Final Project

Guchuan Qiu

Simon Business School

Data introduction

```
library(data.table)
library(tm)
library(wordcloud)
library(SnowballC)
library(stringr)
library(e1071)
library(randomForest)
library(dplyr)
df<-fread("temp1.csv")
n_df<-fread("complaint1700.csv")
p_df<-fread("noncomplaint1700.csv")
nrow(df)     #View the number of tweets to be detected
## [1] 4555
nrow(p_df)  #See the number of positive tweets
## [1] 1700
nrow(n_df)   #See the number of negative tweets
## [1] 1700
```

The dataset that I extracted has 4,555 tweets. In addition, I used 1700 complaint tweets and 1700 non-complaint tweets to build models to determine the class of 4,555 tweets. The text of all tweets were converted to a corpus. The corpus was used to build a term-document matrix (tdm). During the construction, all punctuation, numbers, extra spaces and English stop words were removed, and the resulting matrix contains only terms with a sparsity factor of less than 0.99. I converted term-document matrix to document-term matrix.

Data Exploration

Here, I made a word cloud diagram of the three data sets. From the diagram, I found that the part of speech is similar.

```
tweet_corpus<-VCorpus(VectorSource(df$tweet))  #build a corpus
tweet_tdm<-TermDocumentMatrix(tweet_corpus,control =
list(tolower=T,removeNumbers=T,stopwords=T,removePunctuation=T,stemming=T))

#build TDM
wc_tdm<-slam::rollup(tweet_tdm,2,na.rm=T,FUN=sum)    #count
wc_mm<-as.matrix(wc_tdm)                             #convert to matrix
wc_freq<-sort(rowSums(wc_mm),decreasing = T)         #sort
wc_df<-data.frame(words=names(wc_freq),wc_freq)      #data.frame
wordcloud(head(wc_df$words,100),head(wc_df$wc_freq,100),random.order = F,colors = brewer.pal(8,"Dark2"))
```

The tweet to be tested

```
tweet_corpus<-VCorpus(VectorSource(n_df$tweet))
tweet_tdm<-TermDocumentMatrix(tweet_corpus,control =
list(tolower=F,removeNumbers=T,stopwords=T,removePunctuation=T,stemming=T))
wc_tdm<-slam::rollup(tweet_tdm,2,na.rm=T,FUN=sum)
wc_mm<-as.matrix(wc_tdm)
wc_freq<-sort(rowSums(wc_mm),decreasing = T)
wc_df<-data.frame(words=names(wc_freq),wc_freq)
wordcloud(head(wc_df$words,100),head(wc_df$wc_freq,100),random.order = F,colors = brewer.pal(8,"Dark2"))
```

The tweet of Noncomplaince

```
tweet_corpus<-VCorpus(VectorSource(p_df$tweet))
tweet_tdm<-TermDocumentMatrix(tweet_corpus,control =
list(tolower=F,removeNumbers=T,stopwords=T,removePunctuation=T,stemming=T))
wc_tdm<-slam::rollup(tweet_tdm,2,na.rm=T,FUN=sum)
wc_mm<-as.matrix(wc_tdm)
wc_freq<-sort(rowSums(wc_mm),decreasing = T)
wc_df<-data.frame(words=names(wc_freq),wc_freq)
wordcloud(head(wc_df$words,100),head(wc_df$wc_freq,100),random.order = F,colors = brewer.pal(8,"Dark2"))
```

The tweet of complaince

Model Selection

Here, I chose Naive Bayesian method, support vector machine and random forest model for prediction. Afterwards, I randomly divided the clearly labeled tweets into 2 parts to filter the model.

```
tweets<-c(p_df$tweet,n_df$tweet,df$tweet)        #merge Twitter and build a common
y=c(rep("positive",times=nrow(p_df)),rep("negative",times=nrow(n_df)))   #output
y<-factor(y)
docs <- Corpus(VectorSource(tweets))   #build a corpus
dtm.control = list(tolower=F, removePunctuation=T, removeNumbers=T, stopwords=T, stripWhitespace=T)
dtm.full <- DocumentTermMatrix(docs, control=dtm.control)   #dtm
dtm <- removeSparseTerms(dtm.full,0.99)  #Remove low frequency words
dtm_mm <- as.matrix(dtm)                 #Building a modeling and detection data set
x<-dtm_mm[1:(nrow(n_df)+nrow(p_df)),]
set.seed(1234)
num<-sample(c(0,1),size = length(y),replace = T,prob = c(0.8,0.2))
```

Naive Bayesion

```
nb.model <- naiveBayes( x[num==0,], y[num==0] )    #SVM with sigmoid kernel
pred.class <- predict( nb.model, x[num==1,] )
mean( pred.class== y[num==1] )     # get the accuracy
## [1] 0.6622222
```

Support Vector Machine

```
svm.model <- svm(y[num==0] ~ ., data = x[num==0,],kernel="sigmoid")  #SVM with sigmoid kernel
pred.class <- predict( svm.model, x[num==1,] )
mean(pred.class== y[num==1])    #get the accuracy
## [1] 0.677037
```

randomforest

```
rf_df<-data.frame(y,x)         #data.frame for randomforest
rf.model<-randomForest(y ~ ., data = rf_df[num==0,])      #randomforest
pred<-predict(rf.model,newdata = rf_df[num==1,])
mean( pred== y[num==1] )   #get the accuracy
## [1] 0.6622222
```

It could be found from the results that the prediction accuracy of the SVM model is the highest. So, I used it for prediction. To improve accuracy, I used all labeled tweets for modeling.

```
x_nn <- dtm_mm[(nrow(n_df)+nrow(p_df)+1):length(tweets),]   #tweets to be detected
svm.model <- svm(y ~ ., data = x,probability = TRUE)        #svm model
pred.class <- predict( svm.model, x_nn ,probability = T)    #predict
pred = attr(pred.class, "probabilities")                    #get prediction probability
df$pred<-pred[,2] > 0.80                                    #use a probability of 0.8 for judgment
write.csv(df,file = "df.csv")                               #export
```

After exporting the data to R, I made subjective judgments on the prediction results of the model and amended the results. Next I create the following method to evaluate the model results.

```
ddf<-fread("ddf.csv")                    #Read the file containing non-negative tweets
Accuracy=sum(ddf$pred==1)/sum(df$pred==1)                   #Model's accuracy
Accuracy
## [1] 0.4625
```

Overall, the model I built has the accuracy of 46%. Among 526 test records, there are 242 being correct (marked as '1') based on my own evalutions. It means that among the predicted positive tweets, 46% of them are true positive. This shows that the model has a lot of room for improvement. In my opinion, one of the reason is that the trainging data is not enough, giving 4555 test data. The prediction accuracy could be further improved by expanding the training sample size or using other deep-learning methods.