Star-Rating Evaluation System of User Reviews

Yiting Xiao
Jian Tian
Yongshuai Wang

Introduction & Motivation

An evaluation system composed of star rating and customer review plays a very important role in a customer's decision on which new restaurant to try. Star rating and customer review are of equal importance in evaluation of a restaurant. It is difficult to tell which one is better. Star rating is more straightforward and generalized, while review is more expressive and informative. In reality, however, star rating dominates people's choice because of its highly compressed information which can be obtained in a short time. The overall rating of a restaurant is simply calculated by taking the average star rating from all users indiscriminately. However, the overall rating is likely to be biased, especially if there are fewer reliable star raters. Under such situation, review is a good way to distinguish reliable reviewers from the unreliable and predict the star rating that a customer is likely to give.

Problem definition

To transform our idea into a data analytics problem, we will go through a couple of necessary steps. First, we will decompose the dataset downloaded from Yelp into several parts, and extract the most important components that will be used in our analysis such as user id, review text and the usefulness of the reviews, which directly correspond to the reliability of users. We define users with higher votes of their reviews as more reliable users. Such distinction of users can allow us to make better prediction in further analysis. Second, we will train a prediction model based on a training set retrieved from the Yelp dataset to provide accurate star rating for any piece of reviews. Third, we will prepare data that we get from previous two steps for data visualization. We will visualize our findings such as the relationship between star rating and key buzzword in reviews in a straightforward and elegant way.

Algorithms and Methodologies

One of the key tasks of our project is to assign accurate star ratings of each user review. It is possible sometimes the star rating given by a user does not fully reflect the review he or she provided. Therefore, effective classification algorithms are deployed to accomplish this task.

Because many words in a review have little to no contribution to the classification of star rating, we need first extract useful features from the text. In our project, we first use tokenization to turn the review text as a whole string into tokens, we removed all the stopping words defined by NLTK's English stop words set.

Next we used stemming to turn all words into stemmers so that word that has essentially same meaning with word of different tense or forms will be reduced to same stemmer.

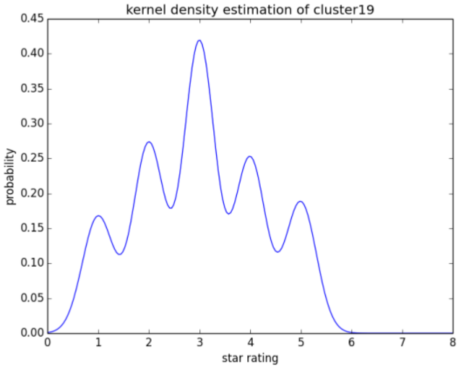
After the preprocessing steps, we need to quantify the contribution of each stemmer to the star rating of each review. We used vector space model to represent our text and quantify each word (stemmer) using Term Frequency Inversed Document Frequency (TF-IDF) to represents the relative importance of each unique stemmer (feature) across all the reviews. In the TF-IDF matrix, each row represents a review and each column represents a unique word. The value of the TF-IDF matrix represents the relative importance of each word in a particular review.

After this step, we are able to conduct classifications or clustering on the TF-IDF matrix. For the purpose of exploratory data analysis, we conducted K-Means algorithm to identify if there is a particular pattern inside each cluster. So we first created a cosine similarity matrix, which measures the similarity between each two reviews in the review set using the following formula:

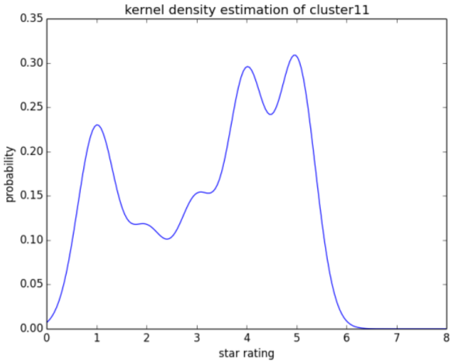
$$cosine(\theta) = \frac{AB}{||A|| * ||B||}$$

Then, we ran K-means clustering algorithm on this cosine similarity matrix, a rule of thumb is to take k = $\sqrt{k/2}$ number of clusters, where k is the number of data points. In our case, we have 2655 reviews, so we take $\sqrt{2655/2} = 36$ clusters. Each cluster of reviews are closer because of their underlying latent semantics are similar. How about the distribution of the star rating inside each cluster of reviews? To satisfy our curiosity, we used kernel density estimation with sklearn.neighbors.kde package. Then, we surprisingly find out that among those clusters of reviews, the probability distributions of star ratings among those clusters mainly conforms to four type of interesting distributions, as shown below:

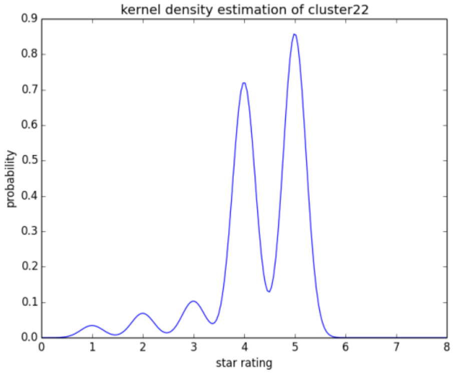
Approximately normally distributed



Distributed with extremes



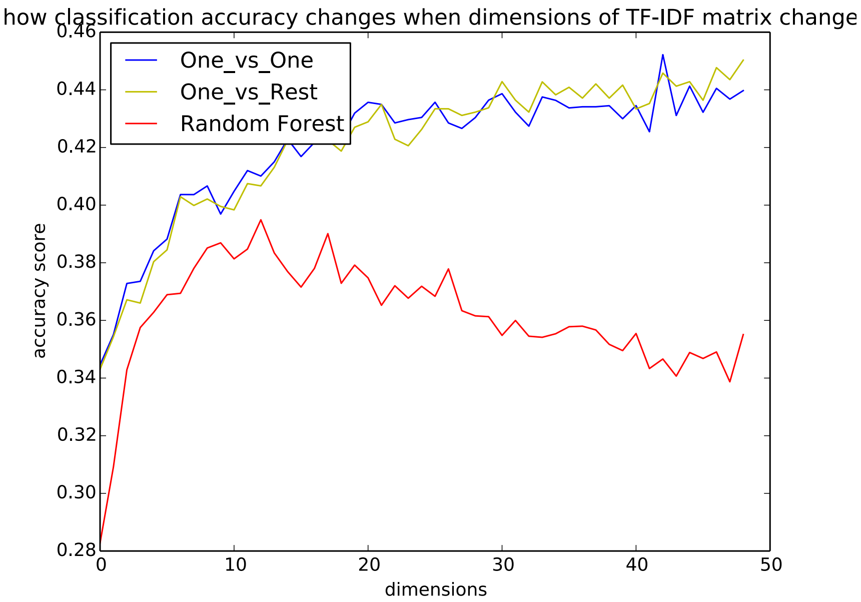
Left skewed or right skewed



As we have mentioned, our ultimate goal is to effectively classify a review with a star rating, so we wonder if running the classification algorithm within each cluster of reviews, or in the same sense, running classification algorithm on data with different star rating distribution could results in different testing accuracy level. However, we need to first select the right classification algorithm to conduct our task.

Because there are 5 star rating levels for our label, we need to conduct a multi-label classification task. We choose to experiment with multiclass linear support vector classifier, both with one vs one strategy and with one vs rest strategy, as well as a random forest algorithm we implemented from scratch without using machine learning package. Our Random Forest algorithm is based on Cutler's PERT algorithm[1].

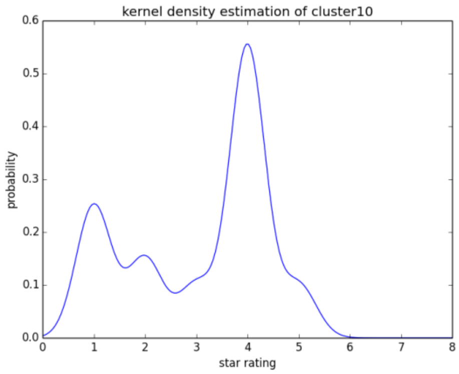
We realized the how the curse of dimensionality could affect our classification accuracy, so as we mentioned in our progress report, we conducted truncated singular vector decomposition to help us decrease the dimensions of our data. Too few dimensions could cause a lot of lost of information, too many dimensions could also make it difficult to classify accurately. Thereafter, by running two versions of multiclass support vector classifiers and the random forest we implemented on a dimension from 1 to 50, we recorded the 10 fold cross validation error of each algorithm on TF-IDF matrix with different dimensions and got the following empirical results:

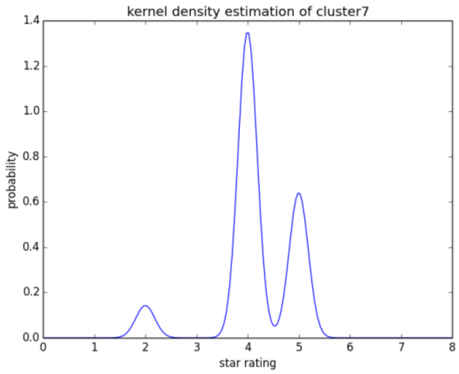


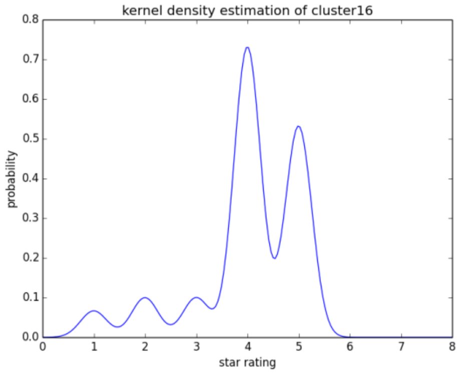
From the above graph, we can see the out of sample testing accuracy of all three algorithms is comparable, the accuracy of random forest is quite low when dimension is small, but jump quite sharply when dimension of data increases. It is also interesting to notice that as the dimension of the TF-IDF matrix increases, the out of sample testing accuracy increases for both versions of support vector machines. It is clear that multiclass support vector classifier outperform the random forest algorithm we implemented.

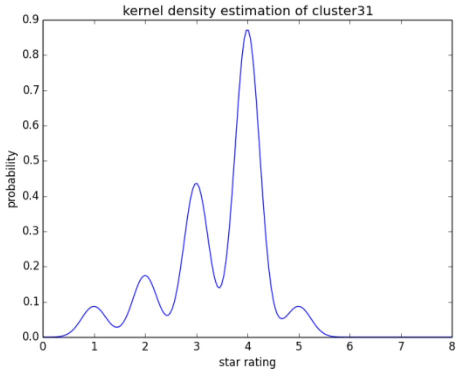
Therefore, we would like to use support vector classifier to take the task of classification. For each cluster of reviews, we separately created a TF-IDF matrix, from the above graph, we concluded that the higher the dimension is, the classification is more likely to be accurate. But as the dimension continue to increase, the rate of accuracy increasing decreases, therefore, we think a dimension of 30 is appropriate for the TF-IDF matrix to be constructed for each cluster of reviews. For each cluster of reviews, we computed the 10 fold cross validation accuracy score of SVM classification algorithm, and sorted the clusters based on the accuracy scores. Our results show that cluster 10 (0.6866), 7(0.6666), 16(0.6375), 31(0.6033) are the 4 clusters with highest accuracy score, whereas cluster 17(0.2181), 34(0.2496), 12(0.2811), 3(0.2877) has the 4 lowest classification accuracy score. What are the distributions of star rating for those clusters looks like?

The star rating distribution of the four clusters with highest accuracy score is shown below:

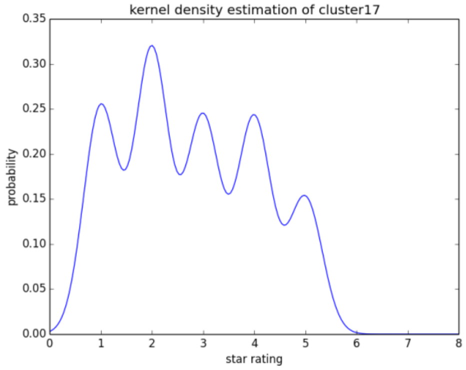


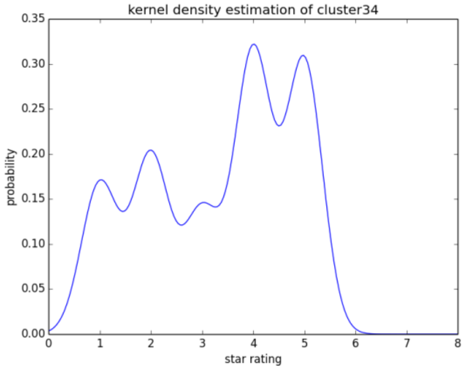


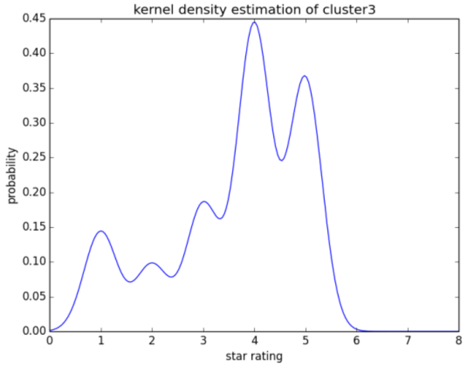


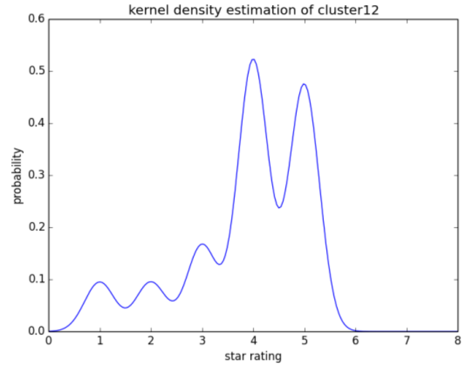


The star rating distributions with lowest accuracy score is shown below:









From above 8 distributions, we can discovered that the cluster with high classification accuracy have at most 1-2 star rating with very significantly high probability and the rest star rating has small probability; On the other hand, the clusters of reviews that have low classification accuracy score are often multimodal, no star rating has probability much higher than other star ratings, which may help the algorithm more easily identify patterns of labels among training data.

This is a quite interesting finding. We now understand how the distribution of training labels could positively or negatively influence the classification accuracy.

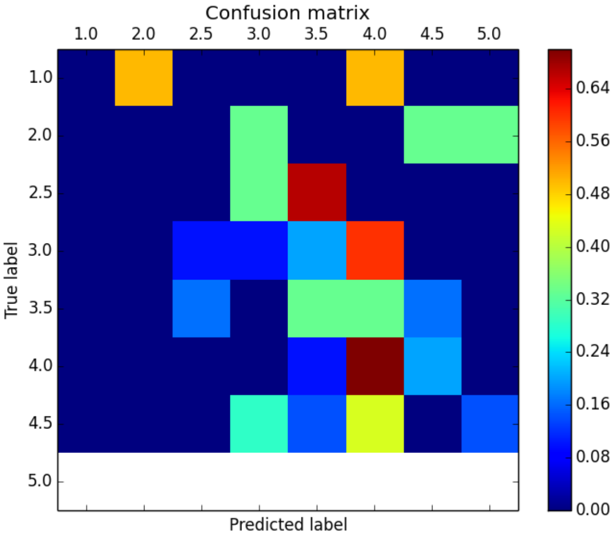
We also utilized GENSIM’s latent semantic indexing model to extract top 16 meaningful topic words from those two groups of reviews, the first group with four clusters of reviews with highest accuracy score has the following meaningful topic words: [('good', 17), ('chicken', 12), ('nice', 8), ('top', 8), ('mexican', 8), ('fast', 8), ('service', 8), ('great', 7), ('pretty', 6), ('wait', 6), ('burgers', 6), ('madison', 5), ('price', 5), ('bar', 5), ('drinks', 4), ('breakfast', 2)], where the first element in the tuple is the topic word, and the second element in the tuple is the frequency of the topic word, the sequence is in reverse order.

Meanwhile, the second group with four clusters of reviews with lowest accuracy score has the following meaningful topic words: [('chinese', 23), ('coupons', 19), ('perkins', 17), ('sick', 15), ('hot', 14), ('pudding', 11), ('counter', 9), ('parking', 9), ('air', 8), ('treat', 7), ('soup', 7), ('steak', 7), ('sandwich', 6), ('burger', 6), ('cold', 6), ('okay', 5)]

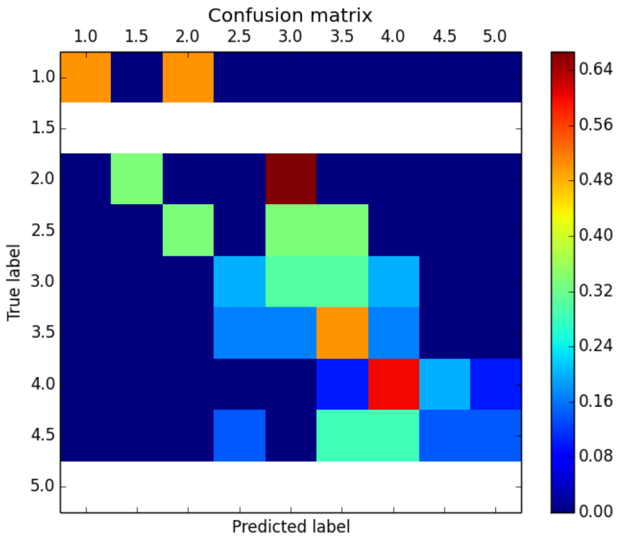
The distribution of the star rating for first group concentrated more on star 4 and star 5, meaning that the first group of reviews may be more positive. The probability of each star rating for the second is more evenly distributed, so the star rating may not be as positive as first group.

Result Evaluation

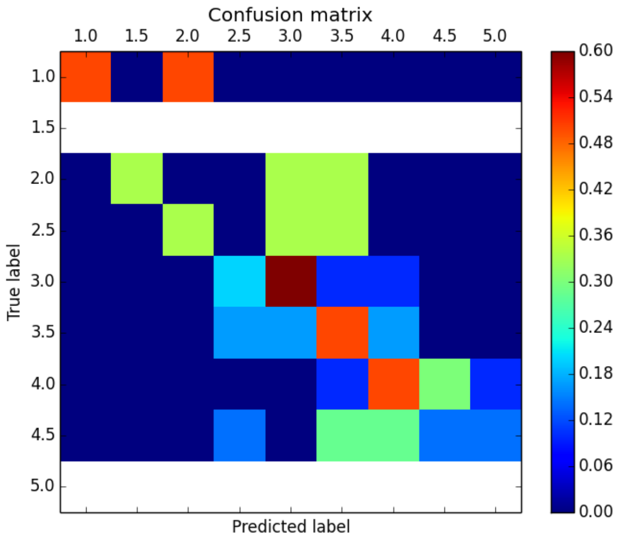
First, we evaluate the three classifiers on the task of business star rating. We report the classification confusion matrix as below. White blocks represent the missing values. We can see that the diagonal part of the matrix has the most significant weight. This shows all the three algorithms can fit the training data well. Visually, we can judge that SVM performs better than Random Forest. The Random Forest model shows the bias of higher rating prediction (giving higher rating).



Confusion Matrix of business with random forest

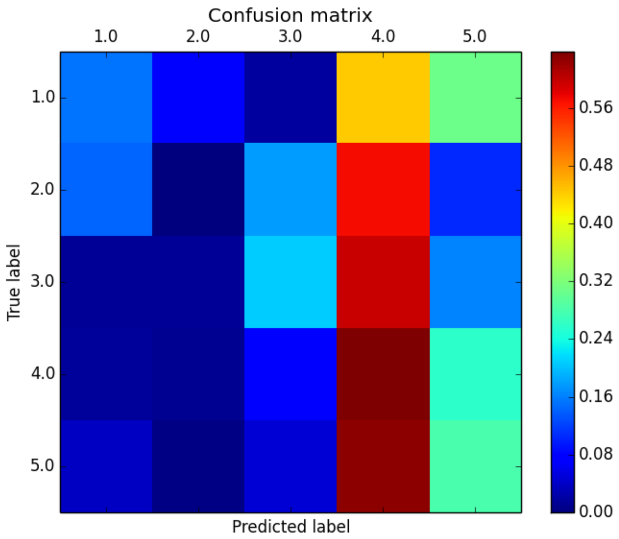


Confusion Matrix of business with one-vs-all SVM

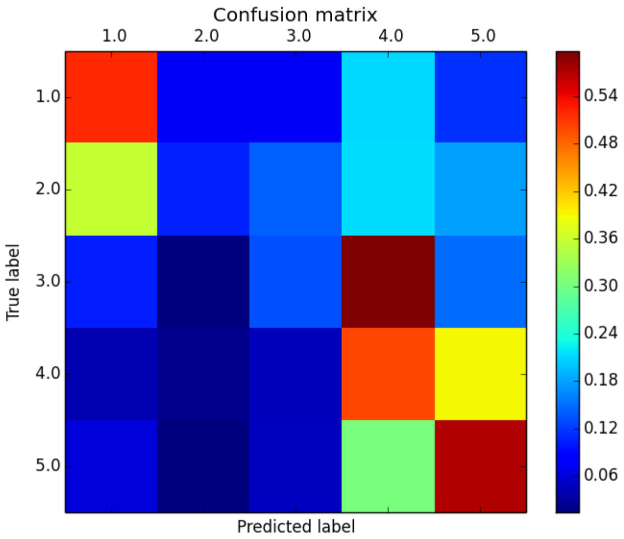


Confusion Matrix of business with one-vs-one SVM

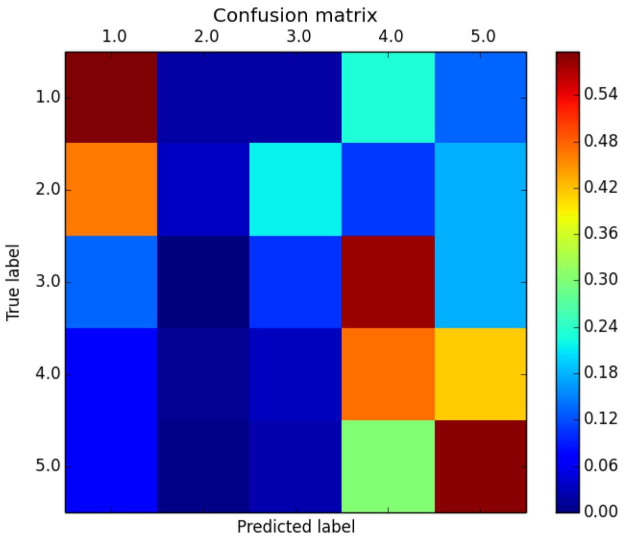
Below we evaluate the prediction results on the single review created by user. The confusion matrices are shown as follows:



Confusion Matrix of reviews with random forest



Confusion Matrix of reviews with one-vs-all SVM



Confusion Matrix of reviews with one-vs-one SVM

In the figure of random forest, we can see that the predicted results are mostly 4.0. It shows that the random forest might be suffering from over fitting. This is due to the lacking of enough training data.

For the results of SVM, we can see it performs well in predicting 1.0 and 5.0 scores. It shows that in these two cases, there might be some features, which are highly related to these two extreme rating (best and worst). However, for the scores from 2.0 to 4.0, it seems that our models are confused. This is also reasonable, since people tend to have different judgment on these median scores, but is more agreeable on extreme ratings.

References:

[1] Adele Cutler, Guohua Zhao, PERT - perfect random tree ensembles, Computing Sciences and Statistics, (2001)