Stat 154 Final Project

https://github.com/yitongpan/STAT154_Yelp_DataSet

Chin Chang (24680837)
Ellie Pan (25955884)
Haoran Li (26869713)
Jingyi Liu (25344563)

Introduction

Yelp is a widely-used platform for user to generate reviews of local businesses. Over the past few years, Yelp has grown tremendously to the point where an individual will dictate their dining decisions based almost solely on a restaurant's Yelp page. Therefore, success on Yelp (in terms of positive reviews and high ratings) is very important to a business.

Our overall goal is to predict a business's average rating based on reviews and other business attributes. We will be focusing on two datasets: the individual user reviews for a business and the information about the business, including things like location, price range, and restaurant attire.

Preliminary EDA

We examined the business dataset to see whether there are any obvious trends in star ratings relative to attributes like whether the restaurant takes reservations or not, the price range of the restaurant, etc. Our (training) data set contains 2477 businesses.

Note that the way Yelp determines the attributes is partly based on user reports. When a user gives a review of a business, he or she has the option (not required) of answering a survey about whether or not a business has specific attributes; for example, for the attribute "WiFi", a user can select "Free", "Paid", "No", and "Not Sure". If the majority of users report a specific attribute, Yelp will display that attribute. For many businesses, not all of the attributes are entered i.e. they are missing, so this could either mean that the users don't care about that attribute or that they don't find it relevant to the business (for example, there is no need to select whether or not McDonald's takes reservations).

We can see the average stars pertaining to a few attributes below:

Table 1:

Parking Lot	Avg Stars	Count
NR	3.17	339
False	3.36	1118
True	3.49	1020

Table 2:

Attire	Avg Stars	Count
NR	3.42	722
casual	3.37	1749
dressy	3.40	5
formal	3.00	1

Table 3:

Price Range	Avg Stars	Count
NR	3.13	91
$	3.45	1453
$$	3.30	912
$$$	3.41	17
$$$$	2.88	4

Notice that for some of the attributes such as "Accepts CC", we have much less data for the businesses that are false - therefore, the average stars can be misleading.

In order to effectively predict the average business star ratings, we decided to separate our analysis into two parts: the individual reviews and the business attributes. The idea is to first predict the rating of each individual review and then calculate the business star rating as an average of all individual review ratings.

Table 4:

Accepts CC	Avg Stars	Count
NR	3.13	86
False	3.71	52
True	3.39	2339

Table 5:

WiFi	Avg Stars	Count
NR	3.24	647
free	3.48	728
no	3.41	1090
paid	3.29	12

Table 6:

Delivery	Avg Stars	Count
NR	3.45	701
False	3.44	1095
True	3.24	681

Afterwards, we would separately predict the average business stars using its business attributes. At last, we would find an appropriate weight between the two predictions to generate the final prediction result.

Individual Reviews Analysis

We recognize that every person has his or her individual preferences. People's standards for the rating rubric are different (e.g. the criteria for 4 stars is different for everyone) and there is a degree of irrationality associated with the review and ratings given. Therefore, we decided to start by splitting the reviews into individual sentences to generate the sentiment (positive/negative) associated with each individual sentence. We manually built our dictionary, which contains sentences and the sentiments corresponding to them. Then we employed the idea of Naive Bayes to perform sentiment analysis and built a prediction model, which we then applied to our entire training set (sentence based). At last, we used Latent Dirichlet Allocation to discover the hidden topics in the reviews and associated the most significant topic with the sentence. We intended to predict the rating through the aggregated sentence sentiment and topic. However, we obtained a relatively low prediction result (0.93 MSE). Through analyzing the result, we found out that there are three main reasons to our high MSE: First, the topics generated by LDA had many overlapping words (one word could appear in several topics). Secondly, people put different weights on different elements in their ratings (e.g. one person may view wait time as more important than food quality). Lastly, the important word features extracted from Naive Bayes directly depends on the restaurant type (e.g. one of the top features is "espresso", but for restaurants that don't serve espresso, this feature is not applicable).

Since LDA appeared to be an inappropriate method to apply in our project, we decided to drop the LDA part and build the model purely based on Naïve Bayes. This time we did not split the reviews into sentences but ran the model on the review level. The reason is that if we were to train based on sentences, we would have to associate each sentence within one review with the same rating star. This distribution would introduce a lot of errors when we run the model. To be more specific, say we have a review with rating star '3' saying that "The chicken is tender. However, the wait time is too long. But we will come again" If we were to break this review into sentences to build our model, we would distribute each sentence with a rating star '3'. This does not make sense because different sentences actually indicate different sentiments and they should be associated with different rating stars. Thus, we didn't run the model on the sentence level. Through running the model on the review level, we got a smaller MSE, 0.80.

In order to obtain a higher prediction result, we switched to SVM. We built the model using SVM on the review level to predict the rating star. In order to find the optimal tuning parameter alpha, we sampled with replacement 1000 times with a five-fold cross validation. After plotting the relationship between the MSE and the alpha, we realized that it was generally a monotonic increasing function with many small fluctuations inside. Therefore, we reduced the range of our testing alpha and rerun the algorithm. We repeated this process several times and obtained an alpha that returned the "global" minimum MSE. We then predicted on the actual testing data set to obtain a set of predicted values. We took the value of alpha that minimized MSE: alpha = 3.43224*10ˆ(-5) , and this parameter gave a 0.375 MSE.

Below are the MSE plotted against the values of alpha that we tried. The left graph corresponds to extremely small values of alpha and the right graph corresponds to a wider interval of alpha.

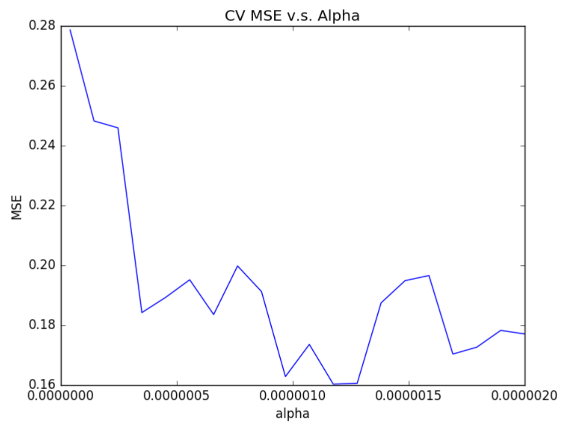

Figure 1: MSE by Parameter

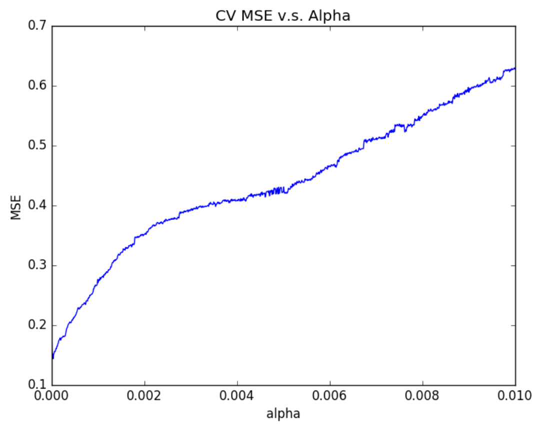

Figure 2: MSE by Parameter

Business Attributes Analysis

Before starting the analysis, we did some data cleaning on the training dataset. First, we dropped the columns with unique values such as names of business or address. There are 2510 businesses in the business dataset while 33 of them have missing values in the attributes column. Since the business attributes column contains the most important features that we want to focus on, it would be hard for us to use businesses without that column. So we discarded those 33 businesses (the number is too small to make an impact). Second, since many of these attributes are missing values (For example, the attribute "Dietary Restrictions - Soy Free" is not reported.), we decided to discard the attributes for which we don't have enough businesses that reported those values. We selected this threshold to be 58% - therefore, if an attribute did not have at least 58% of businesses with reported values, it would be dropped. Lastly, we separated the "categories" column into two columns: one specifies in the type of food (American, Mexican, etc.) and the other specifies in the type of establishment (Food or Restaurant). Moreover, we didn't use the "hours" column as there were too many different time choices for businesses. The best way to use this attribute would be to separate it into whether or not the business is open for different time blocks: for example: breakfast, dinner, etc. Meanwhile, one of the business attributes is whether or not the place is good for breakfast, dinner, etc... We assumed that if the hours of the business were enough of an influence on the stars, then the business attributes would capture its importance. Thus, we dropped the "hours" attribute.

After cleaning the data, we were left with 118 columns and we decided to use one hot encoding to generate a dummy variable for the response value of each attribute. Although we got much more predictors than before, the design matrix was still very sparse due to the one hot encoding.

Finally, when we ran random forest on this dataset and tested it using cross validation, we got an MSE of approximately 0.48. The value is relatively large, which is expectable because we didn't use any text reviews for this prediction. While we did anticipate a correlation between stars and business attributes, we did not expect it to be this large.

Combining the Individual Reviews Analysis and Business Attributes Analysis

The idea of this part is to find a relatively "clever" way to weigh these two methods to obtain our final prediction result. Even though we were able to obtain the prediction result by using two methods in the previous two sections, it is highly unlikely for us to find the correct weight for these two methods without a training set.

We first attempted to apply our model on our entire training set to get these two prediction value columns for the whole dataset. Then we combined the predictions with actual rating stars and sampled with replacement from the dataframe to test the appropriate weight (beta and 1-beta). After testing the weight discretely and recording the MSE, we picked the weight that gave the least MSE. We used that weight to weigh the predictions on the actual testing set and obtained our finalized prediction results.

Before submitting that prediction, we realized that the idea above was not good enough because of the overfitting issue. Therefore, instead of the method above with fixed tuning parameter, alpha, we chose to loop over step 1 and 2 (SVM and CART random forest) to find the ideal values for weight and alpha. Specifically speaking, we first subset a testing set from our training data and sampled it with replacement, then we tuned the alpha across different subsets and changed the weight of these two models. After repeating the procedure, we anticipated an optimal alpha and beta. In theory, the idea described previously is reasonable. However, in practice, eight-hour running still didn't give us the results.

In order to fix the problem above, we decided to employ boosting regression on those two predictors. The idea is to fit a functional basis representation of the data, get residuals, and fit the residuals to the regression model. We repeatedly fitted the residuals and updated the function until we obtained an incremental MSE or hit the 500 iteration times, whichever is faster. At last, we summed all the regressions. Through doing that we obtained our desired weight of these two methods. The MSE we obtained from using this method on the actual testing data is 0.336.

Conclusion

In conclusion, our main goal for this analysis was to create a model that will allow us to accurately predict a business's average star rating based on a set of user reviews and business attributes. We first performed EDA on the business dataset and noticed a few trends in average star ratings relative to certain business attributes. This implied that we could potentially find a few attributes that, if a business were to have them, can result in a higher star rating. After running EDA, we decided to do two analyses: Individual Reviews Analysis and the Business Attributes Analysis. When performing Individual Reviews Analysis, we mainly tried Naïve Bayes and Support Vector Machines (SVM). We built a dictionary that contains sentences and the corresponding sentiments to perform Naïve Bayes method. While Naïve Bayes returned a MSE of 0.80, SVM gave us a better MSE, which is 0.375. Afterwards, we analyzed the business dataset using Random Forest and we got a 0.48 MSE. Note that at this point, we could have run CART on the dataset. This will result in a partitioning of the feature space; in this case, this would give us a set of regions where the regions represent businesses that have specific attributes, and we also have a corresponding estimate for the average stars. Therefore, this will essentially allow us to say "businesses with these attributes can expect this average star rating". Now, since the MSE we obtained from doing these two analyses separately were relatively high, we decided to combine these two analyses and put weights on them to acquire an optimal prediction result. We used boosting regression to obtain the ideal weights on these two predictors. This time, we achieved the smallest MSE among all our attempts and this value is 0.336. Therefore, we have found a model that allows us to predict a business's average star ratings when given user reviews and business attributes.