# Predicting the Usefulness of a Yelp Review Using Machine Learning

Mohammed K. Barakat

November 8, 2015

## Introduction

For many purchased products or offered services there is usually a way to reflect on the customer's experience using online review. Yelp's website is a place that collects such reviews of various businesses.

This research tries to answer the question *"Can we predict to what extent a user's review for a business is useful by predicting the number of "useful" votes the review will receive and by predicting a "usefulness" category of the review?"* The prediction algorithm is based on the user's profile and on quantitative features of the review she/he writes. This analysis is expected to be of interest to yelp.com, yelpers, and businesses as it helps exploit potential useful reviews as soon as they are posted to improve businesses and provide quicker recommendations to potential customers.

**Note:** *To reproduce the same analysis and results you can visit the github repository at (https://github.com/Mohammedkb/MBCapstoneProject) for the complete R code in a .Rmd file.*

## Methods and Data

This section discusses the input data used in the analysis, data processing, Exploratory Data Analysis to give insight into the data, and the Prediction Algorithms used to predict the outcome of review usefulness.

### Input data

The research uses a dataset that is part of the Yelp Dataset Challenge that corresponds to Round 6 of their challenge. The data consists of 5 JSON-formatted files available under this link. The files are: **business** data, **checkin** data, **review** data, **tip** data, and the **user** data. The datasets contain details about businesses and users profiles in addition to users reviews posted between 2004 and 2015. Raw data is first downloaded, unzipped, read into R, then processed into regular data frames instead of JSON nested data frames.

### Data Processing

Raw data needs to be processed in order to extract the required features for further analysis. The analysis explores data variables and selects the outcome and initial predictors. Variables are processed, merged into one final set, and, finally, screened to come up with the vital predictors used in the prediction.

#### Outcome variable and predictors

After studying the variables existing in *userData* and *reviewData* it's become clear that the *votes.useful* variable in the *reviewData* can be used as the **outcome** variable, whereas a combination of other variables from both datasets can be used to build model initial predictors (listed below).

- **text**: the review text (source: *reviewData*)
- **yelping_since**: a user's starting date of yelping (source: *userData*)
- **review_count**: a user's total count of reviews (source: *userData*)
- **friends**: a user's list of friends' IDs (source: *userData*)
- **fans**: a user's count of fans (source: *userData*)
- **elite**: a user's "elite" status in Yelp (source: *userData*)
- **votes.useful**: a user's total count of "useful" votes received in Yelp (source: *userData*)
- **compliments**: a user's total count of compliments received (by type) (source: *userData*)

## Variables processing and datasets merging

The prediction algorithm will predict the extent of review usefulness in two ways:

1. Predicting the number of *useful* votes for a review
2. Predicting the *usefulness* category for a review as either "Not Useful", "Slightly Useful", "Moderately Useful", "Highly Useful", or "Extremely Useful"

Hence, a new categorical variable representing review usefulness is required. Values of this variable are based on the number of *useful* votes using the following assumptions:

- **Not Useful**: if number of *useful* votes is <= 1
- **Slightly Useful**: if number of *useful* votes is >= 2 and <= 5
- **Moderately Useful**: if number of *useful* votes is >= 6 and <= 15
- **Highly Useful**: if number of *useful* votes is >= 16 and <= 25
- **Extremely Useful**: if number of *useful* votes is >= 26

The *usefulCat* function in the code is used to assign the Usefulness Category to each existing review. Besides, initial predictors need to be processed as explained below to make up the final predictors.

- *text* will be used to make the **textLen** predictor which is the length of review text.

- *votes.useful* (source: *userData*), which reflects the overall count of a user's *useful* votes received, will be renamed to **TotVotes.useful** to distinguish it from the outcome variable *votes.useful*

- *yelping_since* will be used to make the **yelp_months** predictor as the yelping period in months.

- *elite* will be used to make up the **is.elite** predictor as whether the user has an elite status or not.

- *friends* will be used to come up with the number of friends of a user as **num_friends** variable.

In addition to the above steps the processTbl function in the code performs the below processing steps:

1. replaces NA's by zeros
2. merges needed variables in both datasets into a new dataset (mergedData) using the user_id.
3. drops unnecessary variables and removes incomplete cases (rows with NA's) from the mergedData

## Variables screening

Since including extra predictors increases standard errors of regression another variable screening step is performed. A **Correlation Test** for each *numeric* variable versus the output variable is done. Variables with the highest Correlation Coefficients (R) are selected.

```
##                 Variable  corr
## 1        TotVotes.useful 0.453
## 2       compliments.cool 0.426
## 3       compliments.note 0.421
## ...                 <NA>   ...
## 15   compliments.profile 0.257
## 16      compliments.list 0.246
## 17           yelp_months 0.199
```

Correlation Test results show that correlation coefficients range between 0.199 and 0.453. By setting a minimum of **0.25** we get **16 final predictors** to be used in prediction models (including non-numeric *is.elite*).

```
##  [1] "textLen"             "review_count"        "fans"
##  [4] "TotVotes.useful"     "compliments.profile" "compliments.cute"
##  [7] "compliments.funny"   "compliments.plain"   "compliments.writer"
## [10] "compliments.note"    "compliments.photos"  "compliments.hot"
## [13] "compliments.cool"    "compliments.more"    "is.elite"
## [16] "num_friends"
```

## Exploratory Data Analysis

In this section we try to explore features of the *mergedData* using descriptive statistics and plots.

### Exploring the outcome variable

By looking at the **Five-Number Summary** of the *useful* votes granted to reviews we can see that *useful* votes range between 0 vote to 166 votes with an average of 1.072 *useful* vote per review.

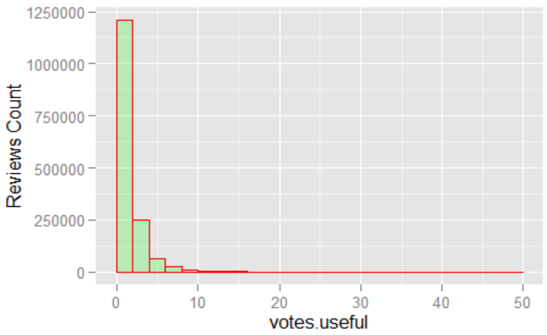
```
summary(mergedData$votes.useful)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   0.000   0.000   1.072   1.000 166.000
```

The **Histogram** and **Cumulative Curve** below show that 99% of reviews have received 0 to 10 *useful* votes.

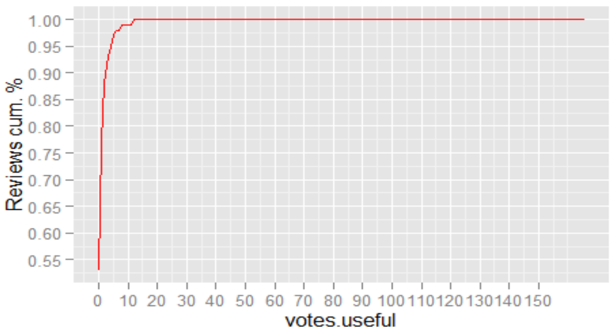
Number of Reviews by count of Useful Votes

Fig.1



Cummulative % of Reviews with 'Useful' Votes

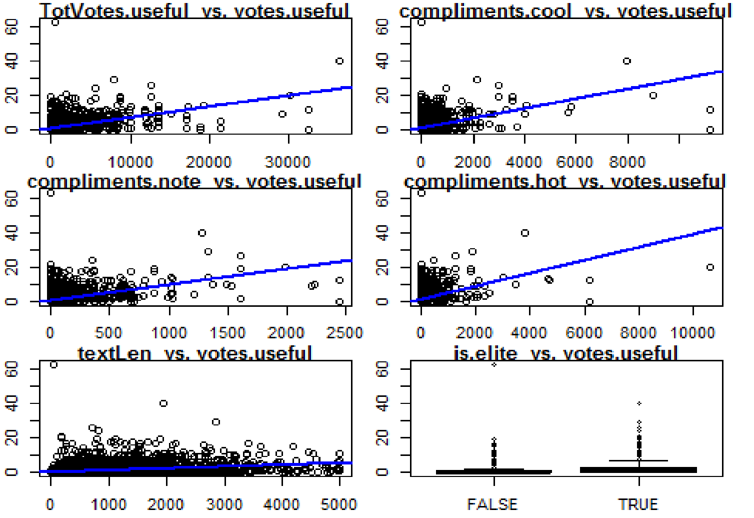
Fig.2



### Exploring relationships between data features

The plots below show the relationships between the outcome variable and some predictors.

Useful Votes vs. some Predictors (Fig.3)



In general, the selected predictors show a positive relationship with the outcome. Yet, the prediction model can give more insight into the ability to predict a useful review using the combination of these predictors.

## Prediction Algorithms

As explained above, the research predicts the extent of usefulness in two ways: by predicting the number of *useful* votes, and by predicting the *usefulness* category of a review. Hence, two models (**Model-A** and **Model-B**) will be fitted then used in both types of predictions. To train the models then test their accuracies we need to split the *mergedData* dataset into *training* and *testing* datasets (70/30 ratio).

```
set.seed(4040)
inTrain<-createDataPartition(y=mergedData$votes.useful,p=0.7,list = FALSE)
training<-mergedData[inTrain,];testing<-mergedData[-inTrain,]
```

Data partitioning resulted in **1098485** records for *training* and **470779** records for *testing*.

### Model-A fitting and training

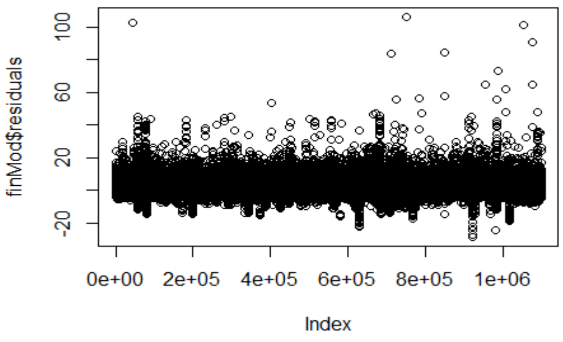
In **Model-A** we try to fit then predict the number of *useful* votes. Since the outcome (**votes.useful**) is a discrete variable we will use the **Generalized Linear Model** (glm) method in our prediction algorithm. For a better estimate of prediction accuracy we will use **10-fold Cross Validation**. (model summary below)

```
## Generalized Linear Model
##
## 1098485 samples
##      20 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 988637, 988637, 988636, 988636, 988637, 988637, ...
## Resampling results
##
##   RMSE      Rsquared  RMSE SD     Rsquared SD
##   1.674976  0.331327  0.02083372  0.007090576
##
##
```

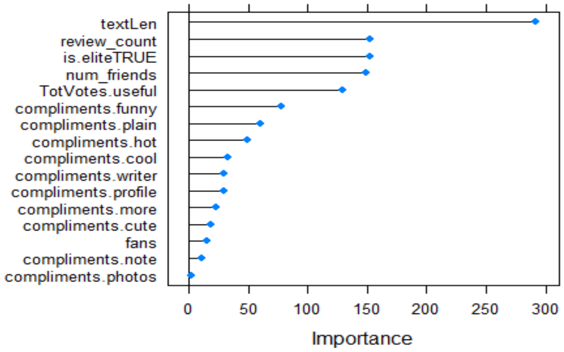
Error of prediction algorithms that are based on regression models are measured in the **RMSE** value (Root Mean Squared Error). The trained model showed **1.67** "useful" votes as RMSE value.

Model accuracy can also be measured by plotting the *residuals* of the model as shown in the figure. The residuals plot shows no specific patterns. Residuals are symmetrical around zero and, hence, randomly distributed. Besides, it is worth checking predictors importance in the model. The Variable Importance plot below shows that the *textLen* variable is the most important whereas the *compliments.photos* is the least.

Model Residuals Plot (Fig.4)



Variables Importance Plot (Fig.5)



### Model-A testing

Now we need to test the model performance in predicting the outcome (*votes.useful*) on the testing dataset.

```
pred<-predict(ModAFit,testing)
ModAFitOutErr<-round(RMSE(pred,testing$votes.useful,na.rm = TRUE),2)
```

The out-of-sample error resulted from the prediction (represented by the RMSE value) is **1.68** "useful" votes.

### Model-B fitting and training

In Model-B we try to fit then predict the *usefulness* category of each review. Since the outcome (**Useful_Category**) is a categorical variable we will use the **Random Forest** prediction algorithm with **5-fold Cross Validation** for better estimate of the prediction error. Shown below is the model summary.

```
## Random Forest
##
## 109848 samples
##     21 predictor
##      5 classes: 'Extremely Useful', 'Highly Useful', 'Moderately Useful', 'Not Use
ful', 'Slightly Useful'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 87878, 87878, 87879, 87879, 87878
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.7898187  0.3002232  0.001759954  0.006201644
##    9    0.7848299  0.3181106  0.001792131  0.006420139
##   16    0.7797502  0.3097951  0.002064759  0.006604697
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

The accuracy resulted from fitting Model-B is **79%**. Knowing that the in-sample error equals (1-Accuracy) the in-sample error is estimated to be **21%**.

### Model-B testing

Now we need to test model performance in predicting the outcome (*Useful_Category*) on the testing dataset.

```
rfPred<-predict(ModBFit,testing)
rfCM<-confusionMatrix(rfPred,testing$Useful_Category);rfCM$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   7.907957e-01   3.015850e-01   7.896310e-01   7.919568e-01   7.700875e-01
## AccuracyPValue  McnemarPValue
##  1.005076e-255   0.000000e+00
```

The accuracy resulted from testing Model-B is **79%**. Knowing that the out-of-sample error equals (1-Accuracy) the out-of-sample error is estimated to be **21%**.

## Results

The analysis used two datasets; *User* and *Review*. The datasets were processed, cleaned, and merged into a single dataset (*mergedData*) that contained two outcome variables (**votes.useful** and **Useful_Category**) and 16 predictors. EDA showed insight into the data features and some correlation between the variables.

### Two models of prediction were trained and tested:

**Model-A**: predicts the number of *useful* votes of a review. It's based on the *glm* algorithm, 10-fold cv, 16 predictors, and the outcome variable (*votes.useful*). In-sample-error (RMSE) was 1.67 "useful" votes, whereas the out-of-sample error was 1.68 "useful" votes when tested on the testing dataset. I.e. the model can predict number of "useful" votes with a deviation of nearly **1.68** votes. A random sample of the predicted number of votes versus the actual votes is shown below.

```
##                      review_id Actual.Votes Predicted.Votes
## 1362941 -NAFwv6T1d7kWkC9vPJsvA            3               4
## 1375182 d2iHH5X6u_NNQh5e74dWYw            0               1
## 1493708 _Fy_4jkJlnxo678Qh9JtpQ            1               2
## 821286  UNW6sZMc3IJlULuG6J8X2A            0               1
## 1497763 WVaheCoNqwJgBA2l5u4kZA            1               0
## 214522  5Ji2qhXxRKI26K_vK_2Bvg            2               2
## 213548  5GDxwCUdR3AUTcMDxXS8FA            0               0
```

**Model-B**: predicts the *usefulness* category for a review. It's based on **Random Forest** algorithm, 5-fold cv, 16 predictors, and the outcome variable (*Useful_Category*). In-sample and out-of-sample errors were **21%**. I.e. the model can predict the usefulness category with **79%** accuracy. A random sample of predicted versus actual "Usefulness Category" is shown below.

```
##                      review_id Actual.Category Predicted.Category
## 1362941 -NAFwv6T1d7kWkC9vPJsvA Slightly Useful    Slightly Useful
## 1375182 d2iHH5X6u_NNQh5e74dWYw      Not Useful         Not Useful
## 1493708 _Fy_4jkJlnxo678Qh9JtpQ      Not Useful         Not Useful
## 821286  UNW6sZMc3IJlULuG6J8X2A      Not Useful         Not Useful
## 1497763 WVaheCoNqwJgBA2l5u4kZA      Not Useful         Not Useful
## 214522  5Ji2qhXxRKI26K_vK_2Bvg Slightly Useful         Not Useful
## 213548  5GDxwCUdR3AUTcMDxXS8FA      Not Useful         Not Useful
```

## Discussion

In conclusion, applying both prediction models (Mode-A and Model-B) to the Yelp data enables us to predict, **with an acceptable accuracy level**, the extent of usefulness of a user's review for a business by predicting the number of "useful" votes and the "usefulness" category. For **Model-A**, the prediction error (**1.68 "useful" votes**) is very reasonable in predicting the number of useful votes. Similarly, in **Model-B** the prediction accuracy (**79%**) looks also reasonable in classifying the usefulness of a user's review. However, both models can be considered for further fine-tuning and improvement.

Both models can have positive implications on the yelp.com, yelpers, and businesses as their prediction capacity helps exploit potential useful reviews as soon as they are posted in an effort to be proactive in improving businesses and to provide quicker recommendations for potential customers.