Text Analysis

1.Summary

```
 In our data, there are 4555 tweets here. Here, we mainly perform sentiment analysis on these 4555 tweets
. We will extract negative words from 1,700 complaint tweets, positive words from 1,700 non-complaint twe
ets, and then use rule-based classification to model to obtain the sentiment analysis prediction results
of 4555 tweets data. Finally, the predicted sentiment analysis results are compared with the manually-sel
ected correct sentiment analysis results to calculate the accuracy of the model classification to evaluat
e the effect of the model.
 The rule-based classification can be used to refer to any classification scheme that make use of IF-THEN
rules for class prediction. So we first use Corpus () function to create a corpus, then remove the number
s, symbols, stop words, etc. in the corpus to get a meaningful corpus as much as possible, and finally fi
lter the emotions based on the high-frequency words in the corpus. Classification of rule words to predic
t sentiment of text.
```

2.The specific process is as follows:

```r
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.5.3
```

```r
library(tm)
```

```
## Warning: package 'tm' was built under R version 3.5.3
```

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

```
## Warning: package 'NLP' was built under R version 3.5.2
```

```r
library(wordcloud)
```

```
## Warning: package 'wordcloud' was built under R version 3.5.3
```

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

```
## Warning: package 'RColorBrewer' was built under R version 3.5.2
```

```
Find negative words from complaint1700.csv to form a thesaurus of negative words
```

```r
complaint_data<-read.csv("C:/Users/ibf/Desktop/complaint1700.csv",
                         header=TRUE, sep=',', quote='"',encoding = "UTF-8")
```

```
Use the Corpus() function to create a corpus called tweet_corpus from tweets
```

```r
tweet_corpus <- Corpus(VectorSource(complaint_data$tweet))
```

```
Create a control list that stores the option
```

```r
mystop<-"flight"
ctrl = list(tolower=T, removePunctuation=T, removeNumbers=T, stopwords=c(stopwords("english"),mystop)
            ,wordLengths=c(-Inf,20),bounds=list(global=c(5,Inf)))
```

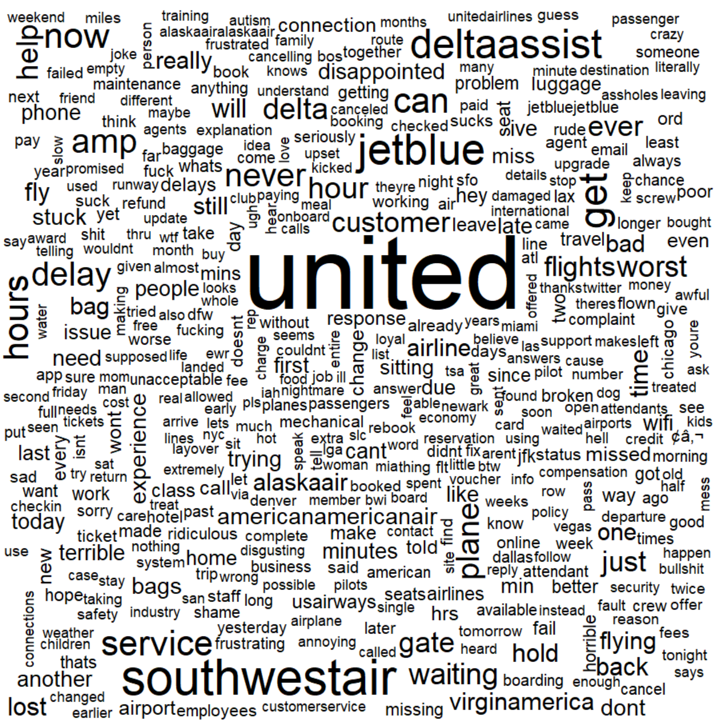
```
Add the argument control to TermDocumentMatrix() that references the control list you created
```

```
dtm1 <- TermDocumentMatrix(tweet_corpus, control=ctrl)
inspect(dtm1)
```

```
## <<TermDocumentMatrix (terms: 464, documents: 1035)>>
## Non-/sparse entries: 8206/472034
## Sparsity           : 98%
## Maximal term length: 19
## Weighting          : term frequency (tf)
## Sample             :
##               Docs
## Terms           133 135 327 328 344 417 475 893 982 983
##   americanair     1   0   1   0   1   1   2   4   2   1
##   can             1   0   4   0   2   1   1   2   0   1
##   delayed         1   4   0   0   0   0   1   1   0   1
##   deltaassist     0   1   0   0   0   0   0   0   0   1
##   get             2   0   3   1   0   1   2   0   2   1
##   jetblue         0   0   1   0   0   0   0   1   0   0
##   now             0   1   0   1   0   1   1   3   2   1
##   service         0   1   1   1   0   0   3   2   1   1
##   southwestair    0   1   0   0   0   1   0   0   0   2
##   united          0   1   2   4   2   1   0   3   2   1
```

```
Here we create a word cloud diagram to view the high-frequency words more intuitively.
```

```
tweet_mat<- as.matrix(dtm1)
set.seed(1234) # for reproducibility
words <- sort(rowSums(tweet_mat),decreasing=TRUE)
df_neg <- data.frame(word = names(words),freq=words)
wordcloud(words = df_neg$word, freq = df_neg$freq)
```



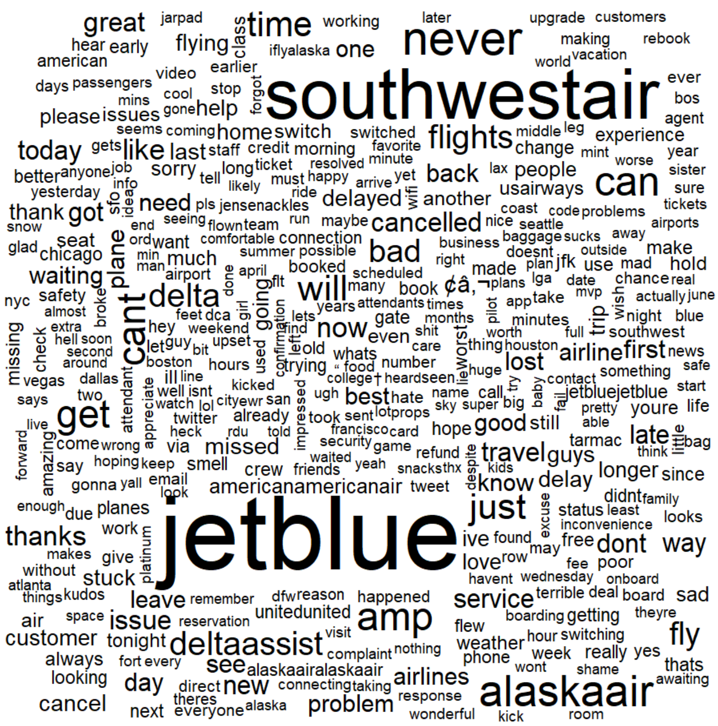

```
Do the same as 1,700 complaint tweets to extract positive word for noncomplaint1700 , as follows:
```

```
noncomplaint_data<-read.csv("C:/Users/ibf/Desktop/noncomplaint1700.csv",
                            header=TRUE, sep=',', quote='"',encoding = "UTF-8")
non_tweet_corpus <- Corpus(VectorSource(noncomplaint_data$tweet))
mystop<-"flight"
ctrl = list(tolower=T, removePunctuation=T, removeNumbers=T, stopwords=c(stopwords("english"),mystop)
            ,wordLengths=c(-Inf,20),bounds=list(global=c(5,Inf)))
dtm2 <- TermDocumentMatrix(non_tweet_corpus, control=ctrl)

inspect(dtm2)
```

```
## <<TermDocumentMatrix (terms: 420, documents: 964)>>
## Non-/sparse entries: 6806/398074
## Sparsity           : 98%
## Maximal term length: 19
## Weighting          : term frequency (tf)
## Sample             :
##                  Docs
## Terms              357 475 497 602 766 853 857 859 864 91
##   alaskaair          0   1   0   0   0   1   1   1   2  2
##   americanair        1   2   0   2   3   2   2   2   0  1
##   amp                2   1   1   0   2   1   1   0   2  1
##   can                2   1   0   0   2   3   1   1   0  0
##   jetblue            0   2   0   1   0   1   7   0   1  1
##   never              2   2   1   0   0   0   1   0   0  3
##   southwestair       0   1   0   2   0   1   1   0   2  1
##   united             0   1   1   1   0   1   4   2   4  1
##   virginamerica      2   3   1   2   0   0   1   0   0  5
##   wait               1   1   0   2   0   2   3   0   0  1
```

```
#Create word cloud diagram
non_tweet_mat<- as.matrix(dtm2)
set.seed(1234) # for reproducibility
words <- sort(rowSums(non_tweet_mat),decreasing=TRUE)
df_pos <- data.frame(word = names(words),freq=words)
wordcloud(words = df_pos$word, freq = df_pos$freq)
```




```
Now, we delete the overlapping words in 1700 complaint tweets and 1700 noncomplaint tweets to ensure the accuracy of positive and negative words to the greatest extent.
```

```
pos<-as.character(df_pos$word)
neg<-as.character(df_neg$word)
pos_words<-pos[!(neg %in% pos)]
pos_words<-na.omit(pos_words)
pos_words<-as.character(pos_words)
neg_words<-neg[!(pos %in% neg)]
```

```
Now use Rule-based Classification to model our data and analyze emotions.
```

```r
data<-read.csv("C:/Users/ibf/Desktop/temp.csv",
               header=TRUE,sep=',', quote='"',encoding = "UTF-8")
# Rule-based Classification
pos2 <- rep(0, nrow(data))
neg2<- rep(0, nrow(data))
#Select the words for sentiment analysis
pos2 <- grepl("best|missed|gate|yes|big|happy|wonderful|impressed|favorite|pretty", data$tweet, ignore.ca
se=TRUE,  perl=TRUE)
neg2 <- grepl("terrible|broken|unacceptable|failed|wrong|update|worse|seriously", data$tweet, ignore.case
=TRUE, perl=TRUE)

#Pick the corresponding id and tweets
pos_data<-data.frame(data$id[pos2],data$tweet[pos2])
names(pos_data)<-c("id","tweets")
```

3.Evaluation model

```
According to the non-complaint text extracted by the model, we perform manual screening, and the true cla
ss obtained is as follows:
```

```r
#positive class=1,negtive class=0
true<-c(0,0,1,0,0,0,0,1,1,1,0,1,1,1,0,1,0,1,0,0,1,1,0,1,0,0,0,1,0,1,1,0,0,0,0,0,0,0,1,0,1,1,0,0,0,1,0,0,0
,0,1,0,0,1,1,1,0,0,0,1,0,0,0,0,1,1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,0,0,1,0,1,1,0,0,0,1,1,0,1,1,0,0,1,0,1,0,
1,1,1,1,1,1,1,0,0,1,1,1,1,0,0,1,1,1,0,0,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,1,1,1,0,0,0,1,1,0,0,0,0,1,1,1,1,1,0
,0,1,1,0,0,0,0,1,0,1,0,0,1,0,0,1,1,1,1,1,1,1,0,0,1,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,1,1,1,0,1,0,1,
0,1,0,1,0,0,0,1,0,0,0,1,0,1,1,1,0,1,1,1,0,1,1,1,0,1,1,1,0,0,0,1,1,0,1,0,1,0,1,0,1,0,1,1,1,0,1,0,0,0,1,1,1
,1,1,0,0,0,1,1,0,1,1,0,1,0,1,0,0,1,1,1,1,1,0,1,0,0,1,0,0,0,1,0,0,0,1,0,0,1,1,0,1,1,0,1,1,0,1,0,0,1,1,0,1,
0,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,1,1,0,1,1,1,0,0,0,1,1,0,0,1,1,1,1,1,0,0,1,1,1,1,1,0,0,1,0,0,1,0,0,1,1,0,1
,0,1,0,0,0,0,0,1,0,0,0,1,1,0,1,1,1,0,1,0,0,1,0,1,0,0,0,1,0,0,1,1,0,1,0,1,0,1,1,1,0,0,0,1,1,1,1,1,0,1,0,0,
0,0,0,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,1,0,0,1,1,0,1,0,1,1,1,0,0,0,1,0,0,1,1,1,0,1,0,0,1,1,1,0,0,1,1,1,1,1,0
,0,1,0,0,1,1,0,1,1,0,0,1,1,1,0,1,1,1,0,0,0,1,0,0,1,1,0)
```

```
Next we create the following method to evaluate the model results.
```

```r
Accuracy =  sum(true)/length(true)
Accuracy
```

```
## [1] 0.4899194
```

```
We can see that the accuracy of the model is 49.0%, and the model effect is ok, but there is still room f
or improvement. It can be improved by expanding the content of the corpus and adding more rule words.
```