A Project Report

on

Airline Sentiment Analysis using Text Mining

Submitted by

Group – 1

Sl.No	Student Name	NUID
1.	Smit Kumar Doshi	001475186
2.	Nishith Burman	001896660

Under the Guidance of

Prof. Xuemin jin

Associate Teaching Professor,

Mechanical and Industrial Engineering

Acknowledgement

We would like to express our sincere gratitude to our project advisor **Prof. Xuemin Jin** for the continuous support of our MS case study and related research, for his patience, motivation, and immense knowledge.

His guidance helped us in all the time of research and writing of this the case study. We could not have imagined having a better professor and mentor for our case study project.

We would also like to our teaching assistant Melike Hazal Can, who provided us the support and guidance throughout the semester

Abstract

In this study the application of data mining in an twitter database is presented, aiming the classification of the Airlines reviews as positive, negative and neutral. The text classification task is performed using different machine learning algorithm such as “Naïve Bayes” and “Support Vector Mechanism” technique. The results show that the most realistic model is “Support Vector Mechanism”, with a success rate of around 80% in classification, and that it is best method for sentiment classification

Table of Content

Introduction to Text Mining

Text mining is a method for drawing out content based on meaning and context from a large body (or bodies) of text. Or it is a method for gathering structured information from unstructured text. Text Mining has very wide application and is especially useful in the Health science field as many valuable information is still stored in the medical prescriptions of doctors. Text Mining uses algorithms of Data Mining, NLP, Statistics and Machine Learning to infer useful information from unstructured texts.

Application of Text Mining

1.) Text categorization into specific domains for example spam - non spam emails or for detecting sexually explicit content.

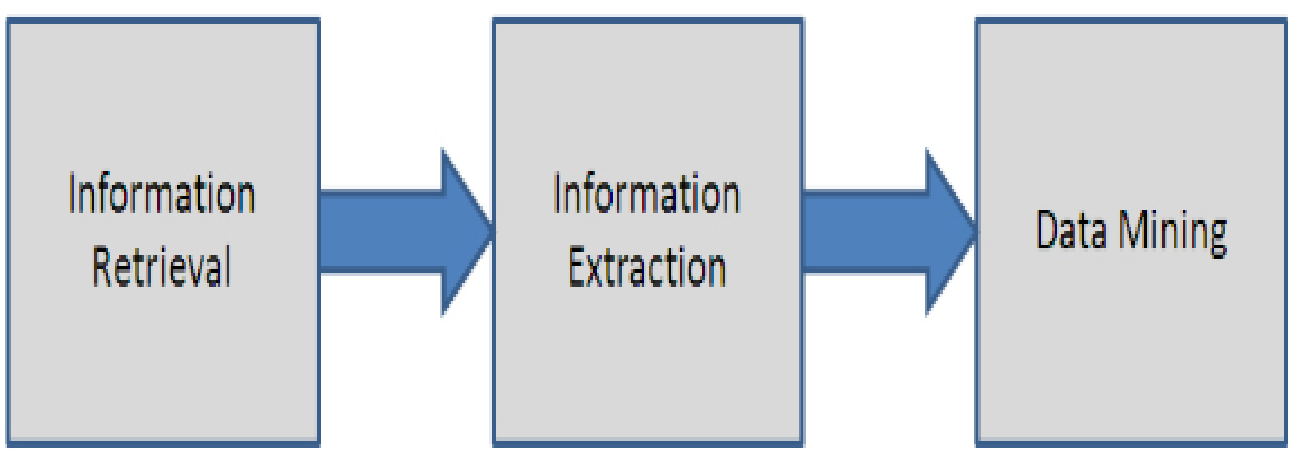
2.) Text clustering to automatically organize a set of documents.

3.) Sentiment analysis to identify and extract subjective information in documents. Detect what your customers are saying about your company when they use social media

4.) Concept/entity extraction that is capable of identifying people, places, organizations, and other entities from documents.

5.) Document summarization to automatically provide the most important points in the original document. This is particularly good for news summary

Process of Text Mining



Literature Review

Sentiment Analysis

Sentiment Analysis also known as *Opinion Mining* is a field within NLP that builds systems that try to identify and extract opinions within text. Usually, besides identifying the opinion, these systems extract attributes of the expression e.g.:

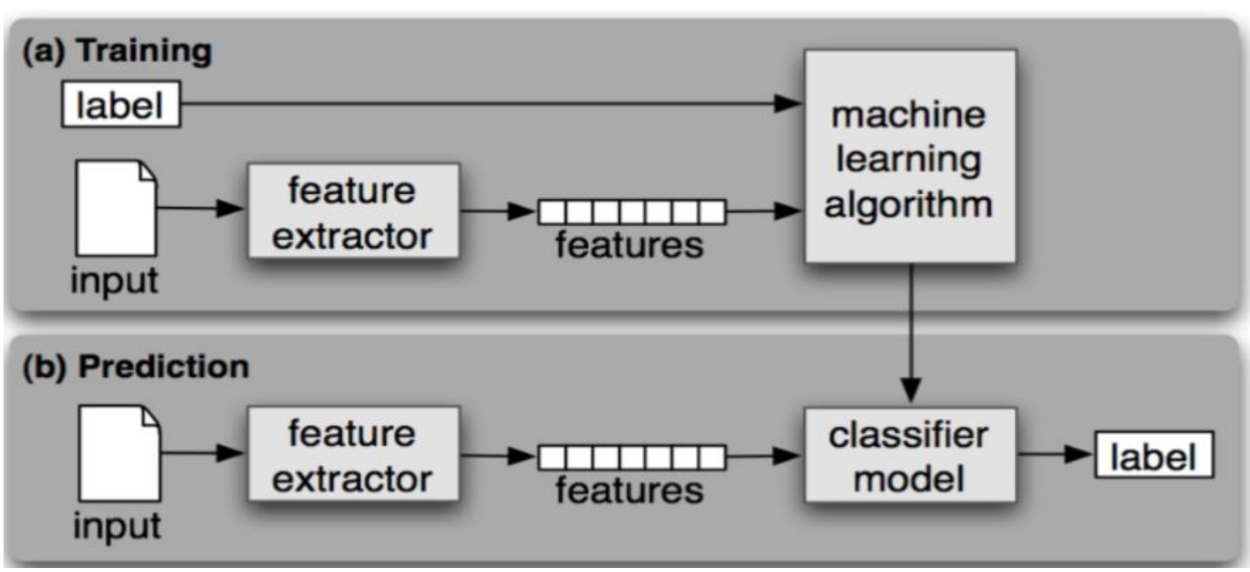
- *Polarity*: if the speaker express a *positive* or *negative* opinion,
- *Subject*: the thing that is being talked about,
- *Opinion holder*: the person, or entity that expresses the opinion.

Process of Sentiment Analysis

There are many methods and algorithms to implement sentiment analysis systems, which can be classified as:

- Rule-based systems that perform sentiment analysis based on a set of manually crafted rules.
- Automatic systems that rely on machine learning techniques to learn from data.
- Hybrid system combine both rule based & automatic approaches.

Automatic System : Automatic methods rely on ML techniques. The sentiment analysis task is usually modeled as a classification problem where a classifier is fed with a text and returns the corresponding category



Problem Statement

There is a lot of competition in the airlines industry. Any negative reviews can decrease the credibility of the carrier. So carriers are working hard to solve any issues whenever they receive any complaints. But lot of times issue are posted on social media so companies can get valuable insights about areas where they can improve if they are able to mine information from these reviews

Objective

Our goal is to use text mining to find the overall sentiment of the reviews . We also want to find out different kinds of trend in this dataset

Solution Design

- Data set : The Airline dataset is available in the Kaggle and it has nearly 10000 reviews and 15 different variables .

- Data Preprocessing/Exploration, Variable Selection : The dataset has 15 variables but after studying the data set t we found out that we just need 2 different variables for our analysis. For the purpose of cleaning the data we first separated sentences in the form of tokens. Then were removed all the emoticons and frequently occurring term from our sentences, then we converted every word in lower cases and lastly we striped white spaces from the dtm.

- Prediction/Classification/Time Series Forecasting/Unsupervised Learning: The dataset we have is labeled dataset(labeled as positive, negative & Neutral). So this is basically a supervised learning . This problem is classification problem and we have performed different classification techniques like Naïve Bayes, Random Forest, Support vector Mechanism etc.

- Predictors/Outcomes : we are doing this classification based on the text data
  1.) predictor variable is : text.
  2.) our Response variable is : airline_ sentiment

- Train/valid set : We have divided the training and validation set 70% : 30%
- Accuracy of Model : To check the accuracy of model we have used confusion matrix and crosstable.

Methodology

```
library(tm)

## Loading required package: NLP

library(e1071)

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

library(dplyr)

library(caret)

library(ggplot2)
library(tidyverse)

## -- Attaching packages ---------------------------------------------------
tidyverse 1.2.1 --

## v tibble  2.0.1     v purrr   0.3.0
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.3.1     v forcats 0.3.0

## -- Conflicts ------------------------------------------------------- tidyv
erse_conflicts() --
## x ggplot2::annotate() masks NLP::annotate()
## x dplyr::filter()     masks stats::filter()
## x dplyr::lag()        masks stats::lag()
## x purrr::lift()       masks caret::lift()

library(textclean)

## Warning: package 'textclean' was built under R version 3.5.3

library(RCurl)

library(tidytext)
library(tidyr)
library(magrittr)

library(RColorBrewer)
library(wordcloud)

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

library(caTools)

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

Loading the Data set

```
reviews_original <- read.csv(file.choose(), header = TRUE, stringsAsFactors =
F )
```

Data cleaning

```
reviews <- reviews_original[c(2,11)]
colnames(reviews) <- c("sentiment", "tweet")
glimpse(reviews)

## Observations: 14,640
## Variables: 2
## $ sentiment <chr> "neutral", "positive", "neutral", "negative", "negat...
## $ tweet     <chr> "@VirginAmerica What @dhepburn said.", "@VirginAmeri...

set.seed(1)
reviews<-reviews[sample(nrow(reviews)),]
reviews<-reviews[sample(nrow(reviews)),]
glimpse(reviews)

## Observations: 14,640
## Variables: 2
## $ sentiment <chr> "negative", "negative", "neutral", "negative", "nega...
## $ tweet     <chr> "@USAirways some people in line have been waiting 55...

reviews$sentiment <- as.factor(reviews$sentiment)

reviews_text <- reviews$tweet
reviews_text <- replace_emoji(reviews_text)
reviews_text <- replace_non_ascii(reviews_text)

corpus <- Corpus(VectorSource(reviews_text))
inspect(corpus[1:3])

## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 3
##
## [1] @USAirways some people in line have been waiting 55 minutes for a cust
omer service representative
## [2] @SouthwestAir are people in an exit row still supposed to give their v
erbal approvals? I was never asked on the flight I'm on...
## [3] @AmericanAir Thanks so much!

corpus_clean <- corpus %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(removePunctuation) %>%
  tm_map(removeNumbers) %>%
  tm_map(removeWords, stopwords(kind="en")) %>%
  tm_map(removeWords, c("virginamerica", "united", "flight","jetblue",
```

```r
                              "usairways","americanair","southwestair","get")) %>%
  tm_map(stripWhitespace)

inspect(corpus_clean[1:10])
```

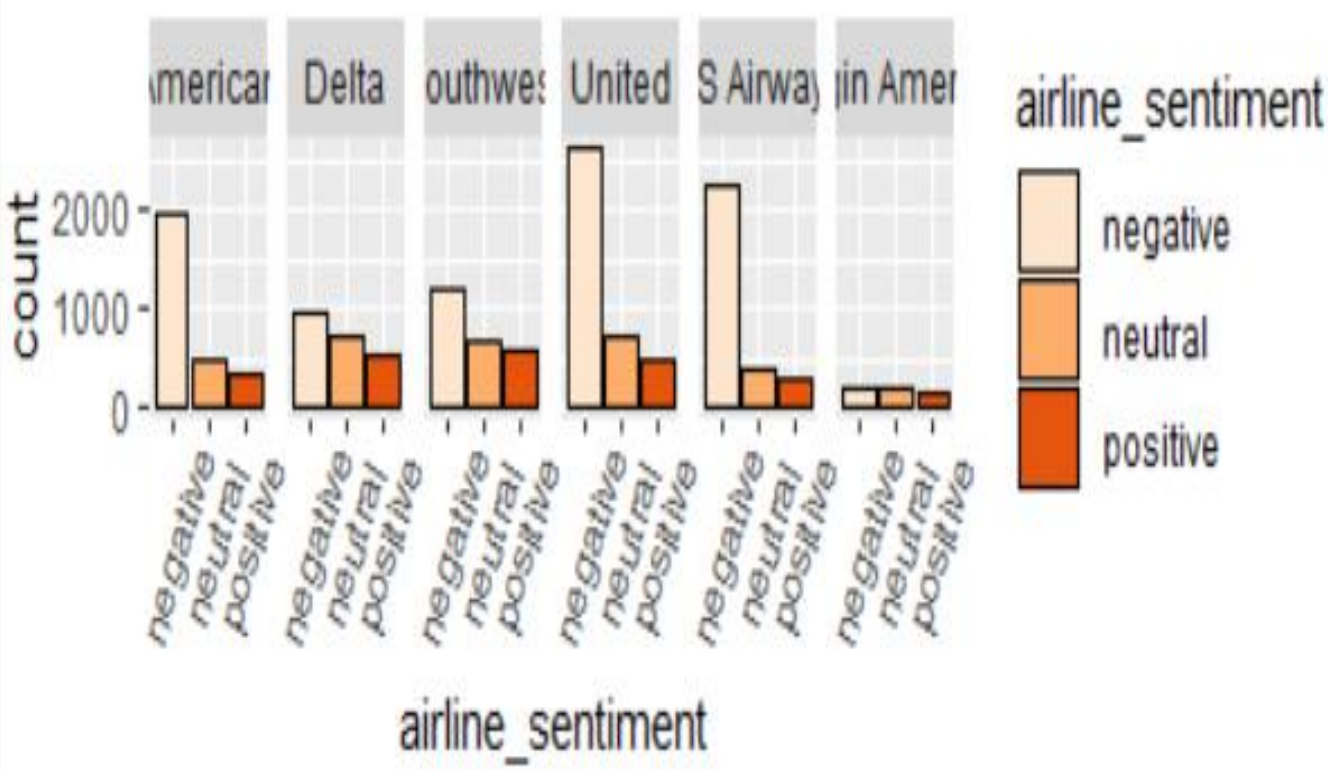
```
## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 19
##
##  [1]  people line waiting minutes customer service representative
##  [2]  people exit row still supposed give verbal approvals never asked im
##  [3]  thanks much
##  [4]  stand baggage claim hour waiting bag knew never made plane
##  [5]  lipstick pig still pig ur new baggage claim sju disaster chairs mins
wait baggage fail
##  [6]  hold almost two hours trying talk someone cancelled flighted tomorro
w morning suggest
##  [7]  website seriously horrible call center almost makes tempting fly som
eone else sodone
##  [8]  plenty empty seats coach dissatisfaction understood crew members tra
veling duty prebooked f
##  [9]  pilots plane says still time ground minutes ago
## [10]  never recd cancelled flightlation notice left w options fly ps drivi
ng la red eye mon w kids
```

```r
dtm <- DocumentTermMatrix(corpus_clean)
inspect(dtm[40:50, 10:15])
```

```
## <<DocumentTermMatrix (documents: 11, terms: 6)>>
## Non-/sparse entries: 2/64
## Sparsity           : 97%
## Maximal term length: 8
## Weighting          : term frequency (tf)
## Sample             :
##     Terms
## Docs exit give never row still supposed
##   40    0    0     0   0     0        0
##   41    0    0     0   0     1        0
##   42    0    0     0   0     0        0
##   43    0    0     0   0     0        0
##   44    0    0     0   0     0        0
##   45    0    0     0   0     0        0
##   46    0    0     0   0     0        0
##   47    0    0     0   0     1        0
##   48    0    0     0   0     0        0
##   49    0    0     0   0     0        0
```

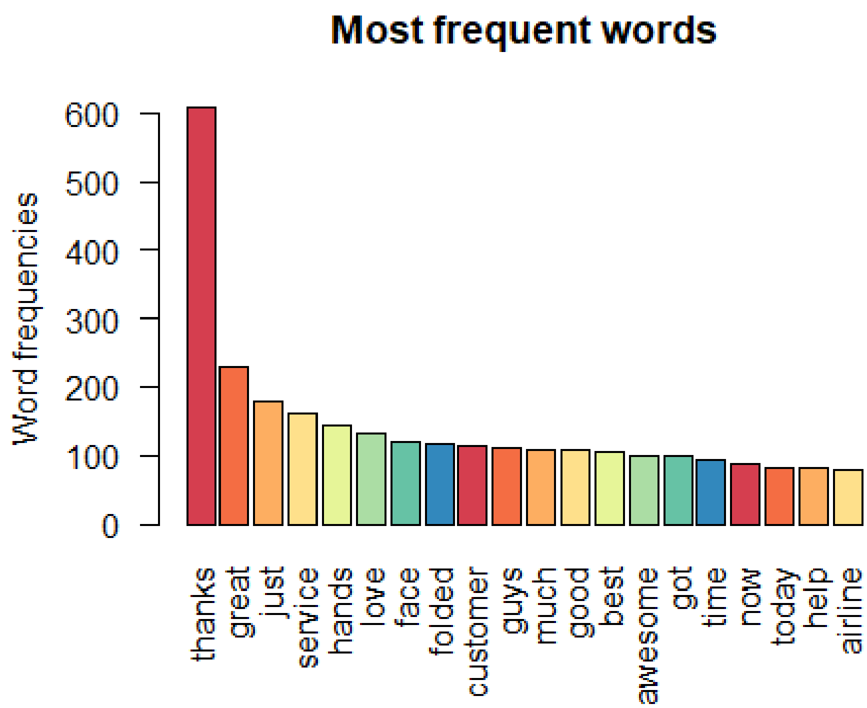
Data Exploration and Data Visualization

```
reviews_original <- read.csv(file.choose(), stringsAsFactors = F)
reviews1 <- reviews_original
#reviews <- reviews_original[c(2,6,11)]

table(reviews1$airline_sentiment)
## negative  neutral positive
##     9178     3099     2363

# So there are 2363 positive and 9178 negative reviews in out dataset
#reviews$text <- as.character(reviews$text)
#reviews_clean <- reviews %>%  unnest_tokens(word, text)

summary(reviews1$airline_sentiment)

##    Length     Class      Mode
##     14640 character

barplot <- ggplot(reviews1, aes(x = airline_sentiment, fill = airline_sentime
nt)) +
  geom_bar(color="black") +
  facet_grid(. ~ airline) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6),
        plot.margin = unit(c(3,0,3,0), "cm"))
barplot + scale_fill_brewer(palette="Oranges")
```



```r
#datacleaning and exploring

### For Positive Term

positive.reviews <- reviews1 %>% filter( airline_sentiment == "positive")
positive.text <- positive.reviews$text
#positive.text <- unnest_tokens(positive.text)
positive.text <-replace_emoji(positive.text)
positive.text <- replace_non_ascii(positive.text)

positive.corpus <- gsub("http.*","",positive.text)
positive.corpus <- Corpus(VectorSource(positive.corpus))
positive.corpus <- tm_map(positive.corpus, removePunctuation)

positive.corpus <- tm_map(positive.corpus, stripWhitespace)

positive.corpus <- tm_map(positive.corpus, content_transformer(tolower))

positive.corpus <- tm_map(positive.corpus, removeNumbers)

positive.corpus <- tm_map(positive.corpus, removeWords, stopwords("english"))

positive.corpus <- tm_map(positive.corpus, removeWords, c("virginamerica", "u
nited","flight","jetblue","usairways","americanair","southwestair","get",
"hour","can","will","amp",'cant',"one", "thank","flighted"))

positive.corpus <-  tm_map(positive.corpus, stripWhitespace)

inspect(positive.corpus[1:3])

## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 5
##
## [1]  plus youve added commercials experience tacky
## [2]  yes nearly every time fly vx ear worm wont go away
## [3]  well didntbut now d


positiveTDM <- TermDocumentMatrix(positive.corpus)

matrix.positive <- as.matrix(positiveTDM) #matric of tdm
sorting.positive <- sort(rowSums(matrix.positive),decreasing = T) #sorting
dataframe.positive <- data.frame(word=names(sorting.positive),freq=sorting.po
sitive) #frequency of words used


Spectral <- brewer.pal(8,"Spectral")
DarkColor <- brewer.pal(8,"Dark2")
barplot(dataframe.positive[1:20,]$freq, las = 2,
        names.arg = dataframe.positive[1:20,]$word,
        col =Spectral, main ="Most frequent words",
        ylab = "Word frequencies")
```

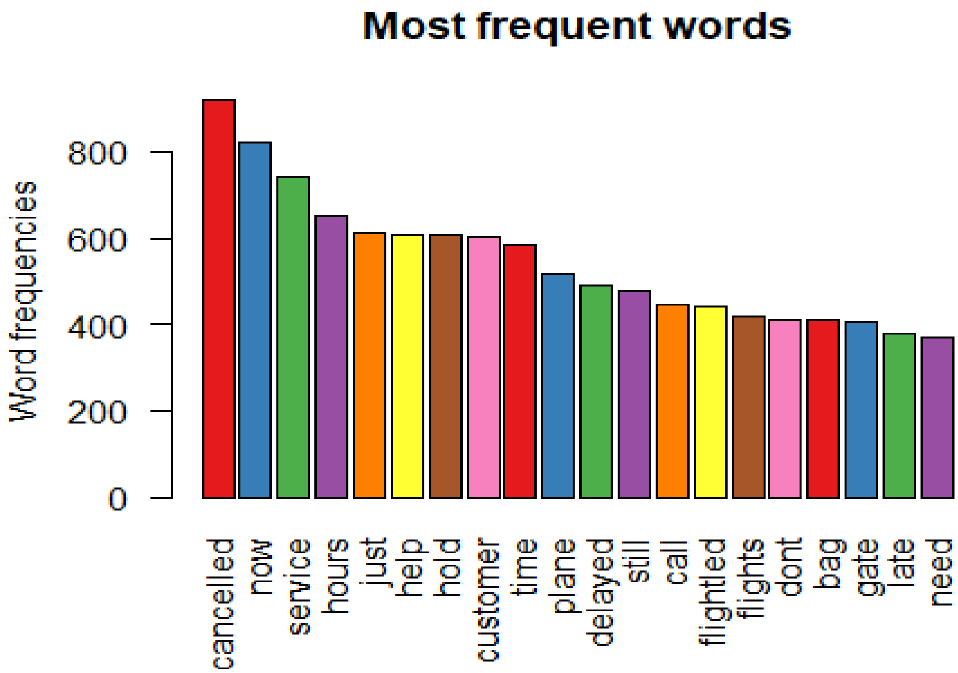
For Positive Terms




```
wordcloud(dataframe.positive$word,dataframe.positive$freq, scale=c(3,0.5), max.words=2000, random.order=FALSE, rot.per=0.4,
          colors=DarkColor,use.r.layout=T)
```

```
### For Negative Terms

negative.reviews <- reviews1 %>% filter( airline_sentiment == "negative")
negative.text <- negative.reviews$text
#negative.text <- unnest_tokens(negative.text)
negative.text <-replace_emoji(negative.text)
negative.text <- replace_non_ascii(negative.text)

negative.corpus <- gsub("http.*","",negative.text)
negative.corpus <- Corpus(VectorSource(negative.corpus))

negative.corpus <- tm_map(negative.corpus, removePunctuation) %>% tm_map(nega
tive.corpus, stripWhitespace) %>% tm_map(negative.corpus, content_transformer
(tolower)) %>% tm_map(negative.corpus, removeNumbers) %>% tm_map(negative.cor
pus, removeWords, stopwords("english")) %>% tm_map(negative.corpus, removeWor
ds, c("virginamerica", "united", "flight","jetblue","usairways","americanair"
,"southwestair","get","hour","can","will","amp",'cant',"one")) %>% tm_map(neg
ative.corpus, stripWhitespace)

inspect(negative.corpus[1:3])

## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 5
##
## [1]  really aggressive blast obnoxious entertainment guests faces little r
ecourse
## [2]  really big bad thing
## [3]  seriously pay seats didnt playing really bad thing flying va


negativeTDM <- TermDocumentMatrix(negative.corpus)



matrix.negative <- as.matrix(negativeTDM) #matric of tdm
sorting.negative <- sort(rowSums(matrix.negative),decreasing = T) #sorting
dataframe.negative <- data.frame(word=names(sorting.negative),freq=sorting.ne
gative) #frequency of words used


Set1 <- brewer.pal(8,"Set1")
Blues <- brewer.pal(6,"Blues")
barplot(dataframe.negative[1:20,]$freq, las = 2,
        names.arg = dataframe.negative[1:20,]$word,
        col =Set1, main ="Most frequent words",
        ylab = "Word frequencies")
```

For Negative Terms




```
wordcloud(dataframe.negative$word,dataframe.negative$freq, scale=c(3,0.5), ma
x.words=200, random.order=FALSE, rot.per=0.25,
          colors=Blues,use.r.layout=T)
```

]

```

```

Formation of training and validation set

```
reviews.train <- reviews[1:9232,]
reviews.test <- reviews[9233:11541,]

dtm.train <- dtm[1:9232,]
dtm.test <- dtm[9233:11541,]


corpus.clean.train <- corpus_clean[1:9232]
corpus.clean.test <- corpus_clean[9233:11541]


dim(dtm.train)

## [1]  9232 14235

fivefreq <- findFreqTerms(dtm.train, 5)
length((fivefreq))

## [1] 2073

dtm.train.nb <- DocumentTermMatrix(corpus.clean.train,
                                   control=list(dictionary = fivefreq))
dim(dtm.train.nb)

## [1] 9232 2073

dtm.test.nb <- DocumentTermMatrix(corpus.clean.test,
                                  control=list(dictionary = fivefreq))
dim(dtm.test.nb)

## [1] 2309 2073

convert_count <- function(x) {
  y <- ifelse(x > 0, 1,0)
  y <- factor(y, levels=c(0,1), labels=c("No", "Yes"))
  y
}
trainNB <- apply(dtm.train.nb, 2, convert_count)
testNB <- apply(dtm.test.nb, 2, convert_count)
```

Naive Bayes ML Algorithm

```
classifier <- naiveBayes(trainNB, reviews.train$sentiment, laplace = 1)

pred <- predict(classifier, newdata=testNB)

conf.mat <- confusionMatrix(pred, reviews.test$sentiment)
conf.mat

## Confusion Matrix and Statistics
##
##             Reference
## Prediction negative neutral positive
##   negative     1234     152       69
##   neutral       142     301       58
##   positive       68      46      239
##
## Overall Statistics
##
##                Accuracy : 0.7683
##                  95% CI : (0.7505, 0.7854)
##     No Information Rate : 0.6254
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.5667
##  Mcnemar's Test P-Value : 0.6298
##
## Statistics by Class:
##
##                      Class: negative Class: neutral Class: positive
## Sensitivity                   0.8546         0.6032          0.6530
## Specificity                   0.7445         0.8895          0.9413
## Pos Pred Value                0.8481         0.6008          0.6771
## Neg Pred Value                0.7541         0.8905          0.9351
## Prevalence                    0.6254         0.2161          0.1585
## Detection Rate                0.5344         0.1304          0.1035
## Detection Prevalence          0.6301         0.2170          0.1529
## Balanced Accuracy             0.7995         0.7464          0.7972

conf.mat$byClass

##                 Sensitivity Specificity Pos Pred Value Neg Pred Value
## Class: negative   0.8545706   0.7445087      0.8481100      0.7540984
## Class: neutral    0.6032064   0.8895028      0.6007984      0.8904867
## Class: positive   0.6530055   0.9413278      0.6770538      0.9350716
##                 Precision    Recall        F1 Prevalence Detection Rate
## Class: negative 0.8481100 0.8545706 0.8513280  0.6253790      0.5344305
## Class: neutral  0.6007984 0.6032064 0.6020000  0.2161109      0.1303595
## Class: positive 0.6770538 0.6530055 0.6648122  0.1585102      0.1035080
##                 Detection Prevalence Balanced Accuracy
## Class: negative            0.6301429         0.7995397
```

```
## Class: neutral             0.2169770          0.7463546
## Class: positive            0.1528800          0.7971667
```

```r
conf.mat$overall['Accuracy']
```

```
## Accuracy
## 0.768298
```

Support Vector ML Algorithm

```r
svm_classifier <- svm(dtm.train.nb,reviews.train$sentiment,kernel = "linear")
predSvm <- predict(svm_classifier,newdata = dtm.test.nb)
conf.mat1 <- confusionMatrix(predSvm, reviews.test$sentiment)
conf.mat1$overall['Accuracy']
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction negative neutral positive
##   negative     1124     325      204
##   neutral       254     144      125
##   positive       66      30       37
##
## Overall Statistics
##
##                Accuracy : 0.5652
##                  95% CI : (0.5447, 0.5855)
##     No Information Rate : 0.6254
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1202
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: negative Class: neutral Class: positive
## Sensitivity                   0.7784        0.28858         0.10109
## Specificity                   0.3884        0.79061         0.95059
## Pos Pred Value                0.6800        0.27533         0.27820
## Neg Pred Value                0.5122        0.80123         0.84881
## Prevalence                    0.6254        0.21611         0.15851
## Detection Rate                0.4868        0.06236         0.01602
## Detection Prevalence          0.7159        0.22650         0.05760
## Balanced Accuracy             0.5834        0.53959         0.52584
```

```r
conf.mat1$overall['Accuracy']
```

```
##  Accuracy
## 0.5651797
```

Conclusion

Text mining is a precious data mining technique which can be used in wide variety of fields for achieving desired outcomes. Sentiment analysis is done with the help of text mining.

Different Machine learning algorithm can be used for the classification of text documents (Random forest, Support vector Mechanism, Naïve Bayes). But we got the highest accuracy with Naïve Bayes Model which gave an accuracy of 77% .

Future Scope

Although text mining was introduced nearly a decade ago, even then we are not able to use it to its fullest potential. Although we have considerably improved the accuracy with the help of method like Hybrid methods. Task like classification are pretty easy but the main challenge is understating the meaning of documents. Even identical word in same order can have different meaning depending on the cultural and social context.

What else can be added to this project

1.) Different other classification techniques like random forest and Knn can be applied and then their accuracy cam pe compared and model with best accuracy can be determined

2.) A Text Mining app can be built using r shiny.

Reference

1.) Data Set Link

a.) https://www.kaggle.com/crowdflower/twitter-airline-sentiment

2.) For data cleaning

a..) Advanced method for data cleaning

https://northeastern.blackboard.com/bbcswebdav/pid-20083836-dt-content-rid-52868302_1/xid-52868302_1

b.) Text cleaning using tm package

http://www.mjdenny.com/Text_Processing_In_R.html

3.) For Data Exploration and data Visualization

a.)] Wickham, Hadley, and Garrett Grolemund. R For Data Science. OReilly, 2017

b.) https://ggplot2.tidyverse.org/

c.) https://r4ds.had.co.nz/

4.) For Modelling

a.) Classification Techniques

https://northeastern.blackboard.com/bbcswebdav/pid-20083835-dt-content-rid-52868303_1/xid-52868303_1

b.) Text Books Referred:

b.1) Machine Learning with R 2nd. Edition. 2015. Lantz

b.2) Data Mining for Business Analytics_ Concepts, Techniques, and Applications in R