Coursera Data Science Capstone Yelp Dataset Final Project

Andre Johnson

November 17, 2015

Introduction and Overview

Yelp was founded in 2004 to help people find great local businesses like restaurants, dentists, hair stylists and mechanics. Yelp had a monthly average of 83 million unique visitors who visited Yelp via their mobile device in Q2 2015*. Yelp has provided a subset of data from its users as part of a Dataset Challenge. This set includes information about local businesses in 10 cities across 4 countries.

The intention of this project is to utilize the reviews data provided in the Yelp Dataset for restaurants and explore the relationship between the quantity/ratios of positive versus negative and profane words used in reviews of restaurants in the dataset and the rating (number of stars) given. Furthermore we will attempt to develop the start of a model that evaluates the text of a review to predict the rating to be given.

Setting up R Environment - Prepare the R environment for analysis.

```
set.seed(1973)
require(ggplot2)
library(plyr)
require(tm)
require("knitr")
```

```
opts_chunk$set(cache=TRUE, cache.path = 'test_files/
', fig.path='figure/')
library(data.table)
library(stringr)
require(stringi)
library(caret)
```

```
require(rpart)
require(rpart.plot)
library(randomForest)
library(corrplot)
library(psych)
require(descr)
```

Task 1 - Data acquisition and cleaning - Download the data and load/manipulate it in R

The goal of this task is to get familiar with the databases and do the necessary cleaning. After this exercise, we should understand what real data looks like and how much effort into cleaning the data.

Download and Read Files

```
setwd("E:/User/Coursera/Capstone/Yelp Dataset Challenge/Data")
#for simplicity, already saved data as rds and read rds file as required
#url <- "https://d396qusza40orc.cloudfront.net/dsscapstone/dataset/yelp_dataset_challenge_academic_dataset.zip"   #if
(!file.exists("yelp_dataset_challenge_academic_dataset.zip")){    # download.file(url,
"yelp_dataset_challenge_academic_dataset.zip")
#unzip("yelp_dataset_challenge_academic_dataset.zip")
```

Read Files

```
#yelpReviewData <- fromJSON(paste("[",paste(readLines("yelp_aca
demic_dataset_review.json"), collapse=","),"]"))
#yelpBusinessData <- fromJSON(paste("[",paste(readLines("yelp_a
cademic_dataset_business.json"), collapse=","),"]"))
#convert the data from JSON to R readable format
#saveRDS(yelpReviewData, "yelpReviewData.rds")
#saveRDS(yelpBusinessData, "yelpBusinessData.rds")
#save data to rds for future use
#for simplicity, saved data as rds and read rds file as require
d
yelpReviewData <- readRDS("yelpReviewData.rds")
yelpBusinessData <- readRDS("yelpBusinessData.rds")
```

Create Sample File - Given the large size of the file, we will create a sample file of randomly selected lines from the yelpReviewData.

```
yelpReviewData <- yelpReviewData[,-c(1:3, 5, 7)]
#Delete unnecessary columns in yelpReviewData - everything but
text, stars, and business_id columns
yelpReviewSample <- yelpReviewData[sample(nrow(yelpReviewData),
size = 2000, replace = FALSE, prob = NULL),]
#create a sample of reviews data to work with
rm(yelpReviewData)
# remove yelpReviewData to save working memory
```

Merge with Data from subset of yelpBusinessData for Restaurants

```
yelpRestaurantBusinessData <- subset(yelpBusinessData, grepl("Restau
rants", yelpBusinessData$categories))
#subset yelpBusinessData into only restaurants
rm(yelpBusinessData) #remove yelpBusinessData to save working m
emory
yelpRestaurantBusinessData <- yelpRestaurantBusinessData[, -c(2
:4, 6:15)]
#delete columns not necessary for analysis
yelpRestaurantBusinessData$categories <- "Restaurants"
#replace categories with simply "Restaurants"

yelpReviewRestData <- merge(yelpReviewSample, yelpRestaurantBus
inessData, sort=FALSE)
#merge two datasets
rm(yelpReviewSample)
```

Clean up yelpReviewRestData

```
yelpReviewRestData$text <- tolower(yelpReviewRestData$text)
#change all letters to lower case
yelpReviewRestData$text <- str_replace_all(yelpReviewRestData$t
ext, "not ", "not" )
#remove space after not to convert to a positive or negative wo
rd
yelpReviewRestData$text <- str_replace_all(yelpReviewRestData$t
ext, "don't ", "don't" )
#remove space after don't to convert to a positive or negative
word
yelpReviewRestData$text <- str_replace_all(yelpReviewRestData$t
ext, "nothing ", "nothing" )
#remove space after nothing to convert to a positive or negativ
e word
```

```
rm(yelpRestaurantBusinessData)
# remove unneeded datasets to save working memory
yelpReviewRestData <- yelpReviewRestData[, -c(1, 4)]
#remove business_id and categories columns as no longer necessa
ry
```

```
yelpReviewRestData$text <- str_replace_all(yelpReviewRestData$t
ext, "never ", "never" )
#remove space after never to convert to a positive or negative
word
trimws(yelpReviewRestData$text)
#clean up text to get rid of extra whitespace
yelpReviewRestData$text <- gsub("[[:punct:]]", "", yelpReviewRe
stData$text)
#remove punctuation
```

Load positive, negative, and profane word lists - The lists of positive, negative, and profane lists have been updated t hroughout the process to try and capture positive, negative, and profane aspects of the reviews as much as possible which over time has improved the model significantly.

```
positivewords <- readLines("positivewords.txt", skipNul = TRUE)
negativewords <- readLines("negativewords.txt", skipNul = TRUE)
profanity <- readLines("profanity.txt", skipNul = TRUE)
```

Populate columns with number of positive, negative, profane, total words and percentages

```
yelpReviewRestData$`Count Positive words` <- lapply(yelpReviewR
estData$text, function(x) sum(stri_detect_fixed(x, positiveword
s)))
#count of positive words
yelpReviewRestData$`Count Negative Words` <- lapply(yelpReviewR
estData$text, function(x) sum(stri_detect_fixed(x, negativeword
s)))
#count of negative words
yelpReviewRestData$`Count Profane Words` <- lapply(yelpReviewRe
stData$text, function(x) sum(stri_detect_fixed(x, profanity)))
#count of profane words
yelpReviewRestData$`Total Words` <- lapply(yelpReviewRestData$t
ext, function(x) length(strsplit(x, ' ')[[1]]))
#count of total words
yelpReviewRestData$`Count Positive words` <- as.numeric(yelpRev
iewRestData$`Count Positive words`)
yelpReviewRestData$`Count Negative Words` <- as.numeric(yelpRev
iewRestData$`Count Negative Words`)
yelpReviewRestData$`Count Profane Words` <- as.numeric(yelpRevi
ewRestData$`Count Profane Words`)
yelpReviewRestData$`Total Words` <- as.numeric(yelpReviewRestDa
ta$`Total Words`)
yelpReviewRestData$stars <- as.numeric(yelpReviewRestData$stars
)
#convert entries to numeric
rm(negativewords)
rm(positivewords)
rm(profanity)
yelpReviewRestData$`Percentage of Pos Words to Total` <- round(
(yelpReviewRestData$`Count Positive words`/ yelpReviewRestData$
`Total Words`) * 100)
yelpReviewRestData$`Percentage of Neg Words to Total` <- round(
(yelpReviewRestData$`Count Negative Words`/ yelpReviewRestData$
`Total Words`) * 100)
#create columns for round of percentage of positive and negativ
e words to total words in review
yelpReviewRestData$`Count Negative Words` <- str_replace_all(ye
lpReviewRestData$`Count Negative Words`, "0", "1" )
#change negative words count where 0 to 1 so that will not resu
lt in Inf
yelpReviewRestData$`Count Negative Words` <- as.numeric(yelpRev
iewRestData$`Count Negative Words`)
yelpReviewRestData$`Ratio of Pos to Neg Words` <- yelpReviewRes
tData$`Count Positive words`/ yelpReviewRestData$`Count Negativ
e Words`
#create column for ratio of positive to negative words
```

```
yelpReviewRestData$`Ratio of Pos to Neg Words`[yelpReviewRestDa
ta$`Ratio of Pos to Neg Words` < 1] <- 0
#Change Ratio of Pos to Neg Words where less than 1 to 0 to buc
ket Negative words exceeding positive words as one factor
yelpReviewRestData <- yelpReviewRestData[, -c(2:4, 6)]
#delete text, count positive words, count negative words, and t
otal words columns as no longer necessary and redundant with ot
her factors
yelpReviewRestDataTrain <- yelpReviewRestData
yelpReviewRestDataTrain$`Ratio of Pos to Neg Words`[yelpReviewR
estDataTrain$`Ratio of Pos to Neg Words` > 3] <- "Very Positive
"
yelpReviewRestDataTrain$`Ratio of Pos to Neg Words`[yelpReviewR
estDataTrain$`Ratio of Pos to Neg Words` <= 3 & yelpReviewRestD
ata$`Ratio of Pos to Neg Words` >=1] <- "Positive"
yelpReviewRestDataTrain$`Ratio of Pos to Neg Words`[yelpReviewR
estDataTrain$`Ratio of Pos to Neg Words` < 1] <- "Negative"
#Bucket Ratios into Negative, Positive, and Very Positive
yelpReviewRestDataTrain$`Percentage of Pos Words to Total`[yelp
ReviewRestDataTrain$`Percentage of Pos Words to Total` > 5] <-
"Very Positive"
yelpReviewRestDataTrain$`Percentage of Pos Words to Total`[yelp
ReviewRestDataTrain$`Percentage of Pos Words to Total` <= 5 & y
elpReviewRestData$`Percentage of Pos Words to Total` >=2] <- "P
ositive"
yelpReviewRestDataTrain$`Percentage of Pos Words to Total`[yelp
ReviewRestDataTrain$`Percentage of Pos Words to Total` < 2] <-
"Negative"
#Bucket Ratios into Negative, Positive, and Very Positive
yelpReviewRestDataTrain$`Percentage of Neg Words to Total`[yelp
ReviewRestDataTrain$`Percentage of Neg Words to Total` > 3] <-
"Very Negative"
yelpReviewRestDataTrain$`Percentage of Neg Words to Total`[yelp
ReviewRestDataTrain$`Percentage of Neg Words to Total` <= 3 & y
elpReviewRestData$`Percentage of Neg Words to Total` >=1] <- "N
egative"
yelpReviewRestDataTrain$`Percentage of Neg Words to Total`[yelp
ReviewRestDataTrain$`Percentage of Neg Words to Total` < 1] <-
"Positive"
#Bucket Ratios into Negative, Positive, and Very Positive
```

After testing the model against individual stars, it was determined that predictability of the model was not nuanced enough to determine whether a 4 or 5 stars, 3 stars, or 1 or 2 star so we decided to bucket the 4 and 5 stars as Good, 1 and 2 stars as Bad and 3 stars as Average to try and improve the predictability in smaller buckets with the possibility of expanding back to 1-5 stars once further enhancement to review of text and the model has been done.

```
yelpReviewRestDataTrain$stars <- str_replace_all(yelpReviewRest
DataTrain$stars, "5", "Good" )
#bucket 4 and 5 stars as "Good"
yelpReviewRestDataTrain$stars <- str_replace_all(yelpReviewRest
DataTrain$stars, "4", "Good" )
#bucket 4 and 5 stars as "Good"
yelpReviewRestDataTrain$stars <- str_replace_all(yelpReviewRest
DataTrain$stars, "3", "Average" )
#bucket 3 stars as "Average"
yelpReviewRestDataTrain$stars <- str_replace_all(yelpReviewRest
DataTrain$stars, "2", "Bad" )
```

```
yelpReviewRestDataTrain$stars <- as.factor(yelpReviewRestDataTr
ain$stars)
yelpReviewRestDataTrain$`Count Profane Words` <- as.factor(yelp
ReviewRestDataTrain$`Count Profane Words`)
yelpReviewRestDataTrain$`Ratio of Pos to Neg Words` <- as.facto
r(yelpReviewRestDataTrain$`Ratio of Pos to Neg Words`)
yelpReviewRestDataTrain$`Percentage of Pos Words to Total` <- a
s.factor(yelpReviewRestDataTrain$`Percentage of Pos Words to To
tal`)
```

```
#bucket 2 and 1 stars as "Bad"
yelpReviewRestDataTrain$stars <- str_replace_all(yelpReviewRestDataTrain$stars, "1", "Bad" )
```

```
yelpReviewRestDataTrain$`Percentage of Neg Words to Total` <- as.factor(yelpReviewRestDataTrain$`Percentage of Neg Words to Total`)
#convert to factors for use in model
```

Task 2 - Exploratory analysis

The first step in building a predictive model for text is understanding the distribution and relationship between the types of words used and the rating. The goal of this task is to understand the basic relationships observed in the data and prepare to build a predictive model. We will perform a thorough exploratory analysis of the data, understanding the distribution of positive, negative and profane words and relationship between the words and the rating. First, we look at frequency of stars, positive to negative, positive and negative words to total, and total profane words in the data.

```
summary(yelpReviewRestDataTrain)
```

```
stars        Count Profane Words Percentage of Pos Words to Total
 Average:193   0:906               Negative     : 67
 Bad    :250   1:276               Positive     :630
 Good   :834   2: 75               Very Positive:580
               3: 16
               4:  4
 Percentage of Neg Words to Total     Ratio of Pos to Neg Words
 Negative     :750                    Negative     :102
 Positive     :332                    Positive     :670
 Very Negative:195                    Very Positive:505
```

```
table(yelpReviewRestData$stars, yelpReviewRestDataTrain$`Percentage of Pos Words to Total`)
```

```
   Negative Positive Very Positive
 1       17       80            21
 2       14       89            29
 3       12      111            70
 4        9      194           210
 5       15      156           250
```

```
table(yelpReviewRestData$stars, yelpReviewRestDataTrain$`Percentage of Neg Words to Total`)
```

```
   Negative Positive Very Negative
 1       61       21            36
 2       90       16            26
 3      122       49            22
 4      262      102            49
 5      215      144            62
```

```
table(yelpReviewRestData$stars, yelpReviewRestDataTrain$`Ratio of Pos to Neg Words`)
```

```
   Negative Positive Very Positive
 1       33       73            12
 2       22       87            23
 3       15      102            76
 4       13      214           186
 5       19      194           208
```

Figure 4 in the Appendix will show a histogram and cross tabulation plots of the different factors and **Figure 5** shows box plots of Stars vs the other factors.

The cross tabulation tables and box plots show a clear trend of higher positive to negative ratio, less negative words, and less profanity in text where the stars are 4 or 5. And the opposite trend of lower positive to negative ratio, more negative words, and more profanity in text where the stars are 1 or 2 which gives hope that a model can be developed to provide some predictability in the stars to be given by analysis of the text written.

Methods - Task 3 - Develop a Model

Having cleaned up the data which now includes 5 factors we are going to develop a model using *Random Forest* technique to allow us to predict the number of stars to be given based on the text submitted. "randomForest implements Breiman's random forest algorithm (based on Breiman and Cutler's original Fortran code) for classification and regression. It can also be used in unsupervised mode for assessing proximities among data points."

Create new Training and Testing data sets

First, with the data cleaned up in the "yelpReviewRestDataTrain" set, we create a training and test set.

```
inTrain = createDataPartition(y=yelpReviewRestDataTrain$star, p=0.7, list=FALSE)
train = yelpReviewRestDataTrain[inTrain,]
test = yelpReviewRestDataTrain[-inTrain,]
```

```
dim(train)
rm(yelpReviewRestDataTrain)
```

Developing a Model

Now that we have a train and test data set we are going to develop a model using *Random Forest* technique to allow us to predict the number of stars to be given based on the text submitted. We will use a 10-fold cross validation in the model.

```
RFcontrol <- trainControl(method="cv", number=10)
RFmodel = train(stars ~ ., method="rf", data=train, trControl=RFcontrol, ntree = 250)
saveRDS(RFmodel, "rfmodel.RDS")
#RFmodel <- readRDS("rfmodel.RDS")
RFmodel
```

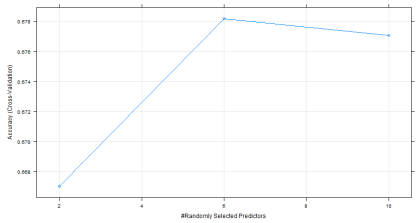
```
Random Forest
895 samples
  4 predictor
  3 classes: 'Average', 'Bad', 'Good'
No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 805, 806, 805, 806, 805, 805, ...
Resampling results across tuning parameters:
  mtry  Accuracy   Kappa       Accuracy SD  Kappa SD
   2    0.6670327  0.09074422  0.02505079   0.06901716
   6    0.6781691  0.17427202  0.02146964   0.06661254
  10    0.6770832  0.16772491  0.02802949   0.08081065

Accuracy was used to select the optimal model using the largest value.  The final value used for the model was mtry = 6.
```

Maximum accuracy is at **mtry = 6** and is **66.78%** with out of sample error of **33.22%** which is a relatively good start. Below is a plot of the random forest model.

Figure 1 - Plot of RFmodel

```
plot(RFmodel)
```



Testing and Evaluating the Model

Now that we have created a model against the training data, let's test it against the test data set.

```
confusionMatrix(test$stars, predict(
RFmodel, test))
```

```
Confusion Matrix and Statistics
          Reference
Prediction Average Bad Good
   Average       1   8   48
   Bad           2  12   61
   Good          1  11  238
Overall Statistics
                Accuracy : 0.6571
                  95% CI : (0.6071, 0.7046)
    No Information Rate : 0.9084
    P-Value [Acc > NIR] : 1

                   Kappa : 0.1162
 Mcnemar's Test P-Value : <2e-16
Statistics by Class:
                     Class: Average Class: Bad Class: Good
Sensitivity                0.250000    0.38710      0.6859
Specificity                0.851852    0.82051      0.6571
Pos Pred Value             0.017544    0.16000      0.9520
Neg Pred Value             0.990769    0.93811      0.1742
Prevalence                 0.010471    0.08115      0.9084
Detection Rate             0.002618    0.03141      0.6230
Detection Prevalence       0.149215    0.19634      0.6545
Balanced Accuracy          0.550926    0.60380      0.6715
```

```
RFModelAccuracy <- mean(predict(RFmodel, test) == test$stars) * 100
RFModelAccuracy
[1] 65.70681
```

The estimated accuracy of the model is **65.71%** when applied to the test data set with an out of sample error of **34.29 %.** A plot of each variable is below.

Figure 2 - Plot of variable importance of RFModel

```
plot(varImp(RFmodel))
```

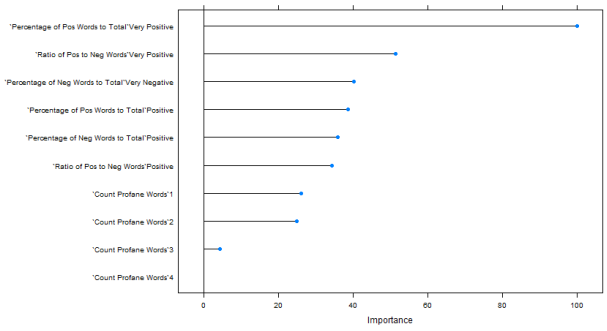
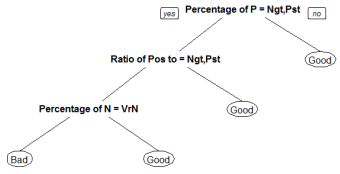


Figure 3 - Plot of Decision Tree model

```
decTree <- rpart(train$stars ~ .,
data=train, method="class")
prp(decTree)
```



Results - Conclusion

A random forest trainer appears to be a great start in developing a model to predict against the data sets with an out of sample of **34.29%.** Further work will be required to refine the model and the predictability, in particular a review of t he positivewords, negativewords, and profanity sets to be able to more accurately capture positive and negative sentiments and nuances in the text and data. As is seemingly apparent in **Figures 4** and **5** in the Appendix, there is a relationship between the number of stars given and the quantity/ratio of positive vs negative words and profane words in the text of the review. While the model ultimately requires further refinement to improve predictability, clearly there is a relationship that exists in the text and the stars given.

Discussion - Summary and Next Steps

This capstone project is the first step in understanding/preprocessing/exploring the data and developing a model that will be able to predict the rating with some consistency based on the text in the review. The work done so far is a good step in the right direction having achieved a **66%** predictability level but further work will be required to refine the model to improve predictability, in particular a review of the positivewords, negativewords, and profanity sets to be able to more accurately capture positive and negative sentiments and nuances in the text and data.

Appendices

Figure 4 - Plots of Stars and Cross Tabulation Tables of Stars vs. Other Factors

```
par(mfrow=c(3,2))
hist(yelpReviewRestData$stars, xlab="stars"
, main = "Histogram of Frequency of Stars",
col = (c("lightblue","gray")))
crosstab(yelpReviewRestData$stars, yelpRevi
ewRestDataTrain$`Count Profane Words`, xlab
="Count of Profane Words", ylab="Stars", ma
in="Cross Table", col = (c("lightblue","gra
y")))
crosstab(yelpReviewRestData$stars, yelpRevi
ewRestDataTrain$`Percentage of Pos Words to
Total`, xlab="Positive Words to Total", yla
b="Stars", main="Cross Table", col = (c("li
ghtblue","gray")))
crosstab(yelpReviewRestData$stars, yelpRevi
ewRestDataTrain$`Percentage of Neg Words to
Total`, xlab="Negative Words to Total", yla
b="Stars", main="Cross Table", col = (c("li
ghtblue","gray")))
crosstab(yelpReviewRestData$stars, yelpRevi
ewRestDataTrain$`Ratio of Pos to Neg Words`
, xlab="Ratio of Positive to Negative Words
", ylab="Stars", main="Cross Table", col =
(c("lightblue","gray")))
```

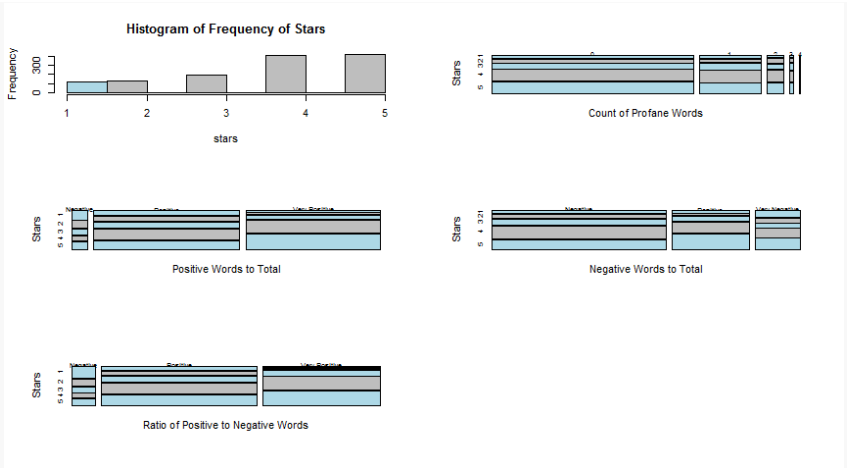
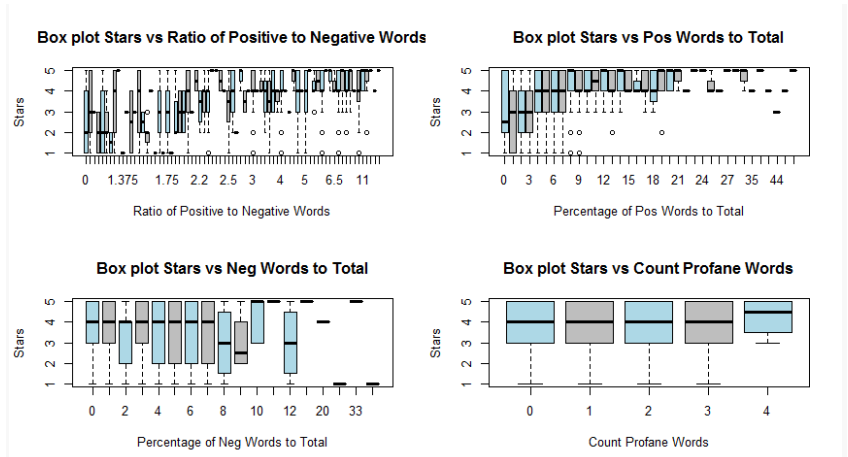


Figure 5 - Boxplots of Stars vs Other Factors

```
par(mfrow=c(2,2))
boxplot(yelpReviewRestData$star ~ yelpRevie
wRestData$`Ratio of Pos to Neg Words`, col
= (c("lightblue","gray")), ylab = "Stars",
xlab = "Ratio of Positive to Negative Words
", main = "Box plot Stars vs Ratio of Posit
ive to Negative Words" )
boxplot(yelpReviewRestData$star ~ yelpRevie
wRestData$`Percentage of Pos Words to Total
`, col = (c("lightblue","gray")), ylab = "S
tars", xlab = "Percentage of Pos Words to T
otal", main = "Box plot Stars vs Pos Words
to Total" )
boxplot(yelpReviewRestData$star ~ yelpRevie
wRestData$`Percentage of Neg Words to Total
`, col = (c("lightblue","gray")), ylab = "S
tars", xlab = "Percentage of Neg Words to T
otal", main = "Box plot Stars vs Neg Words
to Total" )
boxplot(yelpReviewRestData$star ~ yelpRevie
wRestData$`Count Profane Words`, col = (c("
lightblue","gray")), ylab = "Stars", xlab =
"Count Profane Words", main = "Box plot Sta
rs vs Count Profane Words" )
```



References:

**http://www.yelp.com/about
**http://www.enchantedlearning.com/wordlist/negativewords.shtml
**http://www.enchantedlearning.com/wordlist/positivewords.shtml
Banned Words List: http://www.bannedwordlist.com/lists/swearWords.txt
**https://en.wikipedia.org/wiki/N-gram
**http://www.inside-r.org/packages/cran/randomforest/docs/randomforest
Natural language processing Wikipedia page: http://en.wikipedia.org/wiki/Natural_language_processing
Text mining infrastucture in R: http://www.jstatsoft.org/v25/i05/
CRAN Task View: Natural Language Processing: http://cran.r-project.org/web/views/NaturalLanguageProcessing.html
Basic Text Mining in R: https://rstudio-pubs-static.s3.amazonaws.com/31867_8236987cf0a8444e962ccd2aec46d9c3.html