twitter-airline-sentiment/data (https://www.kaggle.com/crowdflower/twitter-airline-sentiment/data)

Hamner, B. (2016). Benhamner/crowdflower-airline-twitter-sentiment. Retrieved June 28, 2020, from https://github.com/benhamner/crowdflower-airline-twitter-sentiment/blob/master/src/process.py (https://github.com/benhamner/crowdflower-airline-twitter-sentiment/blob/master/src/process.py)

IBISWorld. (2020). Industry Market Research, Reports, and Statistics. Retrieved July 01, 2020, from https://www.ibisworld.com/united-states/market-research-reports/domestic-airlines-industry/ (https://www.ibisworld.com/united-states/market-research-reports/domestic-airlines-industry/)

Isidore, C. (2013). US Airways-American Airlines to merge. Retrieved July 02, 2020, from https://money.cnn.com/2013/02/14/news/companies/us-airways-american-airlines-merger/index.html (https://money.cnn.com/2013/02/14/news/companies/us-airways-american-airlines-merger/index.html)

Jurafsky, D. (2019). Text Classification and Naïve Bayes. Retrieved July 01, 2020, from https://web.stanford.edu/~jurafsky/slp3/slides/7_NB.pdf (https://web.stanford.edu/~jurafsky/slp3/slides/7_NB.pdf)

Mazareanu, E. (2020, June 15). U.S. airline industry market share 2019. Retrieved July 01, 2020, from https://www.statista.com/statistics/250577/domestic-market-share-of-leading-us-airlines/ (https://www.statista.com/statistics/250577/domestic-market-share-of-leading-us-airlines/)

The Port Authority of New York and New Jersey. (2019). 2019 Airport Traffic Report. Retrieved July 02, 2020, from https://www.panynj.gov/content/dam/airports/statistics/statistics-general-info/annual-atr/ATR2019.pdf (https://www.panynj.gov/content/dam/airports/statistics/statistics-general-info/annual-atr/ATR2019.pdf)

United States Securities and Exchange Commission. (2013). Twitter, Inc - Initial Public Offering. Retrieved June 27, 2020, from https://www.sec.gov/Archives/edgar/data/1418091/000119312513390321/d564001ds1.htm (https://www.sec.gov/Archives/edgar/data/1418091/000119312513390321/d564001ds1.htm)