Yelp Star Prediction - Team BangBangBang

Benjamin Bang (22879101), Taewook Ha (23176748)
Hyung Jun Kim (23074461), Freddy Xu (26798437)

May 29, 2017

Abstract

This paper summarizes the background, problem, methodology and results of analysis on predicting business stars. In order to make full use of statistical models and machine learning techniques, we utilized the Yelp data basis. The procuess of our analysis was based in exploratory data analysis, feature engineering, and statistical modeling.

1 Exploratory Data Analytics

To predict the star ratings, exploratory data analysis on both business and review data was utilized. Attributes and check-ins were parsed into formats that showed relationships between variables, giving insight into further model selection.

1.1 Business Exploratory Analysis

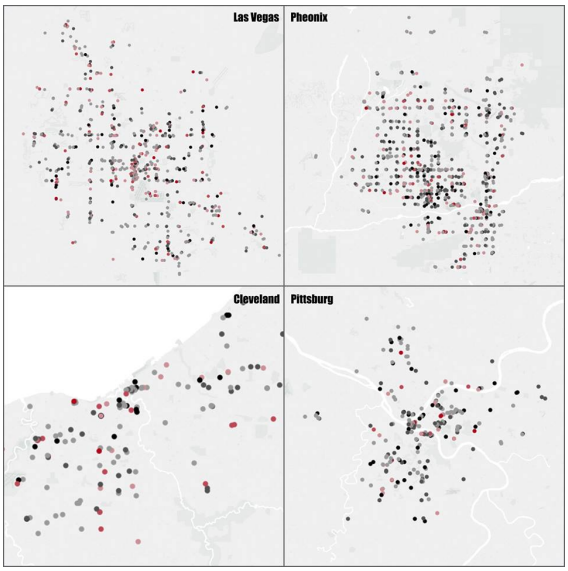


Figure 1 : Star Ratings of Close-by Restaurants

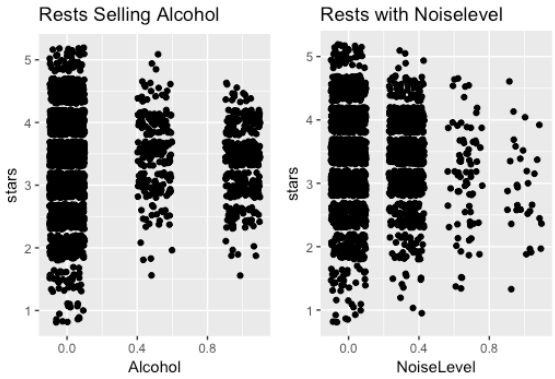


Figure 2 : Star Distribution by Attribute

Figure 1 shows the star ratings of businesses. These businesses are separated by two axes, by the 4 possible food types and by the 4 possible city locations. A natural assumption would be that geographically similar restaurants with the same food typing would share similar star ratings.

Figure 2 shows the distribution of star ratings of businesses separated by 2 example attributes, `Alcohol` and `NoiseLevel`. Some attributes have similar star rating distributions across the different levels, whereas other attributes have significantly differing star rating distributions.

1.2 Review Exploratory Analysis

The review data consists of 116K reviews, by 77K users, for 2.5K businesses. Parsing the review text, and then removing the common stop words (such as "I", "the", "you", "is", "and", "my" etc.), the 5 most common words and their frequency are listed in the first table below. The second table lists the top 5 words that are most commonly associated with businesses of 1, 3, and 5 stars. Identifying these signal words would help distinguish star ratings from one another, an feat that was assisted by clustering the most common words into 5 categories - best, good, average, bad, and worst.

	terms	term_count	doc_count
1	but	98904	57027
2	food	83688	51915
3	not	75441	45219
4	good	71039	46692
5	place	66577	45964

Table 1

5 star	mean	3 star	mean	1 star	mean
addicted	4.72	cents	3.00	disgusted	1.00
upbeat	4.71	ordinary	3.00	cancel	1.13
nevada	4.68	parked	3.00	rudely	1.13
brewing	4.68	notice	3.00	refund	1.16
hooked	4.68	felt	3.00	yelled	1.20

Table 2

1.3 Text Mining through Word2Vec

When dealing with text, sentiment analysis with Word2Vec and Text2Vec packages developed by Mikolov et al [1] is ideal. Word2Vec extends on the neural network for a natural language processing called the Skip-gram model.

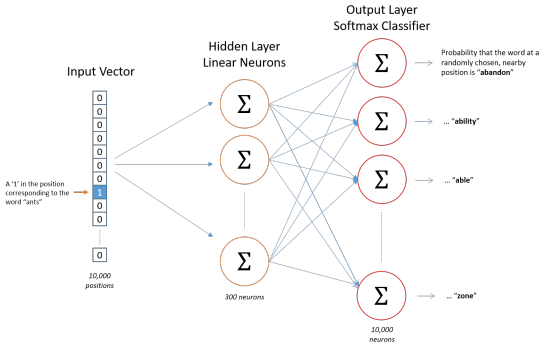


Figure 3 : Word2Vec Neural Net Architecture

It takes a single word in a sentence as a input vector. There is no activation function for hidden layers, but the output layer uses soft-max classifier, resulting in a proximity probability vector of the target word.

However, the task at hand, is not to get the probabilities of words appearing in the n-window of other input words. Instead, the ρ weights of each word used in the hidden layer provide better insight. In this case, ρ was chosen to be 200, a hyperparameter that was cross-validated. The following table showcases an example of 5 words, and their respective ρ vector representation.

[1]Mikolov : http://text2vec.org/index.html

	X1	X2	X3	X4	X5
⟨/s⟩	0.00	0.00	-0.00	-0.00	0.00
the	0.03	0.33	-0.18	0.09	0.09
and	0.00	0.19	-0.18	0.26	0.05
i	0.14	0.11	-0.15	0.17	0.13
a	-0.04	0.08	-0.18	0.03	-0.01

Table 3

As words with similar meaning have similar vector values, clustering words reduced computation complexity and speed. Cosine distances in the similarity matrix through hierarchical clustering is shown in the following figure.

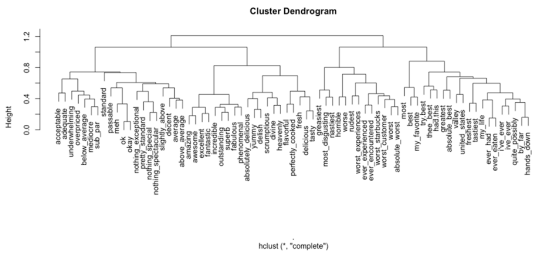


Figure 4: Word2Vec Clustering of Words

Furthermore, we used PCA on the 200 vector space created by Word2Vec. It does a very good job of visualizing the five aforementioned word types ("Best", "Good", "Average", "Bad", "Worst") and plotting their closest words in the dimension of first two PCs. Surprisingly, cosine distance does a great job portraying the distance between two words.

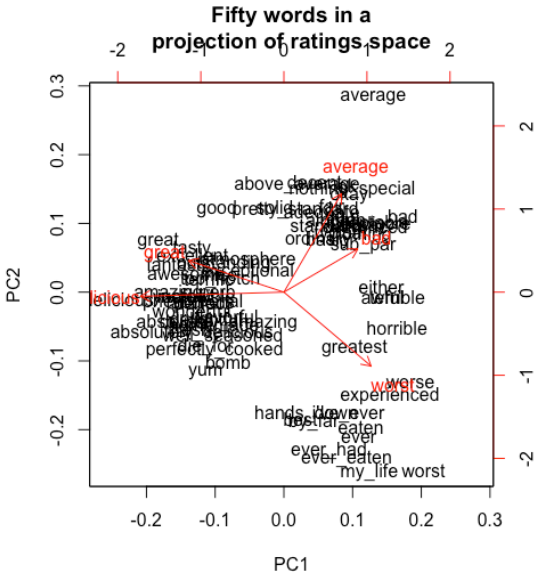


Figure 5: PCA Results and 5 Directions

2 Feature Engineering

Under the assumption mentioned in section 1.1, longitude and latitude values were utilized to find the geo-distances between businesses. Log transformation was applied on different distance boundaries to create distribution of predictors that resembled the original distribution. The figure below shows this transformation on one of the predictors.

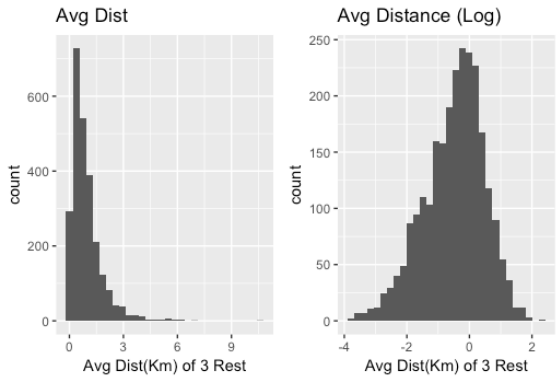


Figure 6 : Log Transformation on Distance

2.1 Pre-processing Review Data

To clean the text, the aforementioned stop words were removed, and stemmed words were grouped together (for example, 'rats' and 'rat' will be treated as the same vector value). With the word bank, multiple n-grams were used to build the vocabulary for the models. Lastly, to reduce sparcity and achieve dimension reduction, hyper parameters such as minimum count threshold of 100-500 and minimum frequency of 0.001 were used to prune the vocabulary. These then were tuned to maximize accuracy.

3 Models

For all of our models, 5-fold cross validation was used to test the results and hold-out set accuracy was used to measure accuracy. This was then compared to the mean-squared root error of the Kaggle competition results. The basic formula is as follows

$$\text{RMSE} = \sqrt{\frac{1}{n} \times \sum_{i=1}^{n} (Y_i - f(X_i))^2}$$

Below are the results of 4 models we ran - Base, Random Forest, GBM on business data, and Multiclass Logistic Regression with L1 penalty.

	CV Acc	Kaggle Acc	CV AUC	Hyper-parameter
Base	0.722	0.717	NA	NA
Conditional Inference Random Forest	0.679	0.704	NA	M-try = 66
GBM	0.688	0.708	NA	$^{\text{trees=100}}_{\text{depth=3}}$
Logistic L1	0.325	0.717	NA	NA

3.1 Base-Model

As starters, a basic classical approach was taken to assess the sentiment of reviews. In order to capture the positivity of the words, a dictionary of words was created, with their values being their respective 'positivity score'. This score was simply the Bayesian average of the review star rating of all the word's appearances. Then, using the dictionary, we predicted the star ratings of new reviews by calculating the mean 'positivity' of words used, weighting on the number of times each word appears in each review.

3.2 Business-Model

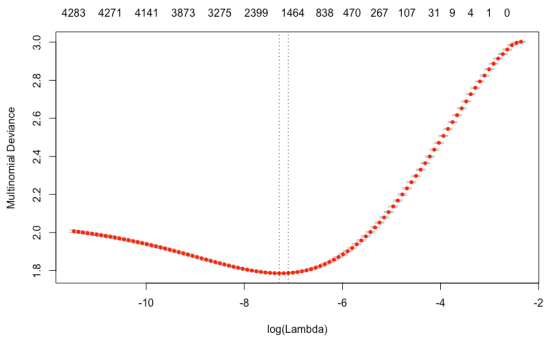
After the base model, we utilized business data to see how model can predict accurately with using business, check-in, and tip dataset. All the attributes of restaurants were converted into one-hot encoding, and check-in values were decomposed into weekday and hour. Distance features [1) average distance of total restaurants and same category restaurants within diameters of 0.1, 0.3, 0.5, 1 (km), and average distance of 3, 5, 10 closest same-category and all restaurants] were also added to see if the extra features describes the target variable well. Total of 117 variables were created and

Gradient Boosting algorithm and Random Forest were applied on this dataset. Random forest regression is an ensemble method that functions by constructing decision trees, producing mean prediction of the individual trees. By producing multiple overfitting trees, random forest corrects it by producing multiple trees. Gradient boosting regression is also an ensemble of weak prediction models. WIth 5k-fold in cross validation, we achieved 0.704 as our lowest rmse with GBM model from the dataset aggregated by business id.

3.3 Logistic Regression with L1 Loss

For multi-class logistic regression, we used Text2Vec developed by Dmitriy Selivanov. Initial analysis was done with 500 minimum term count, 2-gram, 0.001 minimum document proportion. We created a pruned vocabulary, according to such hyper-parameters, vectorized the vocabulary, created document term matrix, transformed it to tfidf(term frequency–inverse document frequency) matrix and used this matrix to fit the model with the stars of each review. Such approach had one of the models used according to the paper by Deniz Demir when he and his team were trying to predict the movie ratings in IMDB site.

We cross-validated using 5 sets and got the accuracy using the hold-out sample. Also, using L1 penalty, the results are as depicted below. Optimal number of parameters is 1801 values out of 5282 vocabulary, with the deviance being 1.78. As this model based on the review data critically outperformed all the other models, we focused on hyper-parameter tuning for this model.



Cross Validation Error for MLR with 2g500

3.4 Hyper-Parameter Tuning

Since cv.glmnet function optimizes λ which is the main parameter for logistic regression with l1 penalty, we looked at even higher level of parameters - the number of grams, and the minimum term counts. Graph below shows the RMSE values pertaining to each of the hyper-parameters. For clarification, 2g500 stands for 2-gram, 500 minimum term count.

	CV RMSE	Kaggle RMSE	Deviance
2g600	0.334	NA	1.79
2g500	0.325	0.521	1.785
2g400	0.33	0.347	1.779
3g600	0.354	NA	1.78
4g400	0.328	0.314	1.767
4g300	0.329	0.325	1.76
5g200	0.334	NA	1.79

Hyper-Parameter Tuning

As we had limited chances of turning the scores in, we have limited number of entries that have the kaggle rmse results. However, we tried out different values of ngrams and minimum number of term counts, and found out that 4-gram, 400 minimum count was the most accurate model out of all the models discussed, which was the last submission we had in Kaggle, which placed 1st place. This model used 1571 terms out of 5282 vocabulary.

4 Further Note: Ensemble

If given more computational power, we would have wanted to combine all the datasets - with features we created using the distance information, review information and their Text2Vec vector values to run Logistic Regression with L1 Loss, SVM, Random Forest, XG Boost, Naive Bayes Classifier, and ensemble the methods by giving the weights to the results based on the accuracy of the models. We tried the general averaging the four results together and tried to cross-validate to get the errors. Unfortunately, simple averaging was worse than both models with RMSE of 0.71, so we had to forgo the basic ensemble method using simple average.

Conclusion

In conclusion, the aim of our analysis lies in finding the optimal model that predicts new restaurants' star ratings as accurate as possible. Utilizing our model, we can estimate star rating of a restaurant based on its reviews, business, check-in, and tip data. Through constructing indepedent models, we compared model predictions both from the scope of our own validation and kaggle submission scores. The model with lowest RMSE score provided insights of the relationship between predictors and star ratings. After a profound assessment of business and review data, we were able to optimize RMSE down to 0.314. Text analysis with 4-ngram without sparse term fully captured sentiment of customer's satisfaction of a restaurant and objective score of a restaurant.

Bibliography

Chris McCormick : http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/
Kun Luo, Meng Li, Shuaiqi Xia, Zhenjie Lin : Prediction of Yelp Star Rating
Junyi Wang : Predicting Yelp Star Ratings Based on Text Analysis of User Reviews
Deniz Demir, Olga Kapralova, Hongze Lai : Predicting IMDB movie ratings using Google Trends
R-Code : https://goo.gl/ZdlE3s