Yelp Dataset: Rating prediction through feature engineering

Xiaoyang Zeng*
University of California - San Diego
9500 Gilman Dr.
San Diego, CA
xiz460@ucsd.edu

Zhaoyi Hou*
University of California - San Diego
9500 Gilman Dr.
San Diego, CA
z9hou@ucsd.edu

Haoshu Qin*
University of California - San Diego
9500 Gilman Dr.
San Diego, CA
h5qin@ucsd.ed

Zhuoyuan Ren*
University of California - San Diego
7675 Palmilla Drive
San diego, CA
zhr003@ucsd.edu

1. INTRODUCTION

For this project, we would like to explore the Yelp dataset. First we will provide an overall view of the dataset. Then we will predict the ratings of restaurants based on certain features.

2. DATASET

2.1 Dataset overview (sizes and meanings)

Originally we have three datasets:

"business" (business.json): contains business data including location data, attributes, and categories. There are **192,609** rows of business data.

"review" (review.json): contains full review text data including the user_id that wrote the review and the business_id the review is written for. We only choose **120,000** rows of review data for the project.

"user" (user.json): including the user's friend mapping and all the metadata associated with the user. There are **1,637,138** tows of user data.

We merged them together so that we have a comprehensive dataset. After merging the three tables with inner join now we have a combined dataset of **106,804** rows where each row holds all information about a piece of review, the business the review is about, and the user who wrote the review. There are **85,092** unique user_ids and **10,583** unique business_ids.

2.2 Rating Distribution

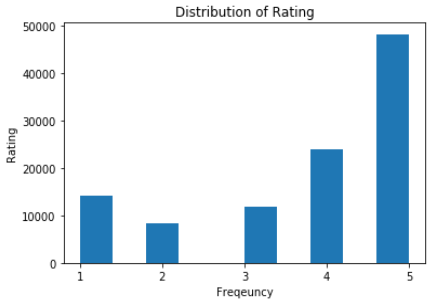

Figure 2.2.1 Rating Distribution

The above bar chart is the distribution of all ratings in our dataset. As we can see below, the rating of "1" frequently appears, while the rating of "2" and "3" occurs fairly infrequently. That aligns with our real-life experience where we either love a restaurant or a service department, or we find it so bad that we want to write a Yelp review to let everyone else know it (See figure 2.2.1) .

2.3 Other interesting findings

2.3.1 Useful Vote and Rating

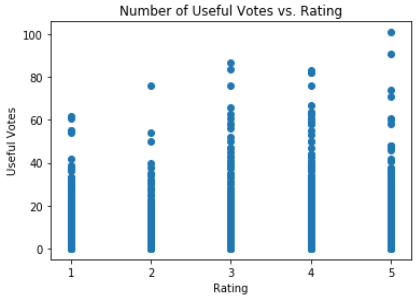

Figure 2.3.1 Useful Vote and Rating

The scatterplot above shows a positive relationship between the number of 'useful' upvotes a review receives and the rating the review gives, suggesting that a review is likely to give a higher rating if we already know the review has many 'useful' votes. It is expected since a higher rating review should suggest a business is nice. And therefore the nice business should attract more viewers to open the interface, see and vote 'useful' on the reviews. Also, reviews giving out extremely low ratings could be too objective or personal, providing little useful opinions to viewers (See figure 2.3.1). .

2.3.2 # of reviews wrote and Rating

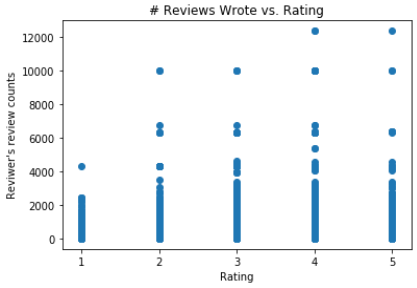

Figure 2.3.2 Number of review written

We draw a scatter plot to show the relationship between the amount of reviews the reviews has written and the rating the review gives. It is quite obvious that there is a positive relationship between the two variants suggesting that a reviewer that has written more reviews tend to give higher ratings. This makes sense since there are many people who only wrote one or a few reviews out of anger and complaints after they had a very negative experience at some places (See figure 2.3.2).

2.3.3 Day of the week and Rating

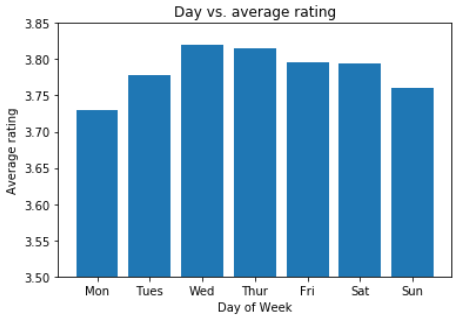

Figure 2.3.3 Day of the week and Rating

Which day of the week a review was given to a business could be potentially correlated with the rating the review gives. We draw a bar plot to compare the average rating all the review give on a certain day of the week side by side. Although the lowest average rating and highest one only a small difference of about 0.1, we still have some interesting findings; The average rating on Monday is the lowest. That could be due to the fact that reviewers start to work or go to school, so they have more negative moods than other days of the week (See figure 2.3.3).

.

2.3.4 Length of review and Rating

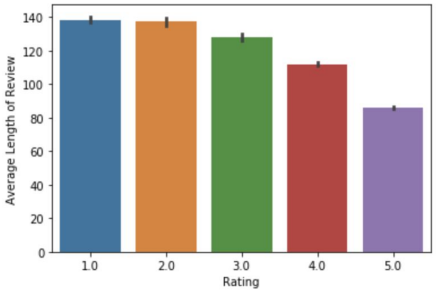

Figure 2.3.4 Review length and Rating

Another interesting feature we notice is that, on average, longer reviews tend to have lower rating. That is actually reasonable: when we are mad about a restaurant or service department, we are likely to leave a long review to describe how badly those people did (See figure 2.3.4).

2.3.5 Category and Rating

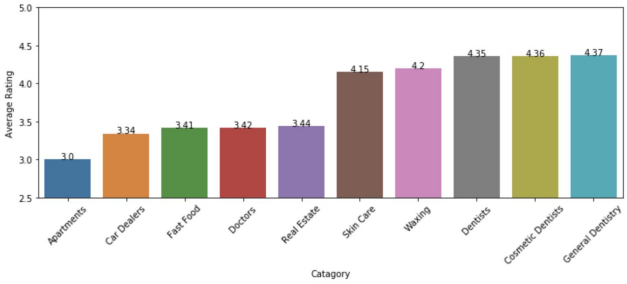

Figure 2.3.5 Category and Rating

In popular categories (those categories with more than 2,000 business), the 5 top-rated and 5 worst-rated categories are shown above, together with their average ratings (See figure 2.3.5).

2.3.6 Word Clouds

Figure 2.3.6.1 Word cloud for 1-star reviews

Figure 2.3.6.2 Wordcloud for 5-star reviews

Above we created two word clouds for review texts of 1-star reviews and 5-star reviews (See figure 2.3.6.1 and 2.3.6.2). We find it really interesting that 5-star reviews are mainly filled with adjective words such as "amazing", "good", "great" and "best". Reviewers that had great experience at a business would use these adjectives to give compliments. In contrast, 1-star reviews are mainly filled with verbs such as "told", "said", "asked", "went", from which we can assume bad reviews are highly associated with communications between the businesses and the customers. Also we can see many 1-star ratings have to do with "time", "service" and "order".

3. PREDICTIVE TASK

Yelp, as a review website, allows users to rate on different businesses they have been to and write review texts on them. One important feature Yelp has is generating recommendations to users based on their past behaviors. Rating, as a sentimental factor, could help on building such recommendation models. By first predicting user rating, we could then recommend to the users businesses that they might give them high star rating. To establish a non-trivial model, the first task we need to solve is to predict star rating as precise as possible.

In the lecture, we have already learned how to use latent factor model to predict rating based on user and business self-variations. (e.g. how much a certain user rating above the global mean of all user rating). Such model completely withdraws the effects of other features dataset provided, such as categories, opening days, review texts, etc.

Therefore, in this paper, we will try to solve the rating prediction from different approaches. With the features we have collected in previous section, We could build a more sophisticated model that outperforms the latent factor model we have in class.

We suppose to solve the prediction task from two approaches: predicting based on the features in review texts and based on the explicit features that dataset obtain. (time stamps, categories). After collecting features, we will train 3 different models for each approaches and evaluate on two metrics.

3.1 Preprocessing and Feature design

3.1.1 Preprocessing

Since the Yelp allows users to write review texts in any format they want, the first thing we need to do is to "clean" the useless characters and mis-spelling words. First, we lower all the capitalized words and remove the punctuation to obtain a complete text corpus. Then, we also replace some mis-spelling and abbreviated words into correct format, such as replace "i'm" to "i am". We also remove the stop words, like "a", "an", etc. Since we are modelling the prediction model with Bag of Words, such stop words are meaningless to the overall corpus meaning.

3.1.2 Feature Design

We selected the following features from the merged dataset for our predictive task.

Category: As mentioned in the dataset section, we identified **155** categories which have at least **1000** businesses on file. Then we did one-hot encoding on those categories and each one of them is a feature for prediction.

Length of words: We created a column about the number of words written on a piece of review text.

Number of "Useful" votes: A column representing how many "Useful" votes a review has received.

Number of Reviews written: A column representing how many reviews a specific user has written in total.

Day of week: We convert the date that a review was written on into which day of the week that was. So we one-hot encoded seven new columns indicating if a review was written on a specific day of the week or *not.*

Popular Words: We choose ten most popular words from all 1-star reviews texts and ten from all 5-star reviews texts. Then we one-hot encoded them into ten new columns, each of which indicating if the word appears in the review text.

3.1.2.1 Tokenization (Bag of Words)

Though the text data contains rich information, common supervised machine learning models could not fit directly on the string corpus. Therefore, we first need to extract the feature vectors from strings by assigning each word with unique id (so called Token) and further represent strings in the occurrence of tokens they have.

After all, we will result in a fixed size feature matrix with one review text per row and one token per column. Such matrix is recognizable to the machine learning model.

3.1.2.2 TD-IDF

From the last part, we obtain a tokenized feature matrix based on the occurence of each words in the review texts. But there is a problem addressing on the normalization of features. For instance, common words, such as "her", "them", etc, generally obtain less information than some other obviously more meaningful but rarer words, such as "good", "bad", etc. So we need a weighting technique to adjust the imbalanced frequency of such words.

The technique we use here is TD-IDF (Term Frequency - Inverse Document Frequency). Beyond the feature matrix we obtain in the last part, TD-IDF will assign less weight to common words and higher weight to those rare but meaningful terms.

After both tokenizing and normalization, we finish our feature extraction from review text.

3.2 Machine Learning Models

We use three supervised machine learning models to fit our feature vectors.

3.2.1 Logistic Regression

Logistic Regression is a probability model that predicts proper label by modelling the conditional probability function P(feature|rating), where feature is the feature vector we extract from dataset and rating is the rating label range from 1 to 5. Given a new feature vector, this probability function is computed for all values of rating label, and the rating value corresponding to the highest probability is output as the final class label (star rating) for this feature vector.

Since Logistic Regression is calculated based on the probability and maximize the likelihood of possible label, it is appropriate to be used to make a prediction that minimize the MSE.

3.2.2 Linear Support Vector Classification

Different from logistic, linear SVC is trying to obtain a decision boundary that optimizes the number of wrong labels we predicted. In general, Linear SVC will generate a different hyperplane than that of Logistic Regression. In some cases, it could be a better model.

3.2.3 SVC (Support Vector Machine)

Similar to the linear SVC, SVC is another SVM with a different kernel function. It services as a heuristic implementation here to compare with the results of linear SVC.

Since the finding an exact hyperplane with rbf kernel is impractical for a large dataset, we use SGD optimized classifier instead.

3.3 Baselines

As is described above, one of the goals of our paper is to find a model incorporating data features that could outperform the latent factor we studied in class. Therefore, latent factor models in class could be appropriate baselines to be compared to.

We build two latent factor models: one only considering the user and business self-variation on rating. (see Figure 3.3.1)

$$f(u,i) = \alpha + \beta_u + \beta_i$$

Figure 3.3.1 Simple latent factor model

α represents the global rating average while β*u* and β*i* represent the effect of certain user and business on the rating.

The other more complicated model further counts the incorporating effect of user and business preference. (see Figure 3.3.2)

$$f(u,i) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i$$

Figure 3.3.2 Complicated latent factor model

In addition to the simple factor model, the latter one considers the user and business effect dependently by generating the approximate rating based on certain user and business rating vectors. In the equation, it is defined as γ*u* and γ*i* where γ*u* is the estimation of user *u*'s rating vector on all business and vice versa. Such estimation can be computed based on the dimensionality reduction technique such as matrix factorization.

Both baselines are implemented based on the contents from lecture workbook. We will compare the performance of baselines and our models based on the metric we defined below.

3.4 Performance metric & Implementation details

After shuffling the existing dataset, We use 90% of it for training and 10% left for testing. For each model, we apply 5-fold grid search cross validation to tune hyperparameters.

Since we are doing a rating prediction task, the metrics we use here are Mean Squared Error and Mean Absolute Error. (see Figure 3.5.1 and Figure 3.5.2). Mean Squared Error is the square of error rate after normalization and Mean Absolute Error is defined as the absolute value of the average error between predictions and true labels.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

Figure 3.5.1 Mean Squared Error

$$\mathrm{ME} = \frac{\sum_{i=1}^{n} y_i - x_i}{n}$$

Figure 3.5.2 Mean Error

The perfect model should have both MSE and MAE zero. So our goal is to find a model with the least MSE and MAE on the test set.

4. MODEL EXPLORATION

As we described above, we will use 5-fold cross validation on each machine learning model for validation test. Then, we will use grid search to find the best model given different hyperparameters exhaustively. To save time on tuning hyperparameters, we only use mean squared error as the metric when we select model.

4.1 Model Selection

4.1.1 Models based on features

We explored three different models: Logistic Regression, Linear SVC and SGD Classification. In order to tune to the best parameter, we used GridSearchCV from Sci-kit Learn.

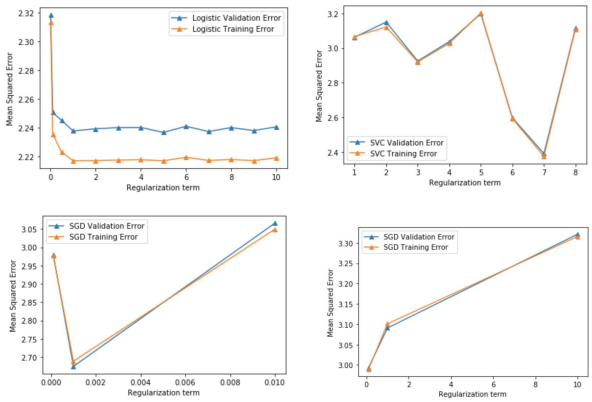

Figure 4.1.1 Model performances

As shown above in figure 4.1.1, y-axis represents the average MSE of 5-fold validation and train set of the model on different regularization terms.

For Logistic Regression, we use grid search on 13 different C values (Inverse of regularization strength; Larger C represents lower strength); for linear SVC, we use grid search on 10 different C values. As we can see from the figure, the SGD Classifier has the worst validation performance, achieving a lowest **MSE 2.67**. The second one is the Linear SVC. Its best **MSE is 2.66**. Finally, the Logistic Regression outperforms the other two models with best **MSE 2.22**.

In fact, this performance is not really good, given that Yelp rating is a number from 1 to 5, with step 1. If our prediction of a rating vary by around 2 points, that is basically "random" guessing. **Although the best model outperforms the baseline, it's not really good.** Therefore, we turned to review text for more features, as presented below.

4.1.2 Models based on review texts

As stated above, we explore three different models: Logistic Regression, Linear SVC and SGD Classification.. To optimize the performance, we perform 5-fold cross validation on each model while trying different regularization hyperparameters.

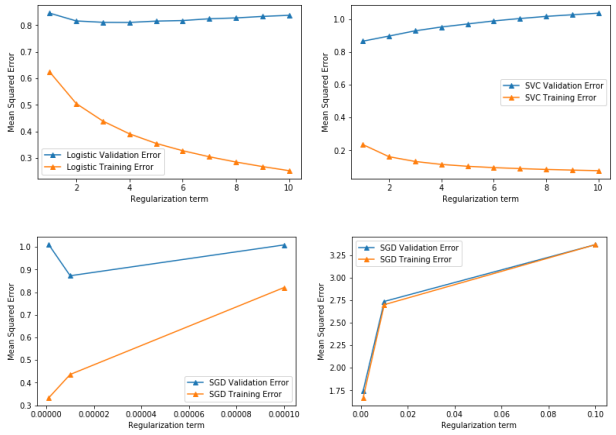

Figure 4.1.2 Model performances

From figure 4.1.2, y-axis represents the average MSE of 5-fold validation and train set of the model on different regularization terms. For linear SVC and Logistic Regression, we use grid search on 10 different C values (Inverse of regularization strength; Larger C represents lower strength). In the figure, the SGD Classifier obviously has the worst validation performance, achieving a lowest **MSE 0.87**. The second best one is the Linear SVC. Its best **MSE is 0.85**. Finally, the Logistic Regression outperforms the other two models with best **MSE 0.80**.

While we lower the regularization term, we find that we are experiencing overfitting issues. From the figure, we could observe that while the training error keeps decreasing with the decrement of the regularization, the validation error is growing. So the best model we choose is the local minima of validation error. By using grid search, we can avoid the problem of overfitting.

As is mentioned above, the best model we obtain is the Logistic Regression model with C = 4.

4.1.3 Strength and weakness of our models

The first model made use of features other than review text, which introduces more possibility for more possibilities, for instance, predicting whether or not a user would like a certain store before getting their review text. However, this approach didn't achieve very good performance for real applications (given the high MSE and MAE, as mentioned above).

The second model only involves review text, which gives far more accurate predictions than the first one. An MSE of as low as 0.8 is actually good enough for meaningful prediction. However, it cannot be used directly as people seldom need a prediction on whether or not they would like a store after they've already visited that place. This can be useful for larger scale inference and further prediction though.

5. RELATED WORK

We selected our dataset from the Yelp database. There are many papers that has been done before on topics regarding Yelp dataset. Predicting ratings from reviews is one of the most occurring one. We found a paper from **UNC Charlotte and UCI** focused on predicting ratings from text-reviews alone.

In this paper, the authors used cross validation for the data selection process. For the baseline design, the authors first established baseline by picking the first K common words from each review. The authors then choose K with the least errors.

For feature engineering 1, the authors first used 'Part-of-Speech' to identify the position of words in a text. For feature engineering 2, the authors selected adjectives as the words to use as the K most frequent word because it is reasonable to assume adjectives as the most descriptive and informative words. Finally, the authors used four learning methods including linear regression, support vector regression, support vector regression-n, decision tree and compared their performance from RMSE.

The state-of-the-art methods used in studying text-based analysis currently include tf-idf, bag of words, part-of-speech, and K most common words.

5.1 Comparison to our own works

Comparing to our assignment, we used latent factor models as our baseline; the literature used K most frequent word as the baseline. We received an **MSE of 2.4** compared to **MSE of 0.36** in the literature.

For feature engineering part, we added some basic features, including category of business, length of review, number of "useful" vote, number of reviews written, day of the week, popular words in 1-star and 5-star reviews, and bag-of-word. However, the paper included some more in-depth feature engineering, including 'Part-of-Speech'.

This gives a more intuitive classifier which achieved a better overall performance: the best model mentioned in this paper achieved a **MSE of 0.36**, much better compared to our best **MSE of 0.80**.

We believe the paper has a better result than us for a couple of reasons.

1. The paper limited the reviews to only about restaurants while we tried to predict ratings for reviews of all types of business on Yelp. This means the paper had an easier and simpler predictive task since that a single category of reviews are more similar to each other in content and styles. Thus it can give the paper a more accurate result.
2. The paper used more complex feature engineering. They used Part-Of-Speech analysis on each review text and selected the top K frequent words amongst all. Also They tried to limit the top K frequent words to be only adjectives. Since they believe reviews mainly use adjectives describe a positive of negative experience at the business.

6. RESULTS & ANALYSIS

After part 4 above, we have done the model selection. We obtain the best models from part 4 and test them on the test set. The following table records the results of different models.

6.1 Report result from the model we build (form of MSE and MAE)

	Model based on selected features		Model based on Review Text	
	MAE	MSE	MAE	MSE
Logistic Regression	**0.89**	**2.27**	**0.41**	**0.76**
linear SVC	**1.35**	**3.56**	**0.43**	**0.81**
SGD Classifier	**1.03**	**2.69**	**0.42**	**0.79**
Baseline 1: Simple latent factor model	**1.30**	**2.40**	**1.30**	**2.46**
Baseline 2: Complete latent factor model	**1.35**	**2.46**	**1.35**	**2.46**

Table 1 Results on test set

From the table (see Table 1) above we could see that Logistic Regression model based on the Review text does the best job on the test set, which has the lowest MAE and MSE (**0.41 & 0.76**). The latent factor baselines do not work well here probably because of the cold start problem. Since we only extract 100,000 dataset over 4,000,000+, there are many new users and businesses in the test set that we didn't see them before. Therefore, the model with rich sentimental feature, such as the raw review text and category features, surely outperform the baselines.

6.2 Significance of our work

The model based on selected features can be used for predicting what rating a user would give to a business place that he/she has never been to. Therefore we can improve our recommendation system where we recommend a list of businesses to a user based on the highest predicted rating that he/she would give.

Model based on review text cannot be used for the purpose since there is no review text for us to create features if a user has not been to a specific business. But we can apply the model when we a user just finished typing a piece of review. We can suggest the star rating the user should give the place based on the review text he/she just finished. This way we can avoid scenario like when a user typed "The food and service are amazing" but he/she accidentally gives a 1-star rating. Our model should predict the user wanted to give a higher rating than 1-star, so we can send a notification where we recommend the user to re-consider the rating just given.

6.3 Best model based on the testing set

Based on the test set, the best model is the Logistic Regression built on review text analysis (using B-O-W and TF-IDF), which can predict the rating of a Yelp review with an error of less than += 0.8

6.4 Visualization of the model parameters

6.4.1 Models based on Non-text features

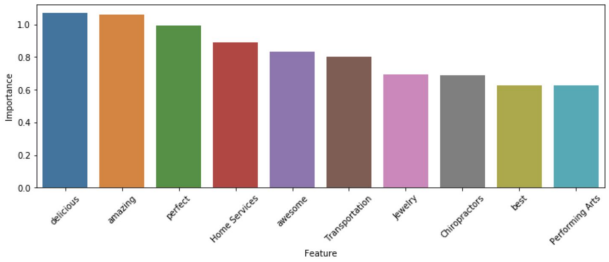

Figure 6.4.1 Visualization of model parameter
(Logistic Regression based on non-text features)

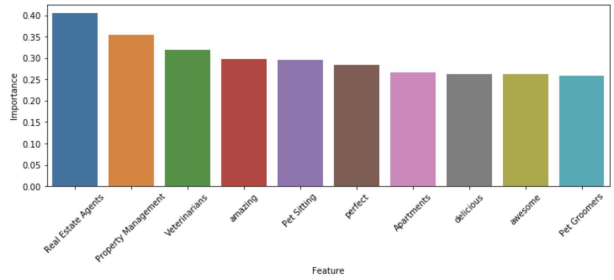

Figure 6.4.2 Visualization of model parameter
(Linear SVC based on non-text features)

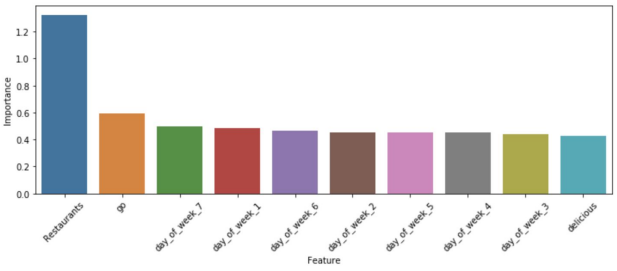

Figure 6.4.3 Visualization of model parameter
(SGD based on non-text features)

As shown above, in our first approach (predicting ratings with limited usage of review text), those features that plays an important role in our model are mostly words with strong emotion (e.g. “delicious”, “perfect”, “amazing”, etc.). This also explains why our second approach, bag-of-word model, is better: it captures the content of the review text more accurately than the first approach.

6.4.2 Models based on review text features

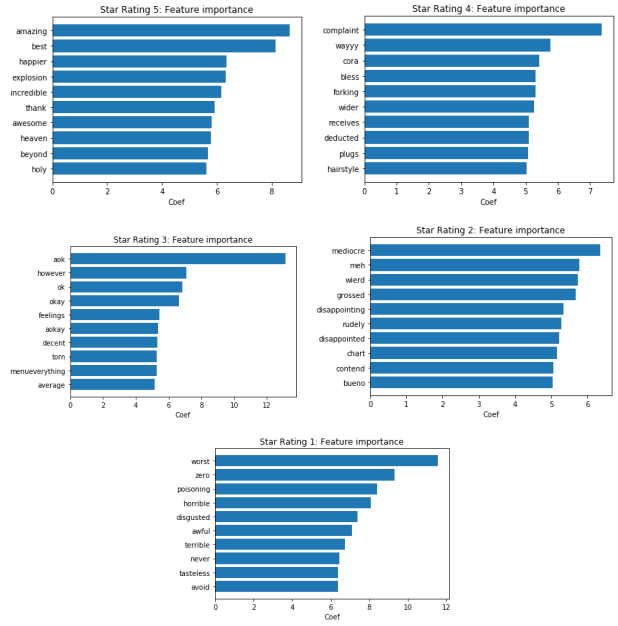

Figure 6.4.2 Feature Importance

In the figure 6.4.2, y-axis represents the string and x-axis express the significance score (coefficient of certain string in Logistic Regression)

As shown in figure 6.4.2, we can see the most important feature words for ratings of different star on the Logistic Regression Model. Most important words in 5-star ratings are “amazing”, “best”, “incredible”, etc. Such words are the adjectives used to describe positive sentiment. Most important words in 1-star ratings are “worst”, “poisoning”, “horrible”, etc. Those words express completely different sentimental feelings than that of the 5-star rating words.

We could also see that the results obtained from analyzing coefficients’ significance score in Logistic Regression draw similar conclusion to the words cloud we have done above. Words do have a significant role in sentimental analysis like rating prediction.

7. CONCLUSION & FUTURE WORKS

In this paper, we perform solutions to the rating prediction task on Yelp dataset. We use 5-fold cross validation to find the best machine learning model. Experimentation and evaluation such model on the test set, we believe that we obtain a non-trivial solution to the rating prediction task and such a solution could be further used to build a recommendation system.

However, we could further improve our model by considering the following adjustments:

1. Though we have done the prediction through two perspectives, there could be a huge improvement on our model if we combine these two aspects together. In such a small dataset, building a more sophisticated model should increase the performance.
2. We could try to use more complicated feature engineering techniques. For instance, in this paper, we only use uni-gram features. We should also capture

bi-gram or tri-gram to obtain more meaningful information.

3. With larger dataset, instead of using traditional techniques, such as TD-IDF, we could then try to use neural network to process text corpus. If we have enough computational power, we could use newly established BERT or other pretrained model to obtain a better result.
4. In addition to current performance metrics, it is better to try our model with other metrics, such as recall, precision and accuracy score.

8. REFERENCES

[1] Mingming, Fan, and Maryam Khademi. 2014. Predicting a Business Star in Yelp from Its Reviews Text Alone. https://arxiv.org/abs/1401.0864v1