Yelp Review Usefulness Recommendation

Team Member:

Xinyue Zhang (xz2139)

Yiyan Chen (yc2462)

Hanlu Zhang (hz1625)

Ben Zhang (bz957)

1. Introduction

The dataset we are working on is the 'Yelp Dataset Challenge Round 10' dataset [1]. The Yelp dataset provides many opportunities for useful predictions and analysis. The business problem that we are interested in is how to evaluate Yelp reviews. The ultimate goal is that the reviews with the most useful information are placed at the top of the review section. This is important because the reviews that are displayed at the top of the review section are the first to be seen and as such these reviews are the most likely to influence a viewer. Nowadays, it is not uncommon to find reviews on the internet that are not very informative or helpful. Many reviews are simply not informative. They may be too short to be of much use, or may simply be rants that contain no beneficial information. Other times the review may not be related to the business at all. Under such circumstances, it would be helpful if a website were able to filter reviews based on their content and determine the ranking in which they are placed. If a website could successfully implement such a system, it would increase the reliability of the website and in turn draw more users. We believe that supervised learning techniques can help provide such a system to address this problem.

Our goal for this project was to build a predictive model which would give a prediction on whether a newly written Yelp review will be useful based on data gathered from various selected features. After having built several different predictive models, selecting different sets of features and tuning the parameters, we found that Random Forest gave the highest AUC among all the tested models. As such, using this model to analyze Yelp reviews for their usefulness provides a means of addressing the business problem we proposed.

2. Data Cleaning and Feature Engineering

The datasets (review.json, user.json and business.json files) contain information about local businesses in 12 metropolitan areas across 4 countries from July 2004 to July 2017. There are 156,639 business data, 1,183,362 user data and 4,736,897 review data. Each text review is a json object in the 'review.json' file, which specifies 9 attributes such as the useful compliments, review text, stars, etc. Business data is represented in the 'business.json' file as a json object which specifies 101 attributes including its name, location, stars, review count and opening hours among many others. Similarly, user data is represented in 'user.json' file as a json object which specifies 22 attributes including the date of registration, total useful compliments, total photos compliments, number of reviews, average rating stars, number of friends, etc.

While the Yelp dataset has many business categories such as restaurants, shopping, hotels and travel, etc., we chose the data on restaurants inside the United States as our object of focus. We chose to use this data because of its representative power and integrity. Business inside of the United States consists of 73% (115,562) of all businesses in the whole dataset, and restaurants make up almost 34% (39,738) of the 115,562

businesses inside the United States data. Moreover, 49% of the 4,736,897 review data are about restaurants, as shown by Figure 2.1. Therefore, in this project, we restricted ourselves to useful review predictions for restaurants only. Thus, the final trimmed dataset that we used consists of 20,026 restaurants, 47,237 users and 82,847 reviews.

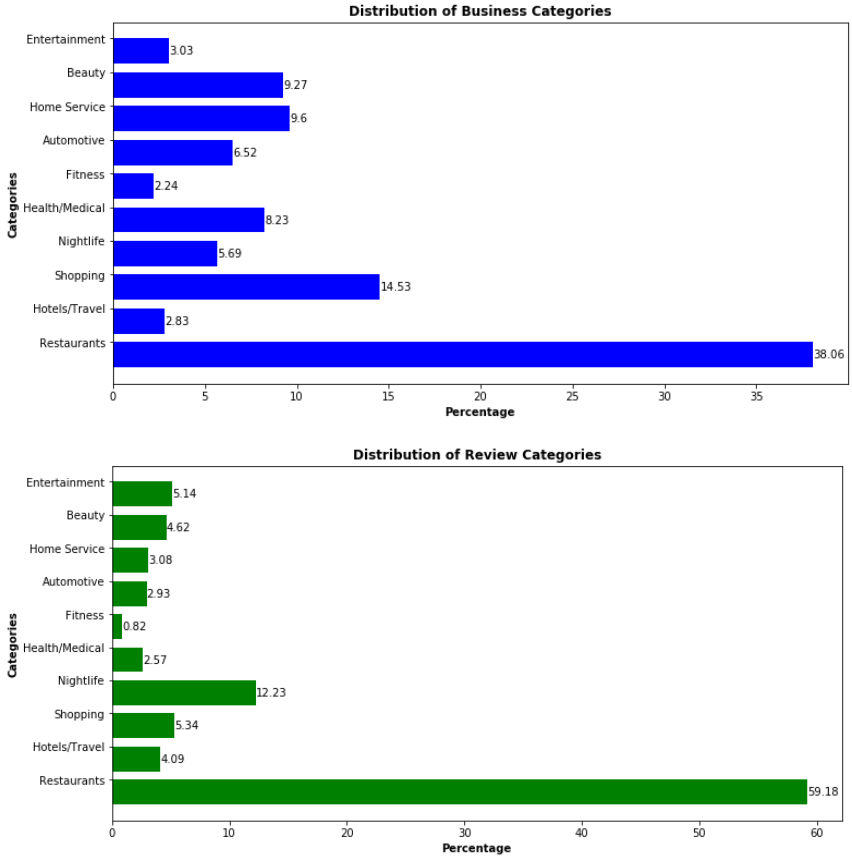


Figure 2.1:
Distribution of business categories among businesses (upper) and reviews (lower)

The dataset was trimmed based on the distribution of the target variable ‘useful’, which gives the number of users that have clicked the useful button under the review posts. We believe that this can act as a direct indication of the usefulness of the review. The distribution of the values of ‘useful’ in the original dataset can be seen in Figure 2.2. The large amount of possibly non-useful reviews brings problems during modeling because of its influence on overall accuracy. In order to address this we merged three datasets: - ‘review’, ‘business’ and ‘user’. Then we weighted the useful value and randomly sampled around 80,000 data points with the relatively balanced useful value. The final dataset consists of around 40,000 data points with ‘useful’ as zero and the other 40,000 data points with ‘useful’ values greater than zero. We can see from Figure 2.2 that the sampled dataset basically preserves the ‘useful’ value distribution of the original dataset except the number of zero values is much smaller.

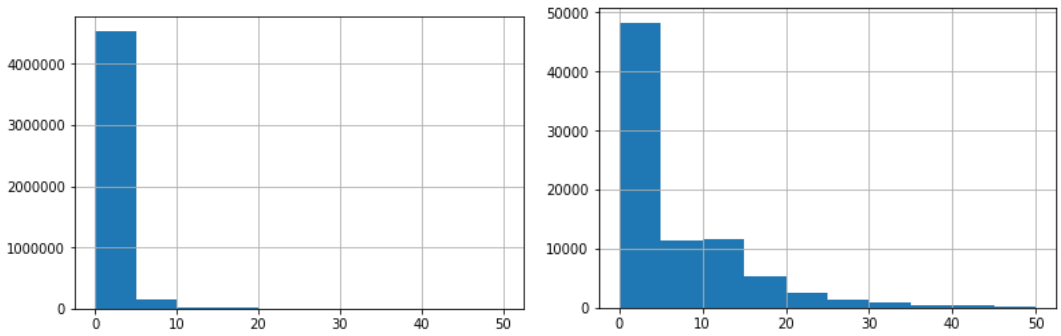


Figure 2.2: Value distribution of useful value before (left) and after (right) sampling

We had an unsuccessful attempt with fitting a regression model, something that will be touched on in the next section, so we decided to convert the target variable into a binary and use a classification target variable for our prediction model. Following the conversion we wanted the useful/useless ration to be large enough to ensure that we could avoid unexpectedly high accuracy in the models due to an unbalanced dataset. We also considered that the age of the post would inevitably affect the amount of users that had an opportunity to click the useful button. In order to reduce the impact that time related features have, we divided the 'useful' variable by the difference in days from when the review was published and when the dataset was given to us (thus providing a new variable denoted as useful/days). Due to the presence of zero values in the denominator we utilized Laplace Smoothing. The mean of the useful/days variable is 0.02746, this value tells us that on average, in 2 months, our new target variable will be greater than 1 (0.02746*60). With this in mind, we converted our final target variable to a binary variable with values of 'useful' divided by 60 (days). As a result, values greater than 1 equal to 1 in the binary and any less than 1 equals to zero. The values greater than 1 in the binary account for 23% of the whole dataset.

After transforming the target variable, we decided to convert all text-based columns to numeric while extracting features and maintaining the most important information. For example, in the business dataset, we converted those columns with data on when each business opens into how many hours a business operates each week. We also converted the values for date review published to how many days the review was displayed on the website before the dataset was given to us. For the categories of business, we converted the categories of cuisines to 8 new binary variables , "American", "Korean", "Italian", "Japanese", "Thai", "Vietnamese", "Chinese" and "Mexican".

For dealing with actual review content, we first recorded the length of the review by word count. We also represented each review as several numerical features based on the frequency of specific words in the review. We recorded the amount of information provided in each review on food quality, food details, quality of service, restaurant atmosphere, cost performance, the convenience of traffic and the waiting time. We created word lists for each topic and counted the frequency of words showing up in the text content of reviews. A detailed word list is attached in Appendix B. Along with that, we also performed text sentiment analysis

using the positive and negative word list provided by Minqing Hu and Bing Liu in their research "Mining and Summarizing Customer Reviews."[2] We created variables indicating scores of being positive or negative for each review.

After that, we dropped all the columns that were not strongly related to the target variable or involved leakage issue. Additionally, we replaced the NaN values with corresponding means of the features, such as price ranges and operating hours. Then we excluded the variables that had high correlation with other variables. A description and notation of variables we used is attached in Appendix A. After finishing all the feature engineering for the model, we normalized the numeric variables for model building.

3. Model Fitting

3.1 Regression

Before working on classification models, we fitted a regression model on the dataset based on the original continuous target variable 'useful' since continuous target variables generally provides more information. However, the linear regression performed poorly for our dataset. Using k-folds cross validation, the mean R square of the baseline linear model was around 0.29. It rose to 0.32 after fitting polynomials and to 0.35 after performing a logarithmic transformation. We also did feature selection, but it didn't significantly improve our result. We believe that this was largely due to the large number of zero target values in our dataset, as can be seen in Figure 2.2 before, and also due to the number of outliers. Using the residual plot in Figure 3.1.1, we can clearly see that outliers are influencing the whole dataset.

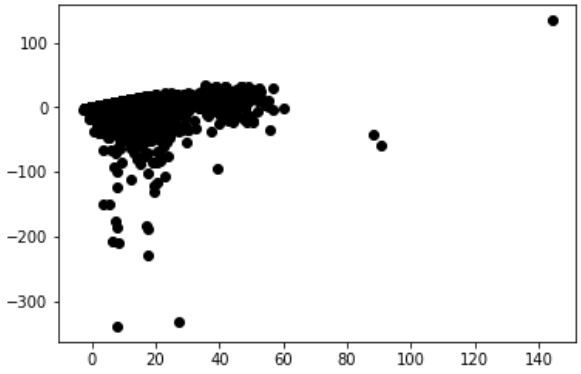


Figure 3.1.1: Residual plot of linear regression with logarithm transformation and polynomial of degree 3

Hence, due to the large percentage of zero target values we have in the dataset and the low performance of the regression model, we decided that converting the target variable into binary and using classification models would be a better option.

3.2 Classification

In terms of classification models, we fitted Logistic Regression, Classification trees, SVM and Random Forest over the dataset. We used AUC ROC metrics among all of the models for evaluating and comparing their performances. We did this because we wanted our false positive rate to be as low as possible. False positives in the dataset mean that reviews predicted as useful could actually be meaningless when

evaluating by the users. Hence with a low false positive rate, the reviews recommended by Yelp would be more likely considered as useful by the users, which increases the practicality of the app and better meets our goal of modeling. In general, Random Forest outperformed all alternative models that were tested. In the following sections we will provide a short description of the performance of each model and demonstrate why we concluded that Random Forest was the optimal choice for us.

3.2.1 Logistic Regression

In general, logistic regression models are simple and reliable for most binary classification problems. With this in mind we first tried to apply a logistic regression.

The default baseline logistic regression model with 28 features gave an AUC value of 0.764. We tuned the C value to see if we could get better model performance. We used K-Folds cross validation, and compared the mean AUCs on each C value from 10^-8 to 10^2. The model chosen based on the one standard error rule gave an AUC value of 0.766, with a C value of 0.01, which didn't improve the model's performance by much.

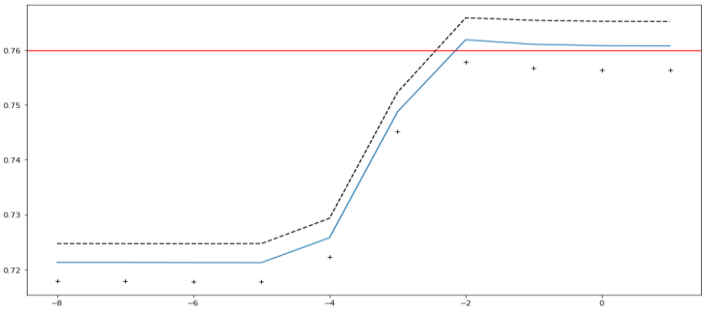


Figure 3.2.1.1: AUC mean and standard deviation of logistic regression with different C values

We also used a recursive feature elimination module imported from sklearn.feature_selection to find the logistic regression model with the ten most important features as explanatory variables. The 10 features selected by RFE were 'stars_review', 'is_open', 'stars_business', 'review_count_user', 'friends_number', 'text_length', 'text_positive', 'Korean', and 'Thai'. The logistic regression model fitted by these 10 features gave an AUC value of 0.759, which was very close to the baseline model. As a result, we believe that this model performed almost the same as the baseline model, with far fewer features, making the model much simpler. We used K-Fold cross validation, and got the mean AUCs on each C value. For C values ranging from 10^-8 to 10^2, we got the plot of the logarithm base 10 of each C value and mean AUCs as shown in Figure 3.2.1.1. The model chosen based on the one standard error rule gave an AUC value of 0.762.

3.2.2 Classification Tree

The classification tree is also a classic classification model and one that is comparatively easier to interpret. When we first fitted the classification tree on

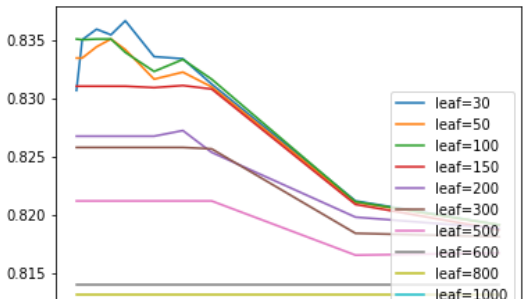


Figure 3.2.2.1: AUC mean with different minimum sample split and minimum leaf value

all 28 variables, the mean AUC using K-Folds cross validation was around 0.85. For simplifying the model, we reduced the number of predictive variables to 5, as a result, the AUC fell to 0.81. After tuning the parameter on minimum split size and minimum leaf, we had a model with AUC 0.835, as seen from Figure 3.2.2.1. The corresponding minimum split size and minimum leaf value were 200 and 30. The 5 variables chosen were 'is_open', 'review_count_business', 'stars_business', 'review_count_user' and 'compliment_photos'.

3.2.3 SVM

Support vector machine is also a classification model that is fairly effective in high dimensional spaces. Due to the large number of predictive variables we have, we included SVM in the analysis. The baseline model had the default parameters, and its AUC was about 0.703, which was moderate for a baseline model. According to the correlation and importance between our features and target variables, we decided to scale down the number of features. The model that performed the best was the one with 15 predicting features: 'stars_review', 'review_count_business', 'stars_business', 'review_count_user', 'average_stars_user', 'compliment_photos', 'since_first_review', 'friends_number', 'text_length', 'text_positive', 'text_negative', 'count_food_detail', 'count_food_quality', 'queue_in_line' and 'service'. In order to further improve the performance of the model, we tuned the penalty parameter C in the range of 10^-8 to 10^2. According to the tuning result and to achieve a balance between property loss and training error, the value of 10^(-4) was selected as giving the best prediction, this generated an AUC value of 0.753.

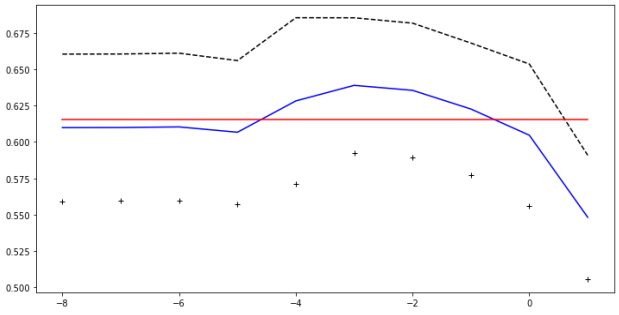


Figure 3.2.3.1: The mean AUCs and its standard deviation of SVM with C from 10^-8 to 10^2

3.2.4 Random Forest

Random Forest is an ensemble method which means that it constructs multiple decision trees at training time. The advantages of an ensemble method are that it can decrease biases, reduce variance and is less likely to overfit. But compared to other classification supervised algorithms, Random Forest is rather slow to run and is difficult to interpret.

The baseline model was the model including all 28 predictors with the default parameters: the number of trees in the forest was 10, the minimum number of samples required to split an internal node was 2. The AUC of this baseline model was about 0.847, which is quite high for a baseline model. To improve the performance of the model, we tuned the parameters and performed feature selections.

The feature_importances_ function in the sklean package shows the importance each predictor contributes in the model. Including the top 5 features in the model, the AUC of this random forest was 0.819, which is extremely close to the baseline one. We think the reduced AUC is acceptable given the increased ease of implementation provided. We also tuned the parameters to make the model perform better. The default minimum number of samples required to be a leaf node is 1. As seen from the Figure 3.2.4.2, the model performed best when min_samples_leaf equaled 25. The default n_estimator is 10. After plotting a few n_estimator values, we found that the model performed better with a higher number of trees in the model. However, the running time increased tremendously with higher n_estimator values with comparatively small improvements to AUC. As such, we decided to choose the n_estimator with the largest marginal AUC of 30. As for the minimum number of samples required to split an internal node, we determined that when min_samples_split equaled 10 the model performed best. After the model tuning procedure, with the improved model, the AUC increased to 0.843.

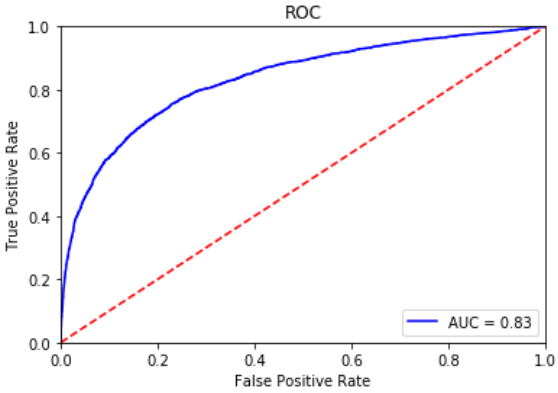


Figure 3.2.4.1: The ROC curve of final Random forest model

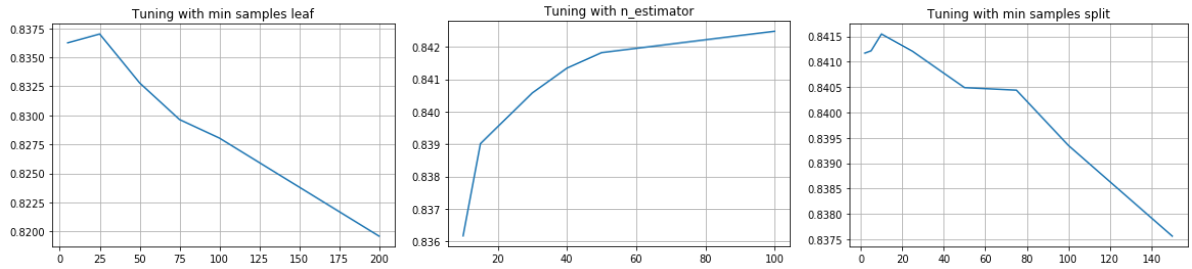


Figure 3.2.4.2: AUC result for each parameter in Ransom Forest

3.2.5 Evaluation

Classification Model Comparison	Number of features in the Best model	AUC for the best model
Logistic Regression	10	0.762
Classification Tree	5	0.835
SVM	15	0.753
Random Forest	5	0.843

Figure 3.2.5.1: Comparison of each model performances

Among all of the evaluated models, random forest model had the highest AUC rate with the fewest features used, so in our case it is the best model to predict whether Yelp reviews will be considered useful by majority of customers. It also has the benefit of being a comparatively robust and accurate model in terms of

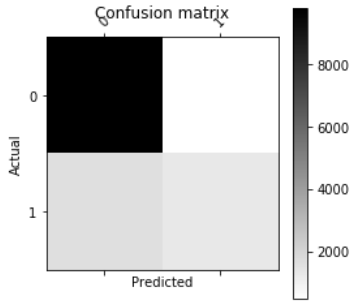
predictions. It only uses 5 features, the true positive rate is about 0.44 and the false positive rate is about 0.04. The percentage of correctly predicted useful reviews is high among all reviews we suggested as useful. Additionally, even though we cannot see the actual extent of positive/negative influence of features on the response in our random forest model, we can use intuitive thinking to infer the causal relation from the feature importance table. For example, 'friends number' is the most important feature here as shown from feature importance, it's reasonable to think that the more friends a user has, the more likely a review would be influential and popular. Thus, in addition to predicting review usefulness, Yelp can also use this random forest model to study factors that make a review popular. With this model, Yelp can attract those customers who value useful information in reviews while also improving their decision making in the future.

```
friends_number 0.403544906552
review_count_user 0.2149698896
review_count_business 0.0776282869392
since_first_review 0.145958029635
text_length 0.157898887274
```

Figure 3.2.5.2: Feature importance table for Random Forest

true positive rate	false positive rate	true negative rate	false negative rate
0.44227	0.044161	0.955839	0.55773

Figure 3.2.5.3: Confusion Matrix for Random Forest



4. Deployment

When it comes to deployment, our model can be integrated into the existing recommendation system of Yelp. Although determining whether a review contains useful information is certainly very important to the usefulness of review, the website should also consider other factors. For example, the quality of photos, whether the information is up to date to current business conditions, and so on so forth. With this in mind, we suggest mixing the results from our model with other relevant review related factors in order to better generate a good ordering of reviews on their page. For evaluating the performances of deployments, we feel that AB testing should be used. We could randomly select some users, provide them with pages of new reviews, and compare their review click rate and useful click rate with other users browsing original pages.

During deployment, this model should not be revealed to actual users. It could become problematic if users know what determines the ranking of their reviews on the website, as they could abuse this information to try and obtain higher-ranking reviews. As an example, one feature important to our model is 'text length'; this feature follows the logic that a longer review is more likely to contain useful information and is used as such in our model. An individual who is aware of this could attempt to shore up their reviews with more words, useful information or not, in order to trick our model. The individual writing the review who has the end goal of simply increasing the word count of their review to trick the system is more likely to include non-useful information which would in turn reduce the likelihood of the model giving good ranking

recommendations based on the exposed feature. This applies to any feature, when exposed they lose much of their usefulness as a feature if individuals try to exploit them.

5. Discussion and Future work

With regards to constraints, our model is not able to detect fake reviews. For example, a fake review is likely to pass by our model if it is well constructed with sufficient text and good usage of photos or other content. However, our model can compensate by analyzing the user who posts the review. The model can use the number of reviews a user has released, the number of friends they have and other factors in order to help evaluate the validity of a review. These factors are difficult to falsify for the sake of posting fake reviews and serve to provide a secondary means of evaluation for reviews.

As mentioned previously, the number of friends a user has is a feature that is important in this model. This can lead to complications with the way the model evaluates reviews from new users. New users may post useful reviews but have a low friend count on their profile. As such, the age of an account is important to consider and thus to avoid the issue of new user accounts, more weight can be place on reviews posted by new users.

Another limitation of this model is that it can only determine whether a review is likely to be useful. How these reviews should be sorted and presented to the users is not decided. However, we believe that this issue could at least partially be resolved by mixing in the useful prediction with the other relevant factors as suggested above.

Another thing to consider, the Yelp dataset we used in this model was already sampled by the Yelp website. We don't know how the Yelp website sampled the data. As such, we cannot know for sure whether or not the data was randomly sampled. If not, this could mean that there is a selection bias in the data set which would affect our model's accuracy during deployment.

For the future, one possible improvement for our model is to add support for languages other than English. Despite the fact that, as discussed earlier, the majority of Yelp users posted English reviews within the United states, reviews in other languages do exist and should be taken into consideration. The text mining based feature generation in our model is only valid for English reviews. It cannot assess reviews written in other languages. We cannot measure the sentiment of Spanish review as an example. In order to include reviews posted in other languages into our sample, additional corresponding language text mining features are required to update our model.

6. Conclusion

In order to recommend useful reviews, we have tried different regression and classification models to predict the usefulness of each review. Among which, Random Forest with 25 min_samples_leaf, 30 n_estimator and 10 min_samples_split offers an AUC of 0.843 and has recorded as the best. This is reasonable

due to the fact Random Forest is an ensemble learning method. With our model, Yelp can dig the gold from bunch of reviews and offer user better experience. Directions for further investments include exploring different choices of classifiers, embedding fake review detection, weighting new users and updating with different languages features.

References:

1. Yelp dataset challenge, 2017. http://www.yelp.com/dataset_challenge.
2. Hu, Minqing, and Bing Liu. "*Mining and Summarizing Customer Reviews.*" Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '04, 2004, doi:10.1145/1014052.1014073.

Appendix A: Variable Descriptions and Notations

VariableName	UniqueVal	Min	Max	Mean	StandardD	25%	50%	75%	Note
stars_review	5	1	5	3.732	1.311	3	4	5	Star rating of the review
is_open	2	0	1	0.86	0.347	1	1	1	0 or 1 for that restaurant closed or open
review_count_business	932	3	6979	495.338	878.926	83	215	507	Number of reviews of the restaurant in total
stars_business	9	1	5	3.776	0.599	3.5	4	4	Star rating of the restaurant in total, rounded to half-stars
attributes.RestaurantsPriceR	5	1	4	1.837	0.668	1	2	2	The price range of the restaurant, from 1, the least expansive to 4, the most expansive
review_count_user	1345	0	7818	349.413	669.243	16	89	399	The number of reviews the user have written
average_stars_user	370	1	5	3.778	0.617	3.54	3.83	4.1	Average rating of all reviews from user, round to 2 decimal place
friends_number	1743	0	8350	599.2392	1168.339	5	86	563	Number of friends of the user
since_first_review	4302	0	4519	1357.024	1128.772	337	1113	2216	The number of days the restaurant opened before the review, calculated by counting the numebr of days between the date of review and the first ever review of that restaurant
text_length	973	1	1041	168.444	153.922	56	121	230	The number of words in review content
text_positive	66	0	71	6.377	5.985	2	5	9	Count of word appearance in review content from positive word list
text_negative	34	0	36	2.173	2.803	0	1	3	Count of word appearance in review content from negative word list
Chinese	2	0	1	0.047	0.212	0	0	0	Whether the business is a Chinese restaurant
Mexican	2	0	1	0.088	0.284	0	0	0	Whether the business is a Mexico restaurant
American	2	0	1	0.291	0.454	0	0	1	Whether the business is a American restaurant
Korean	2	0	1	0.016	0.124	0	0	0	Whether the business is a Korean restaurant
Italian	2	0	1	0.083	0.276	0	0	0	Whether the business is a Italian restaurant
Japanese	2	0	1	0.059	0.236	0	0	0	Whether the business is a Japanese restaurant
Thai	2	0	1	0.025	0.155	0	0	0	Whether the business is a Thai restaurant
Vietnamese	2	0	1	0.016	0.126	0	0	0	Whether the business is a Vietnamese restaurant
count_food_detail	50	0	59	2.746	3.902	0	1	4	Count of word appearance in review content from food detail word list, measureing the extent of information provided in review about food detail
count_food_quality	19	0	22	0.12	0.533	0	0	0	Count of word appearance in review content from food quality word list, measureing the extent of information provided in review about food quality
Useful	2	0	1	0.5	0.5	0	0	1	Target Variable, 1 means the number of useful click the review receive divided by every 60 days since the review is published is greater than 1
operatinghours	289	0	23	8.854	3.682	6.428571	9.428571	11	How many hours the restaurant opens every week
price_cost	7	0	11	0.044	0.226	0	0	0	Count of word appearance in review content from price_cost word list, measureing the extent of information provided in review about whether the restaurant is worth the
traffic	4	0	3	0.004	0.069	0	0	0	Count of word appearance in review content from traffic word list, measureing the extent of information provided in review about traffic or whether the reataurant is
queue_in_line	9	0	8	0.241	0.577	0	0	0	Count of word appearance in review content from queue_in_line word list, measureing the extent of information provided in review about waiting time
service	18	0	17	0.38	0.777	0	0	1	Count of word appearance in review content from service word list, measureing the extent of information provided in review about quality of service
ambience	10	0	9	0.119	0.395	0	0	0	Count of word appearance in review content from ambience word list, measureing the extent of information provided in review about the atmosphere of restaurant

Appendix B: Word Lists

Food quality word list:

['Acidic', 'Acrid', 'Aged', 'Bitter', 'Bittersweet', 'Bland', 'Burnt', 'Buttery', 'Chalky', 'Cheesy', 'Chewy', 'Chocolaty', 'Citrusy', 'Cool', 'Creamy', 'Crispy', 'Crumbly', 'Crunchy', 'Crusty', 'Doughy', 'Dry', 'Earthy', 'Eggy', 'Fatty', 'Fermented', 'Fiery', 'Fishy', 'Fizzy', 'Flakey', 'Flat', 'Flavorful', 'Fresh', 'Fried', 'Fruity', 'Full-bodied', 'Gamey', 'Garlicky', 'Gelatinous', 'Gingery', 'Glazed', 'Grainy', 'Greasy', 'Gooey', 'Gritty', 'Harsh', 'Hearty', 'Heavy', 'Herbal', 'Hot', 'Icy', 'Infused', 'Juicy', 'Lean', 'Light', 'Lemony', 'Malty', 'Mashed', 'Meaty', 'Mellow', 'Mild', 'Minty', 'Moist', 'Mushy', 'Nutty', 'Oily', 'Oniony', 'Overripe', 'Pasty', 'Peppery', 'Pickled', 'Plain', 'Powdery', 'Raw', 'Refreshing', 'Rich', 'Ripe', 'Roasted', 'Robust', 'Rubbery', 'Runny', 'Salty', 'Sautéed', 'Savory', 'Seared', 'Seasoned', 'Sharp', 'Silky', 'Slimy', 'Smokey', 'Smothered', 'Smooth', 'Soggy', 'Soupy', 'Sour', 'Spicy', 'Spongy', 'Stale', 'Sticky', 'Stale', 'Stringy', 'Strong', 'Sugary', 'Sweet', 'Sweet-and-sour', 'Syrupy', 'Tangy', 'Tart', 'Tasteless', 'Tender', 'Toasted', 'Tough', 'Unflavored', 'Unseasoned', 'Velvety', 'Vinegary', 'Watery', 'Whipped', 'Woody', 'Yeasty', 'Zesty', 'Zingy']

Food detail word list from wordnet imported from nltk.corpus:

['mackerel', 'molasses_cookie', 'smoked_haddock', 'coconut_meat', 'berry', 'tournedos', 'green_goods', 'Empire', ‘poitrine_d'agneau’, 'leftovers', 'coho', 'ceriman', 'butter_cookie', 'winter_crookneck_squash', 'cos', 'tea_bread', 'murphy', 'cress', 'ladyfinger', 'rump', 'mouton', 'bacon', 'cruciferous_vegetable', 'yogurt', 'watermelon', 'hot_cereal', 'delicatessen_food', 'meat_loaf', 'sugar_beet', 'horseflesh', 'sugarloaf', 'coffee_roll', 'hotdog_bun', 'elver', 'edible_fruit', 'onion', 'common_sorrel', 'tangelo', 'Twinkie', 'sea_biscuit', 'goa_bean', 'kitambilla', 'sweet_potato', 'dandelion_green', 'organs', 'honey_cake', 'string_cheese', 'dowdy', 'chinook', 'date', ‘devil's_food’, 'volaille', 'cut', 'apple_pie', 'tuna_fish', 'cake_mix', 'bread-stick', 'spud', 'crabapple', 'cobbler', 'medlar', 'pandowdy', 'sinker', 'prosciutto', 'cappelletti', 'leg', 'white_turnip', 'bouchee', 'schnecken', 'Boston_lettuce', 'ravioli', 'long-neck_clam', 'eater', 'lichi', 'carambola', 'sirloin_tip', 'sourdough_bread', 'spiceberry', 'grissino', 'green_soybean', 'spaghetti_squash', 'hotcake', 'Rome_Beauty', 'fritter', 'naan', 'pastry', 'lansat', 'danish', 'hen', 'cornbread', 'refrigerator_cookie', 'sour_gourd', 'sticky_bun', 'bilberry', 'cohoe', 'plantain', 'quahaug', 'bap', 'smoked_mackerel', 'Red_Delicious', 'linguine', 'giblets', 'cooky', 'coho_salmon', 'process_cheese', 'sole', 'sapote', 'pickled_herring', 'tamarindo', 'crepe_Suzette', 'caviar', 'timbale', 'side_of_bacon', …]

Price cost word list:

['economical', 'of quality', 'affordable', 'reasonable', 'cost', 'tips']

Traffic word list:

['Busy', 'accessible', 'garage', 'valet parking', 'approachable', 'convenient', 'parking']

Queue in line word list:
['crowded', 'congested', 'overcrowded', 'full', 'packed', 'overpopulated', 'stuffed', 'jam-packed', 'crawling with', 'wait']

Service word list:

['service', 'waiter', 'waiters', 'waiter's', 'owner', 'manager', 'managers', 'host', 'hosts', 'waitstaff']

Ambience word list:

['atmosphere', 'dress code', 'ambience', 'elegant', 'chill', 'classy', 'traditional', 'divey', 'casual', 'upscale', 'romantic', 'touristy', 'hipster', 'trendy', 'intimate']

Team Member Contribution:

Yiyan Chen:
Data Cleaning, Text Mining, Random Forest Modeling

Hanlu Zhang:
Data Sampling, Text Mining, Logistic Regression Modeling

Ben Zhang:
Data Sampling, Text Mining, SVM Modeling

Xinyue Zhang:
Data Sampling, Feature Conversion, Linear Regression Modeling, Decision Tree Modeling