Social Media Analysis Final Project: Sentimental Analysis

Submitted by Qianyu Dong (MSMA)

1. Approach description

To classify the 4,555 tweets about airline, there are basically two type of approaches: a) machine learning approach: split some data into train set and validation set, use train data to build DTM for each tweet and train classification models, then use validation data to choose the best model, which is finally used to make prediction of the 4,555 tweets; b) rule-based approach: import negative and positive term dictionaries, compute sentimental scores for each sentence, and classify tweets according to sentimental scores.

I tried both approaches, but find out that the machine learning approach did poor with the training data we had. This is probably because the training data we have is limited, including only 1,700 each for complaint and non-complaint sample tweets, which are not enough to get a accurate model. I also made word clouds of the top 20 words respectively in complaint and non-complaint sample tweets, and the result turned out to be highly alike:

Top 20 words in complaint1700.csv

Top 20 words in noncomplaint1700.csv

As the word clouds above show, “wait(ing)”, “cancelled”, “ever”, “amp”, “time”, “just” and “never” can be seen in both complaint and non-complaint set, which means that due to the similarity in features of the two prediction classes, machine

learning classification will be quite difficult.

So, I used rule-based classification to tackle the problem. The key for this approach is dictionaries. I imported Hu & Liu's opinion lexicon (https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html) as basic dictionaries: positive term dictionary and negative term dictionary. Each tweet gets a positive score and a negative score computed by checking whether the tweet contains words in positive/negative dictionary. The more positive/negative words a tweet contains, a higher positive/negative score it will gain. The sentimental score is determined by subtracting negative score from positive score. Tweets with sentimental score > 0 is chosen as final output of non-negative tweets.

To further increase the accuracy of dictionaries for this specific case about airlines, and I modified the positive and negative dictionaries by adding some words like "delay", "cancel" and "hold", which appear frequently in my first version of positive prediction with a sentimental score of 1 but should actually be predicted as negative, and got 182 positive prediction ultimately.

The precision of my classification is 126/182 = 69%.

2. R Codes

```
rm(list=ls())

library(readr)


#setwd("C:/Users/lenovo/Desktop/Final Project")


# generating word clouds for top 20 words in training sets

docs_neg <-Corpus(DirSource( "neg" ))

mystop = c("", "",'flight','airline','united','americanair', 'americanamericanair', 'unitedunited', 'get','now','southwestair','jetblue','jetbluejetblue', 'can','plane','delta','flights','please','virginamerica','alaskaair','southwestsouthwestair')

dtm_neg <- DocumentTermMatrix(docs_neg, control=list(tolower=T, removePunctuation=T, removeNumbers=T, stripWhitespace=T, stopwords=c(mystop, stopwords("english"), stopwords("spanish"))))

freq_neg <- colSums(as.matrix(dtm_neg))

wordcloud(names(freq_neg), freq_neg, max.words=20, colors=brewer.pal(6,"Dark2"))


docs_pos <-Corpus(DirSource( "pos" ))

dtm_pos <- DocumentTermMatrix(docs_pos, control=list(tolower=T, removePunctuation=T, removeNumbers=T, stripWhitespace=T, stopwords=c(mystop, stopwords("english"), stopwords("spanish"))))

freq_pos <- colSums(as.matrix(dtm_pos))

wordcloud(names(freq_pos), freq_pos, max.words=20, colors=brewer.pal(6,"Dark2"))


# import 4,555 lines of test data

test <- read_csv('testset_2.csv')

test <- test[,1:3]

sentences <- test$tweet


# introduce dictionaries

pos = scan("positive-words.txt", what = 'character', comment.char = ';')

neg = scan("negative-words.txt", what = 'character', comment.char = ';')


# function to compute sentiment score for each tweet
```

```
score.sentiment = function(sentences, pos, neg, .progress='none')
{
   require(plyr)
   require(stringr)
   scores = laply(sentences, function(sentence, pos, neg) {
      # clean up sentences
      sentence = gsub('[[:punct:]]', '', sentence) # remove punctuations
      sentence = gsub('[[:cntrl:]]', '', sentence) # remove control characters
      sentence = gsub('\\d+', '', sentence) # remove digits
      sentence = tolower(sentence)
      # split sentence into words
      word.list = str_split(sentence, '\\s+')
      words = unlist(word.list)
      # compare words to pos/neg dictionaries
      pos.matches = match(words, pos)
      neg.matches = match(words, neg)
      pos.matches = !is.na(pos.matches)
      neg.matches = !is.na(neg.matches)
      # compute score
      score = sum(pos.matches) - sum(neg.matches)
      return(score)
   }, pos, neg, .progress=.progress )
   scores.df = data.frame(score=scores, text=sentences)
   return(scores.df)
}

res <- score.sentiment(sentences, pos, neg)
pred_pos <- data.frame(res[res$score > 0, ])
pred_full <- merge(test, pred_pos, by.x = 'tweet', by.y = 'text')
pred_full <- pred_full[,c(2,1)]
colnames(pred_full) <- c('id', 'tweet')
write.csv(pred_full, 'Qianyu_Dong.csv', row.names = FALSE)
```