Project Report

Sentiment Analysis

R file and data set is also shared.

- Preparing the Data
- Building a Classifier
- Classification
- Accuracy

Summary:

Separately downloaded a set of negative [1] and positive [2] tweets from the links below.

The data was prepared using “tm” and “snowballC” library. After that “CaTools library was used to split the dataset. Then a train and test sets were generated. “randomForest” library was used for the classifier and then used “predict” function on the test set and finally generate a table to check the accuracy.

After that 11 random tweets were taken from the internet and manually classified them as positive and negative. 0 suggests that a tweet is negative and 1 suggests that a tweet is positive. An empty dataset was created from the “dataset” that had every word as column and then the new tweets were adjusted in the empty dataset after being prepared for it using different R functions and libraries that includes “dplyr” and “tidytext”. Finally, the new dataset was used with the classifier in the predict function. And then accuracy table was generated to check accuracy of the table. The model predicted all the negative tweets correctly however, just predicted 1 out of 6 positive tweets correctly.

Lastly, to improve the performance once I changed the value of “ntree” and it improved the performance. The other way can be to increase the size of the dataset to generate the classifier. Even changing the value at “removesparseterms” function can help in improving the accuracy.

The code below shares the steps of all the process:

```
# SENTIMENT ANALYSIS

library(readxl)
Sentiments <- read_excel("D:/MASON/1 Semester/AIT-580/Assignment/NLP/A.xls")
View(Sentiments)

install.packages("tm")
library(tm)

corpus <- VCorpus(VectorSource((Sentiments$Tweet)))
```

```
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus,removeNumbers)
corpus <- tm_map(corpus,removePunctuation)

library("SnowballC")
corpus <- tm_map(corpus,removeWords,stopwords())
corpus <- tm_map(corpus,stemDocument)
corpus <- tm_map(corpus,stripWhitespace)

dtm <- DocumentTermMatrix(corpus)

#Removing less repeated words
dtm <- removeSparseTerms(dtm, 0.999)
dataset <- as.data.frame(as.matrix(dtm))
dataset$Rating <- Sentiments$Rating
dataset$Rating <- factor(dataset$Rating, levels = c(0,1))

library(caTools)
set.seed(123)
split <- sample.split(dataset$Rating, SplitRatio = 0.75)
train <- subset(dataset, split == TRUE)
test <- subset(dataset, split == FALSE)

library(randomForest)
classifier <- randomForest(x=train[-943], y = train$Rating, ntree = 10)
y_pred <- predict(classifier, newdata = test[-943])

cm <- table(test[,943], y_pred)
```

Accuracy

```
  y_pred
     0   1
0 296  53
1  67  87
```

Using the above code classifier was developed using “randomForest”. The train and test sets were created to check the performance of the model. The above table shows the accuracy of the model.

Select 10 random tweets from Twitter:
Classify the tweets as 'pos' or 'neg’
Discuss the accuracy of the classification (i.e., is the classification "correct"?)

How can the accuracy be improved?

Ten tweets were taken randomly from the internet and were prepared for the classifier using the following code:

The number columns must be same for both train set and the test.

```
# Generating an empty table to input 10 tweets
tweettest <- dataset[dataset$Rating==2,]

TenTweets <- read_excel("D:/MASON/1 Semester/AIT-580/Assignment/NLP/TenTweets.xls")
TenTweets$id <- seq.int(nrow(TenTweets))

View(TenTweets)

library(dplyr)
library(tidytext)

tab <- data_frame(TenTweets$id, TenTweets$Tweet)
colnames(tab) <- c("id", "tweet")

tab <- tab %>% unnest_tokens(word, tweet)
tab <- tab %>% anti_join(stop_words)

tab$word <- stemDocument(tab$word)

for(i in 1:length(tab$id))
{
 x <- tab$id[i]
 for(j in 1:942)
 {
  if(tab$word[i] == colnames(tweettest[j]))
  {
  tweettest[x,j] <- c(1)
  }
  else
    tweettest[x,j] <- c(0)
 }
}

tweettest$Rating <- TenTweets$Rating

pred <- predict(classifier, newdata = tweettest[-943])
```

```
cm1 <- table(tweettest[,943], pred)
```

Accuracy

```
   pred
    0 1
  0 5 0
  1 5 1
```

According to above table the model predicted 6 out 11 tweets correctly.

The table below show accuracy of the model tested through test set:

```
 y_pred
      0   1
  0 296  53
  1  67  87
```

383 out of 503 were predicted correctly here which is fairly a good performance.

In terms of improving the accuracy one way is to alter the “ntree” number. After changing ntree from 10 to 20 following accuracy was observed.

```
y_pred
      0   1
  0 299  50
  1 62   92
```

391 out of 503 were predicted correctly.

Another way to improve the accuracy can be to add more data to generate a classifier. We can even change the value in the function “removesparseterms.”

Link:

[1]. https://github.com/lesley2958/twilio-sent-analysis/blob/master/pos_tweets.txt

[2]. https://github.com/lesley2958/twilio-sent-analysis/blob/master/neg_tweets.txt