Final Project Report

Along with my tweet dataset (4554 observations) and the training data (complaint1700.csv+noncomplaint1700.csv), I also loaded in the downloaded Bing Liu's Opinion Lexicon to count the appearance of positive words and negative words in each tweet.

Firstly, I removed all the digits and special characters and stop words in the tweets and converted all the words to lowercase. Secondly, to better understand the tweets, I counted all the words from all the tweets and plot the top 15 words found in the tweets. From the plot we could see the top words were mostly about "airlines", "delay" and "time" in the complaint tweets. For the test data (my tweet dataset), I used the sentiment lexicon of "bing" to get the sentiment of each word from the tweets. Based on that, I plotted the top positive words and negative words.

To classify the tweets, I tried both machine learning methods and non-machine learning methods as comparison. I represented all the tweets as corpus, creating a matrix of terms and documents. I put the training matrix into the Naive Bayes model and then test my own data. I got 2110 positive predicted tweets, but the model seems fail to classify some of them.

Next I tried to remove some of the negative words to get a new prediction. I counted the positive words and negative words of each tweet, and if the positive count is larger than the negative count (here the threshold is pos.count > neg.count), the tweet would be classified into the noncomplaint tweet. After that, I got 157 non-complaint tweets. After checking the output tweets, I got the precision of 80.89%.

Code

```r
library(ggplot2)
library(tidytext)
library(stringr)
library(dplyr)
library(base)
library(tm)
library(RColorBrewer)
library(wordcloud)
library(e1071)
```

Load the data

```r
raw.data = read.csv("ExportedData_xuxinran.csv", header = TRUE)
complaint.data = read.csv("complaint1700.csv", header = TRUE)
noncomplaint.data = read.csv("noncomplaint1700.csv", header = TRUE)

neg.words = read.table("negative-words.txt")
pos.words = read.table("positive-words.txt")

data = raw.data
```

Data Cleaning

```r
#remove digits and special characters from tweets
complaint.data$tweet = str_replace_all(complaint.data$tweet, "[[:punct:]]", "")
complaint.data$tweet = str_replace_all(complaint.data$tweet, "[[:digit:]]+", "")
#convert to lower case
complaint.data$tweet = tolower(complaint.data$tweet)

#remove digits and special characters from tweets
noncomplaint.data$tweet = str_replace_all(noncomplaint.data$tweet, "[[:punct:]]", "")
noncomplaint.data$tweet = str_replace_all(noncomplaint.data$tweet, "[[:digit:]]+", "")
#convert to lower case
noncomplaint.data$tweet = tolower(noncomplaint.data$tweet)
```

```r
data$tweet = as.character(data$tweet)
data$X = as.character(data$X)
data$X.1 = as.character(data$X.1)
data$tweet = paste(data$tweet, data$X, data$X.1, sep = "")
data$X <- NULL
data$X.1 <- NULL

#remove digits and special characters from tweets
data$tweet = str_replace_all(data$tweet, "[[:punct:]]", "")
data$tweet = str_replace_all(data$tweet, "[[:digit:]]+", "")
#convert to lower case
data$tweet = tolower(data$tweet)
```

```r
#remove stop words
data("stop_words")
test_clean <- data %>%
  dplyr::select(tweet) %>%
  unnest_tokens(word, tweet) %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```r
complaint_clean <- complaint.data %>%
  dplyr::select(tweet) %>%
  unnest_tokens(word, tweet) %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

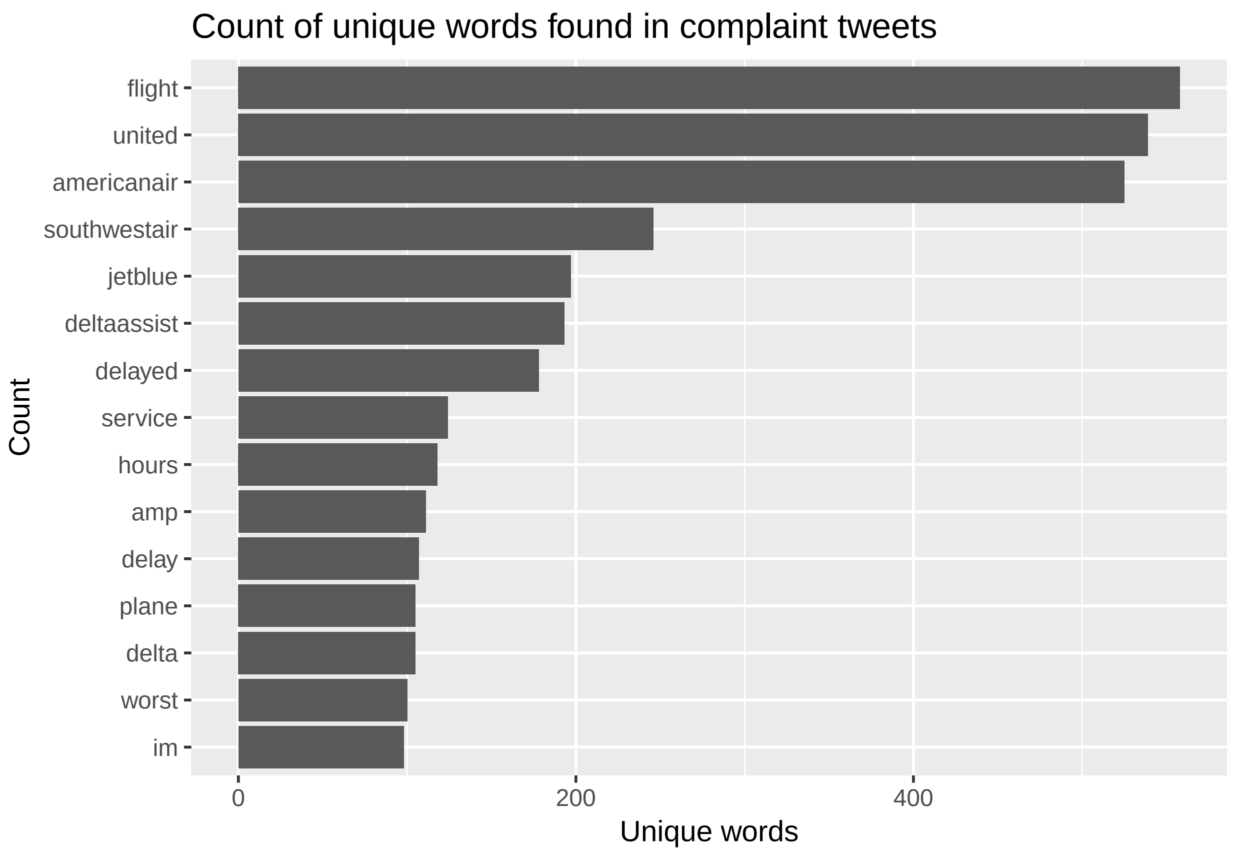
```r
noncomplaint_clean <- noncomplaint.data %>%
  dplyr::select(tweet) %>%
  unnest_tokens(word, tweet) %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

Visualization of the tweets

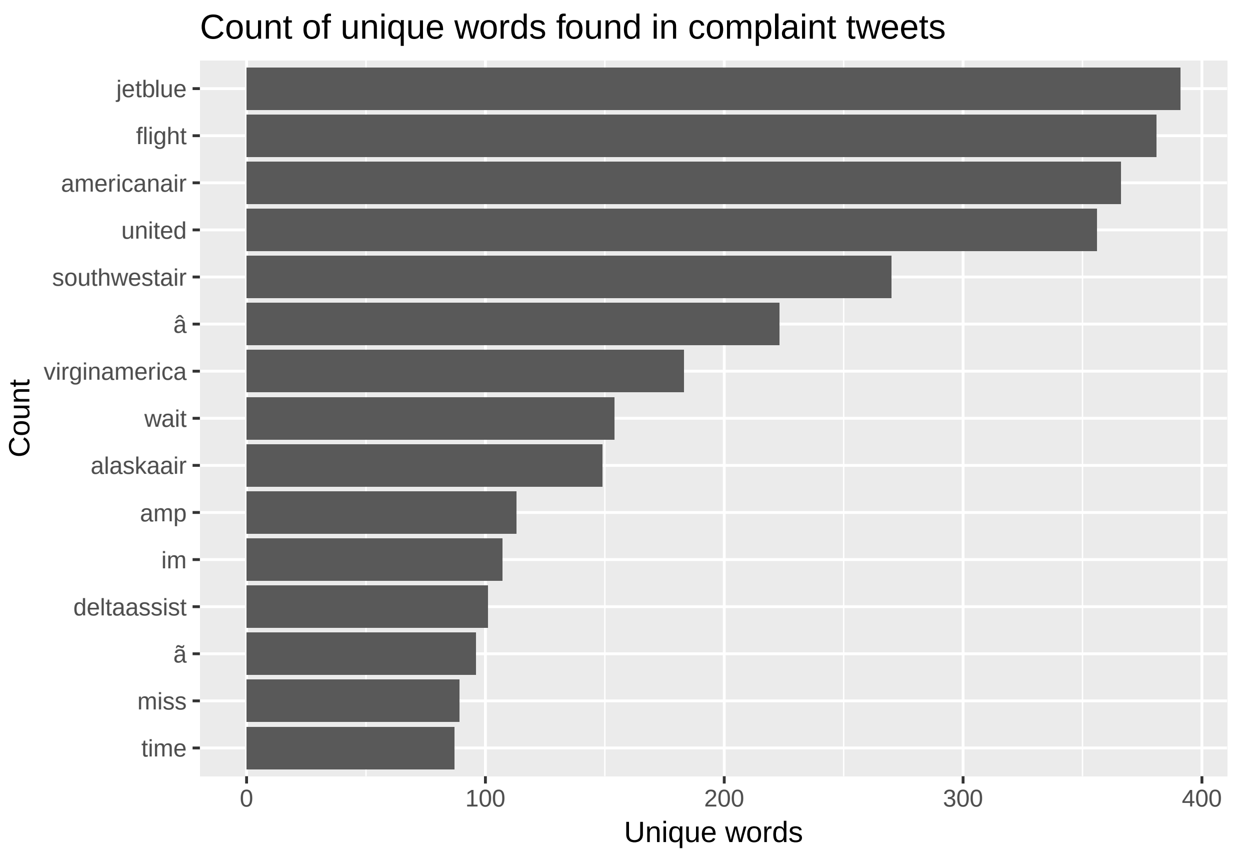
```r
# plot the top 15 words
complaint_clean %>%
  count(word, sort = TRUE) %>%
  top_n(15) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(x = word, y = n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip() +
  labs(x = "Count",
       y = "Unique words",
       title = "Count of unique words found in complaint tweets")
```

```
## Selecting by n
```



```
# plot the top 15 words in noncomplaint tweets
noncomplaint_clean %>%
  count(word, sort = TRUE) %>%
  top_n(15) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(x = word, y = n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip() +
  labs(x = "Count",
       y = "Unique words",
       title = "Count of unique words found in complaint tweets")
```

```
## Selecting by n
```

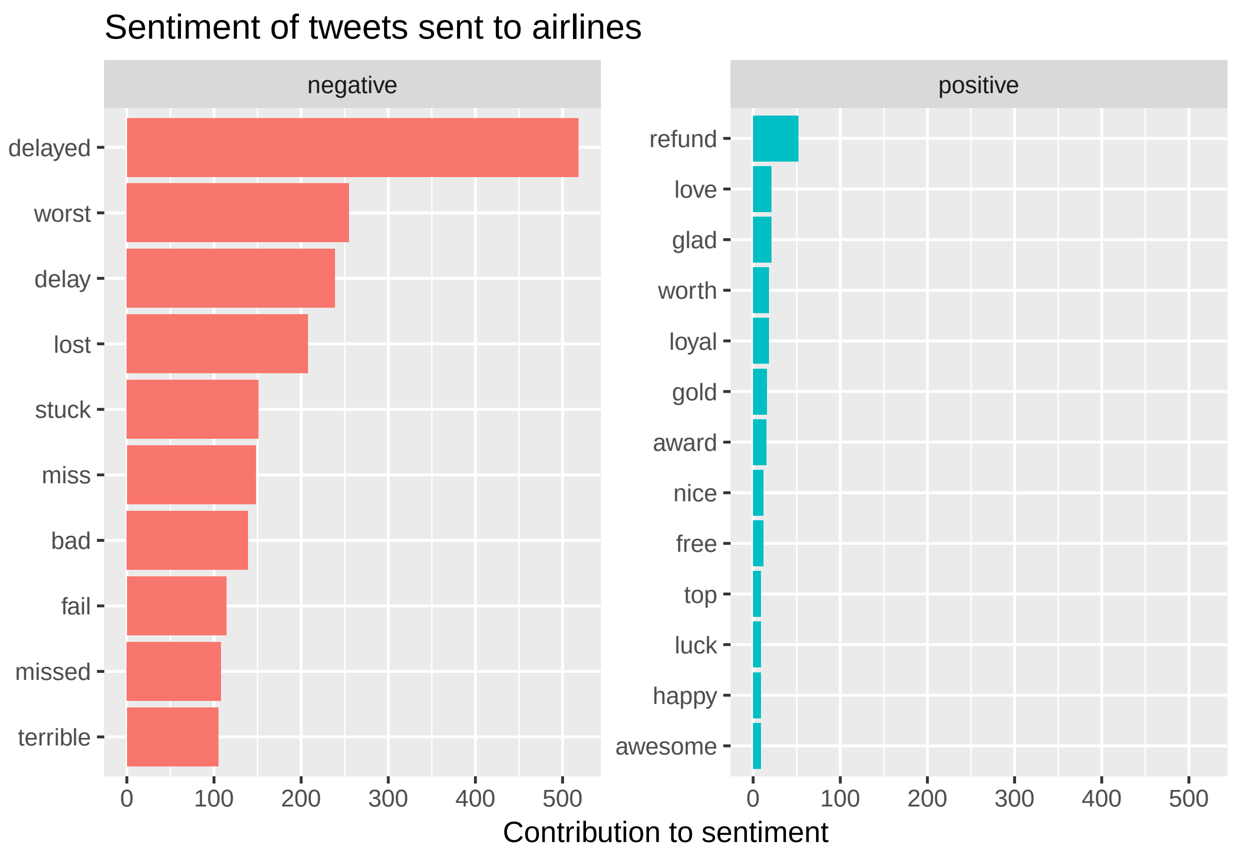


```
bing_word_counts <- test_clean %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

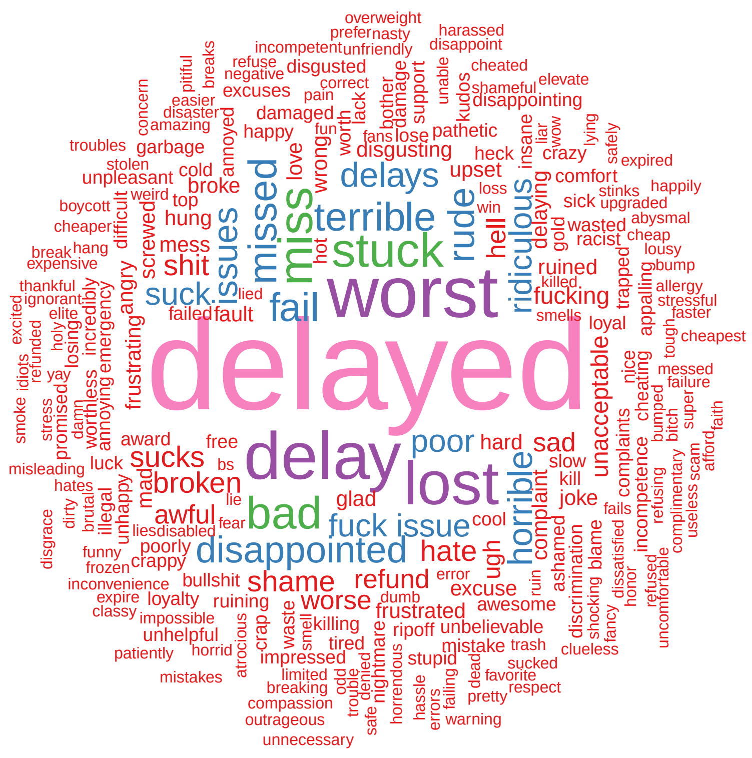
```
## Joining, by = "word"
```

```
bing_word_counts %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(title = "Sentiment of tweets sent to airlines",
       y = "Contribution to sentiment",
       x = NULL) +
  coord_flip()
```

```
## Selecting by n
```



```
df <- bing_word_counts[order(bing_word_counts$n, decreasing=TRUE),]
wordcloud(df$word, df$n, min.freq = 1, max.words=250, random.order=FALSE, rot.per=0.4, colors=brewer.pal
```



Classify the tweets

```
corpus <- Corpus(VectorSource(c(complaint.data$tweet, noncomplaint.data$tweet, data$tweet)))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeWords, stopwords("english"))
corpus <- tm_map(corpus, stemDocument)
tdm <- TermDocumentMatrix(corpus)
mat <- removeSparseTerms(tdm, sparse = 0.98 )
train <- t(as.matrix(mat))[1:3400,]
test <- t(as.matrix(mat))[3401:7954,]
train.label = c(rep(0, 1700), rep(1, 1700))
classifier = naiveBayes(train, as.factor(train.label))
predicted = predict(classifier, test)
data$predict = predicted
```

```
## These two functions are to calculate the positive words and negative words in the tweets.

count_pos = function(sentence){
  word.list = str_split(sentence, '\\s+')
    words = unlist(word.list)
    pos.matches = match(words, pos.words$V1)
    pos.count = sum(!is.na(pos.matches))
    return (pos.count)
}

count_neg = function(sentence){
  word.list = str_split(sentence, '\\s+')
    words = unlist(word.list)
    neg.matches = match(words, neg.words$V1)
    neg.count = sum(!is.na(neg.matches))
    return (neg.count)
}
```

```
data$pos.count = unlist(lapply(data$tweet, count_pos))
data$neg.count = unlist(lapply(data$tweet, count_neg))

non_negative = data[which(((data$pos.count>(data$neg.count+2))
                          |((data$neg.count==0))&(data$pos.count>0))
                          &(data$predict==1)),]
```

```
nrow(non_negative)
```

```
## [1] 157
```

```
head(non_negative$tweet,10)
```

```
##  [1] "united im waiting for your official confirmation about wheres the baggage file report correct
##  [2] "jetblue kellydollyrot that just screams fun so glad to be your ride jb ur supposed  respond be
##  [3] "jetblue i cant wait to find out if im getting flyfi on my thursday flight excited to be flying
##  [4] "forever in our hearts we will never forget you rest in peace fasource americanair united cabin
##  [5] "late post to thank virginamerica for letting me get my sfgiants fix in the air from ordsfo fly
##  [6] "cant wait to go to saint lucia on americanair next year february"
##  [7] "why does southwestair always say theyre inexpensive yet every time i look they never are"
##  [8] "because only  upcoming flight in my southwestair profile is never enough"
##  [9] "never thought id be this excited about a virginamerica dfw  dal flight but i am httptcoxbbilsz
## [10] "kicking off autismspeaks blue horizons for autism thank you jetblue httptcoupdpcsecp"
```

```r
result = merge(non_negative, raw.data, by="id", all.x = TRUE)
result = result[, which(names(result) %in% c("id", "tweet.y"))]
names(result) = c("id", "tweet")
# write.csv(result, file = "non_complaint.csv",row.names=FALSE)
```