# COEN 281 PATTERN RECOGNITION AND DATA MINING

# IMPROVISED YELP RATING SYSTEM

Pooja Gandhi
Rama Gupta
Richa Sharma
SasiKumar Ravichandran

## Introduction

Yelp, is likely to be the first application on most of the people's mind when they go through to decide about restaurants with respect to different cuisines. Most of the users go through the reviews and star ratings to decide about a place.
Yelp reviews provides a very good way to select a restaurant. As, we know review provides rating feature for a restaurant ranging through 1 star to 5 stars and also contains reviews for each business. Yelp recently came up with the idea of having votes for each review. Thus, to promote a review, Users now can vote a review as "Useful", "Funny" or "Cool". This way a user can first check if the review is useful or not and then can read through it. Moreover, other users can give "Useful" votes to reviews they find useful. However, a user may read only a limited number of reviews and may not even read the complete review due to time constraint.

## Problem Statement 1

Although votes for the review made easier for a user to go through a list of reviews but one of the major flaw is that not all the reviews are voted. Also, users hardly scroll down and read the old reviews and vote for them.

## Problem Statement 2

Sometimes it is hard to trust a reviewer or the person who rates. Some people might just rate randomly and that can affect the overall rating of the restaurant.

## Problem Statement 3

Authentic rating: Indian people would be able to rate and review an Indian restaurant better than people from other parts of the world as they know how a particular dish tastes like in India. People who are not from Indian origin may not like the cuisine and might give low rating to the restaurant which can impact the overall rating of a particular restaurant.

## Data Set

Data from Yelp Dataset Challenge was used. The size of the entire dataset was ~2.5GB

The Yelp dataset consists of 3 files:

**yelp_academic_dataset_business.json**

**yelp_academic_dataset_review.json**

**Yelp_academic_dataset_user.json**

## Proposed Solution

## Solution to Problem statement 1

Find an automatic way of predicting whether a review is useful or not. Using Natural Language processing to do text analysis and to predict how useful a particular review is. Considering the classification algorithms, most of NLP algorithms we used are classification models, and they include Logistic Regression, Naive Bayes, CART which is a model based on decision trees, Maximum Entropy in Decision Trees.
Another very well-known model in NLP is the **Bag of Words** model. It is a model used to preprocess the texts to classify before fitting the classification algorithms on the observations containing the texts.

## Solution to Problem statement 2

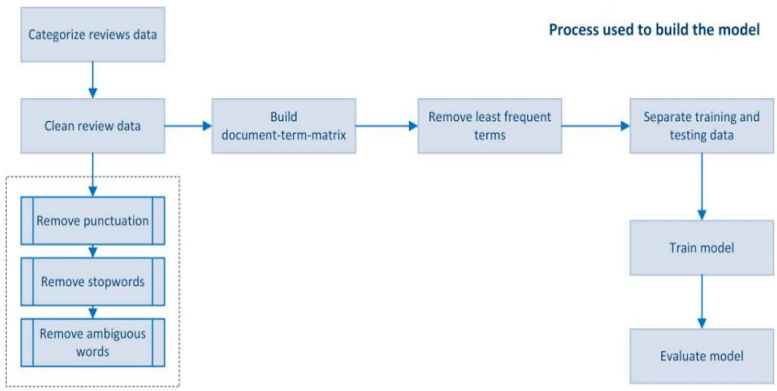
Create a new rating system which gives more weight to those users who have reviewed more restaurants of the same cuisine. For example, if someone has visited and reviewed all 12 restaurants of same cuisine, then their opinion should be given significantly more weight.

## Solution to Problem statement 3

Select only the Indian users and the reviews given by them and calculate the new rating. We have done this for indian people, it can be similarly done for other cuisines as well

# Problem 1 (Implementation)

## Approach




## Converting JSON to CSV

- **Step1**: Read all the json files , Convert it to Csv by removing all the special characters and encoded characters and Generate headers for each column.
- **Step2** : Join 3 csv files [ user.csv , review and business.Csv] to single csv file [ review_user_business.csv] . The join has been done by taking two columns User_id and business id.

### Data Collection

We performed analysis on data and found the following results. The below histogram shows the distribution of votes per review. Here we can see that most of the review have not got voted as useful. But there are sufficient voted reviews which we can use to train our model.

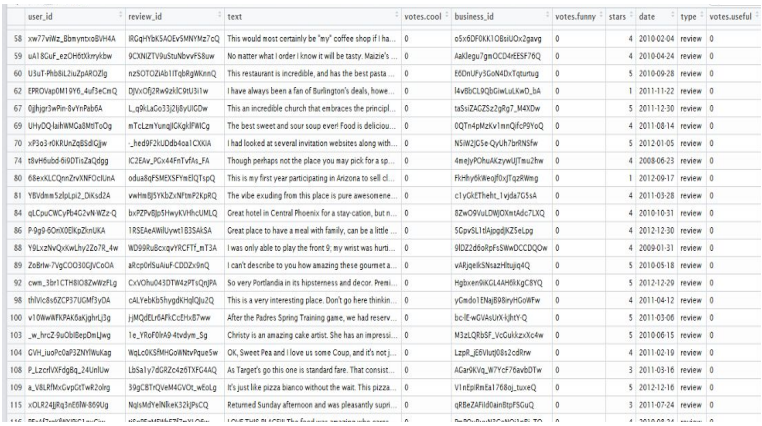
- Select only those reviews that were voted either as “Useful”, “Funny” or “Cool”. Since , most of the reviews didn’t receive any votes we could not use them to train our models. This was one of the challenges that we faced. A zero in the useful review can be due to two reasons, either the review was not useful or it has not been voted at all. But if we consider the reviews that has vote for at least one of the fields ie. “Useful”, “Funny” or “Cool” then a zero in useful votes would mean that the vote is not

useful.

- We looped through the data set to get the reviews that had at least 1 in the votes("Useful", "Funny" or "Cool") section. This was our training data.

*test_data<-read_data[(read_data$votes.useful<1&read_data$votes.cool<1&read_data$votes.funny<1 ) ,]*

*read_data<-read_data[(read_data$votes.useful>0|read_data$votes.cool>0|read_data$votes.funny>0 ) ,]*



| | user_id | review_id | text | votes.cool | business_id | votes.funny | stars | date | type | votes.useful |
|---|---|---|---|---|---|---|---|---|---|---|
| 58 | xw77vWz_Bbmyntxo8vHAA | 8GqHYbK5AOEv5MNYMz7cQ | This would most certainly be "my" coffee shop if I ha... | 0 | o5x6OFOKK1OBsIUOx2gavg | 0 | 4 | 2010-02-04 | review | 0 |
| 59 | uAI 8CuF_ez OH0tXkmYkbw | 9CXNIZTV9uStuNbvvFS8uw | No matter what I order I know it will be tasty. Maizie's ... | 0 | AaKlegu7gmOCD4tEESF76Q | 0 | 4 | 2010-04-24 | review | 0 |
| 60 | U3uT-PhbBiL2iuZpAROZlg | nzSOTOZ4b1ITqbRgWKnnQ | This restaurant is incredible, and has the best pasta ... | 0 | E6OnUFy3GoN4DxTqturtug | 0 | 5 | 2010-09-28 | review | 0 |
| 62 | EPROVapDM19Y6_4uf3eCmQ | QjVxOfj2Rw9zklC0tU3i1w | I have always been a fan of Burlington's deals, howe... | 0 | I4vBbCL0QbGiwLuLKwD_bA | 0 | 1 | 2011-11-22 | review | 0 |
| 67 | 0jjhjgr3wPm-9vYnPabSA | L_q9kLaGo33j2ljByUlGDw | This an incredible church that embraces the principl... | 0 | taSsiZACZSz2gRg7_M4XDw | 0 | 5 | 2011-12-30 | review | 0 |
| 69 | UHyOQ-laihWMGa6MtSToOg | mTcLzmYumqlGKqkIFWICg | The best sweet and sour soup ever! Food is deliciou... | 0 | 0QTn4pMzKv1nnQifcP9YoQ | 0 | 4 | 2011-08-14 | review | 0 |
| 70 | xF3o3-tGkRUhZqBSdlCjjw | _jhed9F2kUDdb4oa1CKXiA | I had looked at several invitation websites along with... | 0 | N5iW2jCSe-QyUh7brRN5fw | 0 | 5 | 2012-01-05 | review | 0 |
| 74 | t8vH6ubd-6i0DTIsZaQdgg | IC2EAv_PCx44FnTvfAs_FA | Though perhaps not the place you may pick for a sp... | 0 | 4mejyPOhuAKzywUjTmu2hw | 0 | 4 | 2008-06-23 | review | 0 |
| 80 | 68exXLCQnnZrvXNFOcIUnA | oduia8qFSMEXSFYmElQTspQ | This is my first year participating in Arizona to sell cl... | 0 | FkHhy6kWeqfIOxJTqzRWng | 0 | 1 | 2012-09-17 | review | 0 |
| 81 | YBVdmm5dpLpi2_DKsdJ2A | vwhtmBj5YkbZxNFtmPZkpRQ | The vibe exuding from this place is pure awesomene... | 0 | c1yGkEThe ht_1vjda7G5sA | 0 | 4 | 2011-03-28 | review | 0 |
| 84 | qLCpuCWCyPb4G2vWWZz-Q | bxPZPvBjbSHwyKVHhcUMLQ | Great hotel in Central Phoenix for a stay-cation, but n... | 0 | 8ZwO9VuLDWjOXmtAdc7LXQ | 0 | 4 | 2010-10-31 | review | 0 |
| 86 | P-9g9-6OnX0EKpZtnUKA | 1RSEAeAWiUywt1B3SAiSA | Great place to have a meal with family, can be a little ... | 0 | 5Cpv5L1fAjpqdjKZ5eLpg | 0 | 4 | 2012-12-30 | review | 0 |
| 88 | Y9LxzNvQxKwLhyZ2o78_4w | WD99RuBcxqvYRCFTf_mT3A | I was only able to play the front 9; my wrist was hurti... | 0 | 9fDZ2d6oRpFsSWwOCCDQOw | 0 | 4 | 2009-01-31 | review | 0 |
| 89 | ZoBrIw-7VgCOD30GjVCoOA | a8cp0rlSuAiuF-CDDZv9mQ | I can't describe to you how amazing these gourmet a... | 0 | vARjgeIkSNsazHtujq4Q | 0 | 5 | 2010-05-18 | review | 0 |
| 92 | cwm_3br1CTH8IOBZwWzFLg | CxVOhu043DTW4z9TsQnJPA | So very Portlandia in its hipsterness and decor. Premi... | 0 | HgbxenGiKGL4AH9IkKgCBYQ | 0 | 5 | 2012-12-29 | review | 0 |
| 98 | thIVic8s6ZCP37UGMf3yDA | cALYebKb5hygdKHqlQju2Q | This is a very interesting place. Don't go here thinkin... | 0 | yGmdo1ENajB9BiryHGoWFw | 0 | 4 | 2011-04-12 | review | 0 |
| 100 | vl0WwAfVKPAk6AKjghrLJ3g | jJMQdELr6AFkCcEHxB7ww | After the Padres Spring Training game, we had reserv... | 0 | bciE-wGVAsUrXkjhtY-Q | 0 | 5 | 2011-03-06 | review | 0 |
| 103 | _w_hrcZ-9uObiBepDmLjwg | 1e_YRoF0lrA9-4tvdym_Sg | Christy is an amazing cake artist. She has an impressi... | 0 | M3zLQRbSF_VcGukkzxXc4w | 0 | 5 | 2010-06-15 | review | 0 |
| 104 | GVH_juoPc0aP3ZNYlWuKag | WqLc0KSfMHGoWNzvPqueSw | OK, Sweet Pea and I love us some Coup, and it's not j... | 0 | LzpR_jE6Vlutj08s2cdRrw | 0 | 4 | 2011-02-19 | review | 0 |
| 108 | P_LzcrfiVXFdgBq_24UnIUw | Lb5a1y7dGRZc426TXFG4AQ | As Target's go this one is standard fare. That consist... | 0 | AGar9kVq_W7YcF76avbDTw | 0 | 3 | 2011-03-16 | review | 0 |
| 109 | a_V8LRfMxGvpCtTwR2oIng | 59gCBTrQVeM4GVOt_wEoLg | It's just like pizza bianco without the wait. This pizza... | 0 | V1nEpiRmEa1768oj_tuxeQ | 0 | 5 | 2012-12-16 | review | 0 |
| 115 | xOLR24JJRq3nE6lN-869Ug | NqlsMdYeINlkeK32KjPsCQ | Returned Sunday afternoon and was pleasantly supri... | 0 | qRBeZAFIid0ainBtpF5GuQ | 0 | 3 | 2011-07-24 | review | 0 |

We divided this data into training and test to verify the accuracy of our results.

## Data Cleaning and preprocessing

Text mining is done in R using the the text mining framework provided by the tm package. It has methods to import data, create corpus, preprocess data and create term-document matrix.

Though, this was a time consuming step but it really helped to achieve high quality analysis.

- **Removing punctuation:**

  Computers treat punctuation and special characters as words.

  We used the following to remove the punctuation

  *corpus = tm_map(corpus, removePunctuation)*

- **Removing numbers:**

  *tm_map(corpus, removeNumbers)*

- **Converting to lowercase:**

  Then we changed the complete text to lowercase so that each word appears exactly the same whenever it appears.

  *corpus = tm_map(corpus,content_transformer(tolower))*

- **Removing whitespace:**

  There are white spaces in the reviews at many places. We got rid of all the white spaces

  *corpus = tm_map(corpus, stripWhitespace)*

- **Removing any URL:**

  *corpus = tm_map(corpus, removeURL)*

- **Removing "stopwords"**

  Stop words are the common words that are uninteresting and do not help in analytics.In each review there are a lot of these stop words.Commonly found stop words are a, an , the, if etc. these words can confuse our analytics if not removed.

  *corpus = tm_map(corpus, removeWords,stopwords())*

- **Removing particular words**

  We removed the words that appeared in the output but were not of any value for analysis.

- **Stemming document**

  One word can have many forms but have same meaning. Eg love, loving, loved.
  So we removed common word endings. This process is called as stemming of data.
  For stemming we need R library called SnowballC.
  *install.packages("SnowballC")*
  *library(SnowballC)*
  *corpus = tm_map(corpus, stemDocument)*

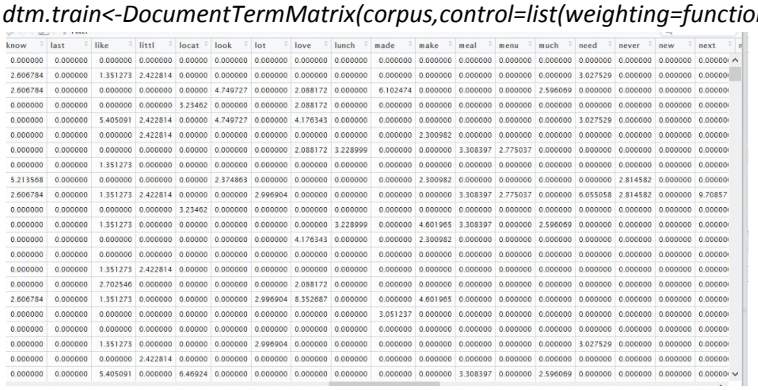
- **Convert to plain text document**

  After the data has been preprocessed it is converted to plain text document.
  *corpus<- tm_map(corpus, PlainTextDocument)*

## Creating bag of words model

Next we create a **Document Term Matrix (DTM)**. DTM reflects the number of times each word appears in the corpus.
Rows correspond to reviews and columns correspond to the words found in reviews.

*dtm.train<-DocumentTermMatrix(corpus,control=list(weighting=function(x)weightTfIdf(x,normalize = FALSE)))*



| know | last | like | littl | locat | look | lot | love | lunch | made | make | meal | menu | much | need | never | new | next |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 2.606784 | 0.000000 | 1.351273 | 2.422814 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.027529 | 0.000000 | 0.000000 | 0.00000 |
| 2.606784 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 4.749727 | 0.000000 | 2.088172 | 0.000000 | 6.102474 | 0.000000 | 0.000000 | 0.000000 | 2.596069 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.23462 | 0.000000 | 0.000000 | 2.088172 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 5.405091 | 2.422814 | 0.00000 | 4.749727 | 0.000000 | 4.176343 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.027529 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 2.422814 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.300982 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 2.088172 | 3.228999 | 0.000000 | 0.000000 | 3.308397 | 2.775037 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 1.351273 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 5.213568 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 2.374865 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.300982 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.814582 | 0.000000 | 0.00000 |
| 2.606784 | 0.000000 | 1.351273 | 2.422814 | 0.00000 | 0.000000 | 2.996904 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.308397 | 2.775037 | 0.000000 | 6.055058 | 2.814582 | 0.000000 | 9.70857 |
| 0.000000 | 0.000000 | 0.000000 | 3.23462 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 1.351273 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 3.228999 | 0.000000 | 4.601965 | 3.308397 | 0.000000 | 2.596069 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 4.176343 | 0.000000 | 0.000000 | 2.300982 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 1.351273 | 2.422814 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 2.702546 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 2.088172 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 2.606784 | 0.000000 | 1.351273 | 0.000000 | 0.00000 | 0.000000 | 2.996904 | 8.352687 | 0.000000 | 0.000000 | 4.601965 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.051237 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 1.351273 | 0.000000 | 0.00000 | 0.000000 | 2.996904 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.027529 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 0.000000 | 2.422814 | 0.00000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |
| 0.000000 | 0.000000 | 5.405091 | 0.000000 | 6.46924 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 3.308397 | 0.000000 | 2.596069 | 0.000000 | 0.000000 | 0.000000 | 0.00000 |

- **Removing sparse terms**

  Then we saw that there are many terms that do not appear very often in the reviews, we call this as "Sparse terms"
  *dtm.train<-removeSparseTerms(dtm.train, 0.85)*
  We will use this bag of words to train our models

- **Word cloud**

  Word cloud is a text mining method that highlight the most frequently used keywords in a paragraph of texts.
  We have used this method to get the most frequent words in our reviews
  A text mining package (tm) and wordcloud generator package (wordcloud) are available in R to generate word Cloud.


- **Converting to dataFrame**
  Then we convert the document term matrix to a data frame as our models take data frames as input.

- **Append Data Frame**
  Then for each row in the frame we append a value of useful votes. Hence, we append a column to the frame as vores.useful
  *train_dataSet = as.data.frame(as.matrix(dtm.train))*
  *train_dataSet$votes = read_data$votes.useful*

## Modelling

## Random Forest Method

Random Forest operates by constructing a multitude of trees at training time and outputting the class and providing the mean prediction.
Figure shows the result of Random Forest method for our dataset. We ran for different values of n ie. the number of trees. The accuracy improved a little as we increased the number of trees from till 40 but after that there was no significant improvement in the accuracy. We also observed an improvement in accuracy as we increased the amount of training data to 100,000.

**Accuracy = 0.52**

```
ctrl = trainControl(method = "cv", number = 10)
forestmodel <- train(factor(votes) ~ .,  data=training_set,method ="rf", ntree=100, trControl = ctrl,  type= c("raw"))
forestPredict <- predict(forestmodel, newdata = test_set[-246], type= c("raw"))
votes.useful = forestPredict
submit <- data.frame(votes.useful)
cm = table(test_set[,246], forestPredict )
plot(forestmodel)
```

```
> cm
   forestPredict
      0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
  0   0  44   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  1   3 241  18   7   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  2   0 103  19   3   0   1   0   0   0   0   0   0   0   0   0   0   0   0
  3   0  55   4   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  4   0  34   4   1   1   0   0   0   0   0   0   0   0   0   0   0   0   0
  5   0  16   1   1   0   1   0   0   0   0   0   0   0   0   0   0   0   0
  6   0  12   2   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  7   0   4   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  8   0   4   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  9   0   6   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  10  0   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  11  0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  13  0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  15  0   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  17  0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

## Naive Bayes Model

- Simple classification method based on Bayes rule.
- It depends on very simple representation of document.
- Classify the training dataset on the basis of votes whether it is useful or not.
- Figure shows the confusion matrix which predicts the votes on our training dataset.

We encoded the feature vector to restrict the values of votes in the range of 1-5

**Accuracy = 0.42**

```
# naiveBayes
classifier = naiveBayes(x= training_set[-165], y = factor(training_set$votes , levels = c(0,1,2,3,4,5)))
votes_pred = predict(classifier, newdata= factor(test_set[-165]))
submit <- data.frame(votes_pred)
cm = table(test_set[,165], votes_pred )
#plot(classifier)
```

| | votes_pred | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| 0 | 5 | 35 | 0 | 4 | 1 | 1 |
| 1 | 18 | 190 | 14 | 17 | 14 | 16 |
| 2 | 5 | 81 | 5 | 19 | 3 | 13 |
| 3 | 1 | 36 | 7 | 6 | 2 | 8 |
| 4 | 2 | 23 | 3 | 8 | 0 | 4 |
| 5 | 1 | 8 | 3 | 4 | 1 | 2 |

## KNN

This classification algorithm stores available cases and classify new cases based on Euclidean distance. We implemented this model for different values of k and we found that it worked better as we increased the value of k. It worked best for k=15

**Accuracy : 0.538**

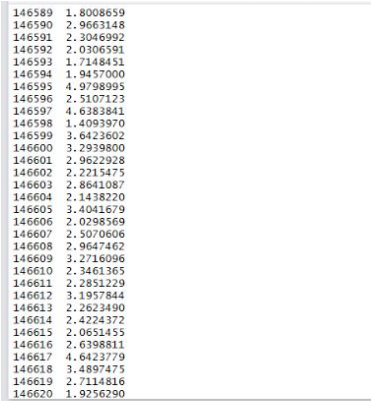
```
library(class)
knn = knn(train = training_set[, -104],
          test = test_set[,-104],
          cl = training_set[, 104],
          k = 15)

cm = table(test_set[,104], knn )
plot(knn)
```

| | knn | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 0 | 0 | 44 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 255 | 12 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 115 | 9 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 1 | 54 | 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 1 | 34 | 3 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 13 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 13 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 5 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## Multiple Linear Regression

Used multiple linear regression to train the model and then converted float data to integer values using case statements. Anything > = 5 is considered as 5 ie. very useful.

```
# linear regression
regressor=glm(formula = factor(votes) ~ ., data = training_set,  family = binomial)
votes_pred = predict(regressor, newdata= test_set[-104])
submit <- data.frame(votes_pred)
submit1= data.frame(floor(votes_pred))
submit2 = data.frame(votes_pred)
floor(submit2)
cm = table(test_set[,104],floor(submit2 ))
```



```
146589  1.8008659
146590  2.9663148
146591  2.3046992
146592  2.0306591
146593  1.7148451
146594  1.9457000
146595  4.9798995
146596  2.5107123
146597  4.6383841
146598  1.4093970
146599  3.6423602
146600  3.2939800
146601  2.9622928
146602  2.2215475
146603  2.8641087
146604  2.1438220
146605  3.4041679
146606  2.0298569
146607  2.5070606
146608  2.9647462
146609  3.2716096
146610  2.3461365
146611  2.2851229
146612  3.1957844
146613  2.2623490
146614  2.4224372
146615  2.0651455
146616  2.6398811
146617  4.6423779
146618  3.4897475
146619  2.7114816
146620  1.9256290
```

These values are converted to integers using floor function.

## Packages used

- ❏ RandomForest- to implement random Forest algorithm
- ❏ RTextTools- includes 9 algos for ensemble classification(Random Forest, decision Trees)
- ❏ e1071- naive bayes, SVM
- ❏ tm- is a framework for text mining
- ❏ Word cloud-helps to analyse texts and quickly visualise the keywords as word cloud
- ❏ Caret- is a set of functions that attempts to streamline the process for creating predictive models.
- ❏ wordcloud
- ❏ SnowballC
- ❏ rpart

## Problem 2 (Implementation):

In Yelp,there are millions of users rating restaurants on a daily basis. Rating and prioritizing the reviews is very important to provide accurate information about a particular restaurant. Keeping this in mind, we have come up with a solution to characterize users depending on the number of reviews they have registered for a specific restaurant which in turn decides the overall rating of the restaurant. We have given more weightage to users who have given more reviews to restaurants rather than users who have reviewed less. The proposed algorithm is :

### Step 1

Consider only those reviewers who have given more number of reviews than other reviewers on the same cuisine and then compute the new star rating by using the following approach.

**1**.Create a new attribute is_Indian which have two output values. If Categories="Indian", Value is "True ", Else "False".

**2**. Find total number of reviews given by each user for Indian Cuisine. This will help us to know the frequent reviewers who likes to rate usually on yelp.

**3**. Calculate weighted stars by Multiplying the stars and total reviews given by each user.

**4.** Formulate the following to get the new rating pattern for each Indian restaurant.

**5**. Calculate the new stars rating by dividing the sum of weighted stars by total reviews given by different users for particular restaurant in a particular city.

### Result 1

| usiness_id | stars | username | city | business_name | categories | review_count | avg_stars | is_indian |
|---|---|---|---|---|---|---|---|---|
| NCgQMF6fkh2q1Is92X8wg | 1 | Alisha | Las Vegas | Montana Meat Company | ['Steakhouses', 'Restaurants'] | 49 | 3.0 | FALSE |
| vnADjcjlE489gUKxVC3cg | 1 | alison | Henderson | Roma Grill | ['Restaurants', 'Italian'] | 19 | 3.5 | FALSE |
| RBUU1y4yjrK0SPAd8z0-w | 1 | Alison | Phoenix | Indian Delhi Palace | ['Pakistani', 'Indian', 'Buffets', 'Restaurants'] | 128 | 3.5 | TRUE |
| A9aCw2kTgN8p2SLdm2mDA | 1 | Alison | Phoenix | Taste of India | ['Pakistani', 'Indian', 'Restaurants'] | 24 | 2.0 | TRUE |
| r9rQKypHgC_J1Afufxzlw | 1 | Alison | Scottsdale | Wildflower Bread Company | ['Bakeries', 'Food', 'Sandwiches', 'Restaurants'] | 109 | 4.0 | FALSE |
| pGv6ES2hzPCtCUQgipwXw | 1 | Alison | Madison | New Orleans Take-Out | ['Sandwiches', 'Cajun/Creole', 'Seafood', 'Restaurants'] | 43 | 4.0 | FALSE |
| detLT5gBDPneWZUfyf7_A | 1 | Alison | Madison | Nick's Restaurant | ['Greek', 'Bars', 'Mediterranean', 'Nightlife', 'Restaurants'] | 48 | 3.5 | FALSE |
| IZ259hZ6Yww3C0o07MgUQ | 1 | Alison | Madison | Fazoli's | ['Pizza', 'Restaurants'] | 8 | 3.0 | FALSE |
| s5nBRc-6BqFT_1-ZaVVug | 1 | Alison | Madison | New Orleans Take-Out | ['Cajun/Creole', 'Restaurants'] | 48 | 4.0 | FALSE |
| IAXWfskIsATjpui2C8HilQ | 1 | Alison | Madison | Topper's Pizza | ['Caterers', 'Pizza', 'Event Planning & Services', 'Resta... | 12 | 3.0 | FALSE |
| pSBHeixIP2YxLy4L1f-HA | 1 | Alison | Madison | Ella's Deli | ['Food', 'Ice Cream & Frozen Yogurt', 'Delis', 'American ... | 112 | 3.0 | FALSE |
| sBet6HtxE0x41cUYl8vgQ | 1 | Alison | Madison | The Kollege Klub | ['Nightlife', 'Bars', 'American (New)', 'Sports Bars', 'Rest... | 19 | 2.5 | FALSE |
| GCetjJwnPU5bqoNxFM3sw | 1 | Alison | Madison | Opus Lounge | ['Tapas/Small Plates', 'Bars', 'Nightlife', 'Lounges', 'Rest... | 49 | 4.0 | FALSE |
| io5aGXkf0aEeGCcSpWkRg | 1 | Alison | Madison | Tex Tubb's Taco Palace | ['Mexican', 'Tex-Mex', 'Restaurants'] | 114 | 3.5 | FALSE |
| oub5SnLqMCjm0uN0y1nA | 1 | Alison | Madison | Pasqual's | ['Mexican', 'Tex-Mex', 'Restaurants'] | 55 | 3.0 | FALSE |
| xOS4S91Vq9pnyo2Up1Zjg | 1 | Alison | Madison | Yummy Buffet | ['Buffets', 'Chinese', 'Restaurants'] | 4 | 1.5 | FALSE |
| TuDep1-bkTot7PQU3_ceg | 1 | Alison | Madison | Tilted Kilt Pub & Eatery | ['Pubs', 'Bars', 'Fast Food', 'Nightlife', 'Restaurants'] | 7 | 2.5 | FALSE |
| 5XliadakX5VWw3mdcsHug | 1 | Alison | Madison | Mad Dogs Eatery | ['Hot Dogs', 'Restaurants'] | 17 | 3.0 | FALSE |
| 8cTq3_LI2E8_EvD0dCsow | 1 | Alison | Madison | Daisy Cafe and Cupcakery | ['Bakeries', 'Food', 'Breakfast & Brunch', 'Restaurants'] | 137 | 4.0 | FALSE |
| 42TdBYPhu5YL0S1MTx2A | 1 | Alison | Madison | Fat Sandwich Company | ['Sandwiches', 'Restaurants'] | 15 | 2.0 | FALSE |
| bmcCE_cLq4WO8ZMKlmaLw | 1 | Alison | Las Vegas | Hash House A Go Go | ['Breakfast & Brunch', 'American (New)', 'Restaurants'] | 1759 | 4.0 | FALSE |

## Step 2:

**(1)** Find total number of reviews given by each user for Indian Cuisine. This will help us to know the frequent reviewers who likes to rate usually on yelp.

## Result 2

| | user_id | tot_rev |
|---|---|---|
| 5775 | z6qJ5ZJz979Mpv7eatTIHA | 29 |
| 5774 | jolzw_aUiNvBTuGoytrH7g | 26 |
| 5773 | M6oU3OBf_E6gqlfkLGlStQ | 25 |
| 5772 | GkWuTgewni9bzPM4HUCO-g | 23 |
| 5771 | AjDQWpCanz_g3rHuBQ5B1g | 20 |
| 5770 | 8GPJ4VUxbKMhrT3V3Ql8CQ | 17 |
| 5766 | 4CgusCZkipvUhvBZrRD46w | 16 |
| 5767 | KmZ9l-NoWdJJ3CG_euUJRA | 16 |
| 5768 | SIfJLNMv7vBwo-fSipxNgg | 16 |
| 5769 | wx12_24dFiL1Pc0H_PygLw | 16 |
| 5764 | 0bNXP9quoJEgyVZu9ipGgQ | 15 |

Total Indian Category reviews= 5777

## Step 3

**(1)** Calculate weighted stars by Multiplying the stars and total reviews given by each user.
**(2)** Formulate the following to get the new rating pattern for each Indian restaurant.

**(3)** Calculate the new stars rating by dividing the sum of weighted stars by total reviews given by different users for particular restaurant in a particular city.

**(4)** Calculate the new rating results.

### Result 3

New Rating Star pattern is depicted by new attribute and earlier one by avg. This helps in calculating more accurate rating for a restaurant by giving more importance to reviewers who reviews more.

Filter

| | city | business_name | avg_stars | cnt | avg | new | dif |
|---|---|---|---|---|---|---|---|
| 1 | Avondale | India Garden | 4.0 | 94 | 4.202128 | 4.160584 | -0.041543718 |
| 2 | Cambridge | Fusion | 4.5 | 5 | 4.600000 | 4.980198 | 0.380198020 |
| 3 | Cave Creek | Indian Village | 4.0 | 19 | 4.000000 | 3.957265 | -0.042735043 |
| 4 | Chandler | Bay Leaf Cafe | 3.5 | 40 | 3.575000 | 3.812000 | 0.237000000 |
| 5 | Chandler | Biryani Kitchens | 3.5 | 3 | 3.666667 | 4.500000 | 0.833333333 |
| 6 | Chandler | Cafe Krishna | 3.5 | 56 | 3.428571 | 3.630890 | 0.202318624 |
| 7 | Chandler | Copper Kettle | 3.0 | 30 | 3.300000 | 3.666667 | 0.366666667 |
| 8 | Chandler | Curry House | 2.5 | 9 | 2.666667 | 2.230769 | -0.435897436 |
| 9 | Chandler | India Gate | 3.5 | 49 | 3.448980 | 3.076763 | -0.372216106 |
| 10 | Chandler | Indian Paradise | 4.5 | 7 | 4.428571 | 4.473684 | 0.045112782 |
| 11 | Chandler | Indus Village | 4.0 | 20 | 4.150000 | 3.932551 | -0.217448680 |

## Problem 3 (Implementation):

Authenticity and ethnicity of the reviewers is more important to rate a specific restaurant. In this problem, we have tried to rate the Indian restaurants by giving more preference to Indian reviewers to others as people with an ethnic background are dependable and are accurate when rating their cuisines. This assumption may not be hundred percent but it will definitely has an edge compared to non-indian reviewers.
Create an individual rating for users based on their names that relates them to their Ethnicity.

Consider Indian Cuisine and Calculate new rating on the basis of stars given by Indian reviewers for a restaurant.

### Step 1

**(1)** Filter Indian names from username column according to the unique Indian name list and result is stored in reviewer_indian_name

### Result 1

Unique India Name List:

| Indian_Names |
|---|
| Aayush |
| Abhi |
| Abhijeet |
| Abhijit |
| Asmita |
| Asodha |
| Atreyee |
| Atul |
| Bhavana |
| Bhavik |
| Bhrata |
| Bhujang |
| Sheetal |
| Shikha |
| Shilpa |
| Srikanth |
| Vinod |
| Vinoth |
| Yogesh |
| Yuvaraj |

**(2)** Compare username with the unique indian name list and generate reviewer_indian_name as ‘True’ or ‘False’

| | username | city | business_name | categories | review_count | avg_stars | is_indian | reviewer_indian_name |
|---|---|---|---|---|---|---|---|---|
| 3 | Karen | Edinburgh | Bombay Spice | ['Pakistani', 'Indian', 'Restaurants'] | 6 | 3.5 | TRUE | FALSE |
| 3 | Jennifer | Edinburgh | Bombay Spice | ['Pakistani', 'Indian', 'Restaurants'] | 6 | 3.5 | TRUE | FALSE |
| 4 | Mihali | Edinburgh | The Prince of India | ['Indian', 'Restaurants'] | 4 | 3.5 | TRUE | FALSE |
| 4 | Hannah | Edinburgh | The Prince of India | ['Indian', 'Restaurants'] | 4 | 3.5 | TRUE | FALSE |
| 4 | Beth | Edinburgh | The Prince of India | ['Indian', 'Restaurants'] | 4 | 3.5 | TRUE | FALSE |
| 5 | Raja | Madison | Maharani Restaurant | ['Pakistani', 'Indian', 'Restaurants'] | 97 | 3.5 | TRUE | TRUE |
| 4 | Shanda | Madison | Maharani Restaurant | ['Pakistani', 'Indian', 'Restaurants'] | 97 | 3.5 | TRUE | TRUE |
| 3 | Abhijit | Madison | Maharani Restaurant | ['Pakistani', 'Indian', 'Restaurants'] | 97 | 3.5 | TRUE | TRUE |
| 3 | Ami | Madison | Maharani Restaurant | ['Pakistani', 'Indian', 'Restaurants'] | 97 | 3.5 | TRUE | TRUE |
| 4 | Deepak | Madison | Maharani Restaurant | ['Pakistani', 'Indian', 'Restaurants'] | 97 | 3.5 | TRUE | TRUE |
| 5 | Ankita | Madison | Maharani Restaurant | ['Pakistani', 'Indian', 'Restaurants'] | 97 | 3.5 | TRUE | TRUE |

Showing 3,346 to 3,358 of 8,230 entries

**Indian Reviewers: 723**
**Non-Indian Reviewers: 7507**

## Step 2

**(1)** Calculate Indian-Stars based on the following - Account on stars given by Indian users for Indian cuisine in a restaurant.

**(2)** Multiply the stars with reviewer_indian_name value.

**(3)** Generate New Immigrant Rating.

**(4)** Re-compute the Indian-Stars.

## Result 2

Indian-Stars for new rating system have been computed . Through these stars, It becomes more easier to choose an Indian restaurant.

| | city | business_name | avg_stars | count | nin | pin | avg | ias | dif |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Edinburgh | Indian Cavalry Club | 3.0 | 5 | 1 | 0.20000000 | 3.000000 | 1.000000 | -2.000000000 |
| 2 | Edinburgh | Lancers Brasserie | 4.0 | 7 | 1 | 0.14285714 | 3.857143 | 1.000000 | -2.857142857 |
| 3 | Edinburgh | Mezbaan South Indian Restaurant | 3.5 | 11 | 1 | 0.09090909 | 3.727273 | 1.000000 | -2.727272727 |
| 4 | Las Vegas | Pyaar India Restaurant | 3.0 | 21 | 1 | 0.04761905 | 2.952381 | 1.000000 | -1.952380952 |
| 5 | Las Vegas | Sai India Curry | 3.0 | 11 | 3 | 0.27272727 | 3.000000 | 1.000000 | -2.000000000 |
| 6 | Tempe | Pasand | 3.0 | 34 | 2 | 0.05882353 | 3.176471 | 1.500000 | -1.676470588 |
| 7 | Tempe | Delhi Palace Cuisine of India | 3.5 | 93 | 6 | 0.06451613 | 3.763441 | 1.666667 | -2.096774194 |
| 8 | Edinburgh | Guchhi Indian Seafood and Bar | 3.5 | 11 | 1 | 0.09090909 | 3.545455 | 2.000000 | -1.545454545 |
| 9 | Las Vegas | Bollywood Grill Indian Cusine | 2.5 | 39 | 2 | 0.05128205 | 2.743590 | 2.000000 | -0.743589744 |
| 10 | Madison | Taj Indian Restaurant | 4.0 | 36 | 3 | 0.08333333 | 3.944444 | 2.000000 | -1.944444444 |
| 11 | Mesa | India's Grill | 4.0 | 28 | 2 | 0.07142857 | 3.750000 | 2.000000 | -1.750000000 |
| 12 | Phoenix | India Palace | 3.5 | 104 | 3 | 0.02884615 | 3.778846 | 2.000000 | -1.778846154 |

Showing 1 to 12 of 199 entries

## Challenges

Since we all were new to both R and Python we had to learn the very basics of both the languages.
Dataset was too large, our machines crashed several times and it took several hours to train the models
Most of the values were 1 so difficult to train and improve the model

## Learnings

- Learnt the Natural Language processing, text analysis both in R and Python (Although implemented in R)
- Learnt different classification and regression models.
- Referenced lecture notes and took Udemy courses to understand the implementation of the algorithms
- Understood which model should be used in which case, what their hyper parameters are and how the performance can be improved.

- We not only learned the algorithms theoretically but also implemented them and got a hands on experience on them.

## Failures

- We tried to run our models by varying the hyperParameters. We ran Random forest for different values of n ie. the number of trees. We ran it for n = 10, 20, 50, 100, 150 but the accuracy was not improving. One of the main reason for the low accuracy is that most of the reviews that we have have a voting of 1. So our model learns the same thing and hence predicts the values as 1.
- We also improved the Document term matrix by removing the words that we felt were not important and can hamper the result. That improved the accuracy very little but not too significant.
- We tried to run SVM to classify. After seeing the result we could actually relate to what we read in class. That SVM classifies the data only in two classes

## Conclusion

Used Text classification to predict the usefulness of the reviews and compared various models. We found that Random Forest gave us the best results.
We also tried to tweak the Yelp Rating System and recalculated the rating of the businesses to give user an additional rating option based on trustworthiness and ethnicity.

## Division of labour

Pooja and Rama :
Conversion of Json to CSV, table joins, cleaning of data, understanding different models
Steps involved to implement Text Classification, creating DTM, word cloud, Implemented different models
Richa and Sasi : Task 1 : Research work and Implemented solution for problem statement 3– Authenticity Rating work on problem statement 2- weighted user rating system

## References

http://www.cs.cmu.edu/~knigam/papers/multinomial-aaaiws98.pdf

https://www.yelp.com/dataset_challenge

http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

http://sebastianraschka.com/Articles/2014_naive_bayes_1.html

http://cs229.stanford.edu/proj2014/Xinyue%20Liu,%20Michel%20Schoemaker,%20Nan%20Zhang,Predicting%20Usefulness%20of%20Yelp%20Reviews.pdf

https://www.kaggle.com/c/yelp-recruiting/forums/t/4135/the-purpose-of-predicting-usefulness-and-way-to-evalute-the-prediction?forumMessageId=21849

https://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/031.pdf
https://rpubs.com/mohammedkb/YelpProject