Exploring The Ins and Outs of Yelp: A Machine Learning Approach

Emma Chi (UID: 605-480-973), Helena Hu (UID: 805-520-690),
Matthew Li (UID: 905-537-336) and Sivaji Turimella (UID: 105-535-845)

Department of Statistics

Abstract

The Yelp Dataset is an interesting problem for data scientists and machine learning. With a slew of data about businesses and locations, it's important to clean through the data and figure out what is important and what could be an interesting problem to look at. Employing NLP Pre-Processing models, we cleaned through review data. After that, we employed a number of supervised models and a K-Means Clustering unsupervised model to wrangle through the data. We saw insights in clustering the users into 4 groups. We also see the important factors in predicting a Star rating of a review.

1 Introduction

Yelp is an online platform designed for consumers to discover and connect local businesses, allowing users to leave reviews, check-in to reservations and virtually wait in line at restaurants. Consumers can both write reviews and rate businesses on a 5-star rating scale (1 being the lowest, 5 being the highest); users can also interact with other users' reviews by rating them as Useful, Funny, or Cool.

The Yelp dataset is a collection of information from Yelp containing 6,990,280 reviews from 150,346 businesses spanning 11 geographic locations, released to the public as part of the 2014 Yelp Dataset Challenge. We were tasked to use the Yelp dataset to define our own problem and to implement statistical techniques learned throughout the course of the quarter to analyze a subset of the data provided. In this paper, we explore potential factors of one's business experience that contributed to a user leaving a significantly lower or higher review than they typically would. The Yelp dataset consists of 5 textual JSON files: 'business.json', 'review.json', 'user.json', 'checkin.json', and 'tip.json'. The sixth file, 'photo.json, containing photo data, is not relevant for the assigned task as hand. Each JSON file gives a single object type with information on one specific business, user, or review [Asghar, 2016]. The subset of data provided selects variables only from the 'business.json', 'review.json', and 'user.json' files, providing us information on the businesses and the user who left the review attached to each business, as well as the review information itself. The data was filtered to only include businesses in the Santa Barbara metropolitan area, with reviews left by users who have left at least 150 reviews on Yelp.

With the subset of data provided, we decided to explore deeper into what factors contribute towards a user deviating from the standard number of stars they would typically leave for a business.

2 Preparing the Data

2.1 Pre-Processing

In order to make use of the review text, we needed to contextualize it and process it. We introduced a number of NLP methods. In order to explore deeper into what factors might make a user leave a higher or lower review than their typical standard, it is important to note that stronger language in a review might be used. We utilized standard Python libraries to remove punctuation, determinants, and otherwise insignificant language from the reviews. We then ran term frequency–inverse document frequency analysis (TF-IDF). TF-IDF is a text mining method used to demonstrate the impact of select words in a given collection of text. TF-IDF assisted our search for any strong language to take note of in terms of what users are identifying as key traits in their customer experiences. We combined TF-IDF methods with Sentiment Analysis. We used a standard list of positive words and negative words, then counted how many positive words were in a review. We then counted how many negative words were in the same review. This created two new numerical variables, 'Positive'

(the number of positive words in a review) and 'Negative' (the number of negative words in a review). Finally, we mutated the 'Elite' variable to a single number; the total number of years a user has been an elite reviewer.

2.2 Selecting Relevant Reviews

In order to build our model, we needed to first identify our main variables of interest, as well as extract the cases most relevant to our study. We created our own variable based on the data called Avg_User_Star and StDev_User_Star. Avg_User_Star is the mean star rating a user gives (we found the mean star grouped by user id). We followed a similar process for StDev_User_Star.

Using this new variable, we extracted the most significant cases of a user leaving a review outside of their norm. For this, we followed the empirical rule, which states that 68% of the data lies between one standard deviation above and below the mean. To extract our cases to study, we subset the data to only include reviews in which users had given an amount of stars more or less than one standard deviation from their average star rating.

We conducted exploratory analysis and plotted the distributions of all users' average star ratings and the distributions of users' average star ratings that are more than one standard deviation away from the mean (defined as "relevant" users in Figure 2). As expected, we can see that the left tail is much longer in Figure 1 than in Figure 2, and that the frequencies in Figure 2 follow a more normal distribution.

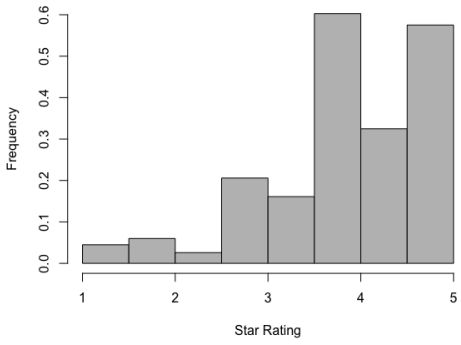


Figure 1: Users' Avg Star Ratings

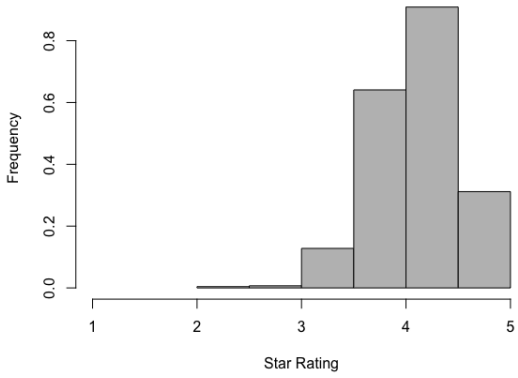


Figure 2: Relevant Users' Avg Star Ratings

3 Unsupervised Learning Analysis

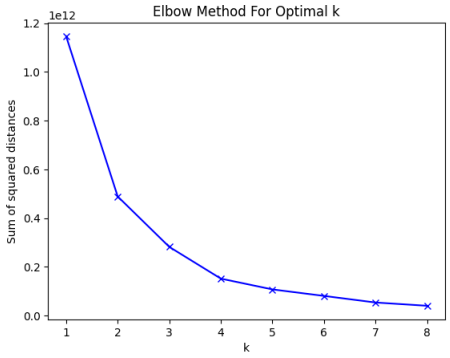
3.1 K-Means Clustering

We performed unsupervised machine learning in the form of K-means clustering, in which datapoints are clustered around a centroid based on the nearest mean [Alade, 2018]. For this, we grouped the data and chose only numerical predictors that were relevant to our task (['User_Useful_count', 'User_Cool_count', 'User_Funny_count', 'Elite', 'User_Fans', 'Avg_User_Star', 'Positive', 'Negative']).

Positive and Negative are predictors from the Reviews. Positive is a TF-IDF vectorization of all the positive words in the review, and Negative a TF-IDF vectorization of all the negative words in the review. The TF-IDF vectorization was noted above in 2.1 Pre-Processing. This allowed us to use textual data for clustering.

We then scaled the data using and used the elbow method, running many different k-means with different values to find the most optimal K, resulting in k = 4.

Figure 3: Using Elbow Method to find k



We excluded common words in order to determine whether the respective star ratings of each cluster are attributed to certain words in the reviews, creating these word clouds in Python [Kharwal, 2021]. Based on the results, it appeared to us that many of the reviews were being clustered together by business type and the users' experiences at those businesses.

Figure 4: Clusters 1,2,3,4 (left to right, top to bottom)

4 Supervised Learning Methods

4.1 K-fold Cross Validation & Measuring Model Success

For our models that we will introduce in this section, we utilized K-fold cross validation to analyze the effectiveness on each of the different supervised learning methods. K-fold cross validation partitions the data into a selected number k splits, then proceeds to test the model defined on one of the folds, using the remaining k-1 folds as training data. This process is repeated k number of times until all folds have served as the testing dataset. K-fold cross validation generally helps accurately measure model performance and bring down bias in estimates, under the assumption that the dataset is randomized [Brownlee, 2018]. For each of the methods we utilized, we used K-fold cross validation to test the success of the model, with a selection of either k = 5 or 10.

For our predictors predicting Star, we ended up with ["Star","Funny", "meanS", "sdS","Negative", "Positive"]. These predictors were decided based on the accuracy with respect to KNN. We ran a backwards selection model to see which predictors would generate the highest accuracy. We started with all the predictors and ended up with the following 6 predictors.

4.2 Method 1: K-Nearest Neighbors (KNN)

KNN is a rather simple model that we chose to implement because of its nonparametric nature; due to the nature of this dataset, we wanted to select a model that could account for incredibly wide varieties of quantitative data [Lateef, 2022]. KNN's flexibility largely contributed to our model selection decision. KNN also clusters data based on similar features, so by running KNN, we aimed to find similarities in our predictors that would help distinguish

what features were the most impactful in the resulting star rating.

To develop our KNN model, we first selected variables based on the permutations for KNN accuracy, choosing the variables with the highest permutations. We then normalized and spliced the data, separating it into training and testing sets. Running this model with the automatically selected k value of 5 in k-fold cross validation, we obtained an overall accuracy of 0.5351, which was lower accuracy than expected.

	Reference				
Prediction	1	2	3	4	5
1	150	97	30	7	36
2	167	280	103	24	126
3	34	103	387	59	214
4	4	9	24	188	17
5	54	182	273	86	893

Table 1: Confusion Matrix for KNN

	1	2	3	4	5
F1 Value	0.41152	0.40846	0.4796	0.62046	0.6438
True Negative Rate	0.94583	0.85396	0.8498	0.98303	0.7368
Accuracy	0.65629	0.63563	0.6618	0.74976	0.7156
Precision	0.46875	0.4	0.4856	0.77686	0.6001
Recall	0.36675	0.41729	0.4737	0.51648	0.6944

Table 2: Results of KNN on Star Prediction

4.3 Method 2: Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis is a linear, non-parametric modeling tool that clusters the data in distinct groups modeled by linear boundary lines. The model seeks to maximize the variances between each cluster of data [Dash, 2021]. Since our data was predicting more than just binary classes, and since more realistic data tends to spread quite widely, we were not expecting LDA to perform as well as our other modeling options because of the strict linear boundaries LDA employs. However, we decided to employ it to see how it would fare against other supervised learning methods and because of its wide use in prediction modeling and classification.

	Reference				
Prediction	1	2	3	4	5
1	206	86	1	0	37
2	147	296	61	6	88
3	10	80	426	131	151
4	0	0	4	162	47
5	46	209	325	65	963

Table 3: Confusion Matrix for LDA

	1	2	3	4	5
F1 Value	0.55751	0.46651	0.56153	0.51538	0.6655
True Negative Rate	0.96048	0.89499	0.8637	0.98398	0.7147
Accuracy	0.73208	0.66806	0.6926	0.71452	0.7318
Precision	0.62424	0.49498	0.5338	0.76056	0.5989
Recall	0.50367	0.44113	0.5214	0.44505	0.7488

Table 4: Results of LDA in Star Prediction

4.4 Method 3: Quadratic Discriminant Analysis (QDA)

QDA is a slightly more flexible supervised learning method, being that it is an umbrella function over Linear Discriminant Analysis and does not assume equal covariance matrices [Ghojogh and Crowley, 2019]. QDA aims to perform a similar job as LDA, but the boundaries used in QDA do not have to be linear, rather it is either an ellipsoid or a hyperbola. We were interested in discovering how QDA would measure up against LDA, and found that the LDA actually performed better than the QDA in producing the true negative rate for our model.

Using a similar methodology that we employed for our LDA model, we ran a 10-k Cross Fold Validation QDA model, which gave an accuracy of 0.6944.

	Reference				
Prediction	1	2	3	4	5
1	198	67	6	10	31
2	141	339	44	9	77
3	26	107	568	14	110
4	4	13	23	304	14
5	40	145	176	27	1054

Table 5: Confusion Matrix for QDA

	1	2	3	4	5
F1 Value	0.54924	0.52927	0.6918	0.84211	0.7727
True Negative Rate	0.96367	0.90577	0.9059	0.98303	0.8284
Accuracy	0.72389	0.70549	0.8005	0.9091	0.824
Precision	0.63462	0.55574	0.6885	0.84916	0.7309
Recall	0.48411	0.50522	0.6952	0.83516	0.8196

Table 6: Results of QDA on Star Prediction

4.5 Method 4: Support Vector Machine (SVM)

Support Vector Machine is a supervised learning machine model that uses classification algorithms for two-group classification problems. The number of features are defined, and each feature has its own dimension– from there, the data points are classified by similarity [Gandhi, 2018]. SVM searches to maximize distance in the plane in order to create the most distinct groups of dimilar data. We employed SVM because it is an easier way to understand distinct and complex relationships between datapoints without needing to perform rigorous transformations on the dataset. The accuracy obtained with this model was 0.6349.

	Reference				
Prediction	1	2	3	4	5
1	202	60	2	0	27
2	135	328	22	3	90
3	30	110	486	32	125
4	0	0	14	242	50
5	42	173	293	87	994

Table 7: Confusion matrix for SVM

	1	2	3	4	5
F1 Value	0.57714	0.52522	0.6075	0.72239	0.6915
True Negative Rate	0.97164	0.91307	0.8912	0.97989	0.7368
Accuracy	0.73276	0.70095	0.743	0.82236	0.7549
Precision	0.69416	0.56747	0.6207	0.79085	0.6256
Recall	0.49389	0.48882	0.5949	0.66484	0.7729

Table 8: Results of SVM on Star Prediction

5 Analysis and Conclusion

5.1 Results and Interpretation

We primarily used the supervised learning methods KNN, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and Support Vector Machine in order to develop predictions for star ratings based on significant features. In terms of variables we did not select, the categories including pertaining to a user such as 'User_review_count' and 'User_Useful_count' were surprisingly ineffective. This is probably due to our preprocessing process, were only users with more than 150 reviews were considered. This large sample size decreases bias and allows for reviewers to be more accurate and not write one-off negative reviews. Thus, a given general user was of less concern in predicting stars.

The KNN model gave an overall accuracy of 0.5351, which was lower accuracy than expected. The F1 values are relatively low in comparison to those generated by some of our other models, though the true negative rates appear to be on par with the other models. LDA performed better than we expected, producing fairly high true negative rates with an average that was relatively on par with our other models. Using 10-k fold Cross Validation, our LDA model gave an overall accuracy of 0.5788 with true negative rates generally higher than the KNN model but lower than the QDA model. The F1 values of the LDA model are relatively similar to those of the KNN model, but are significantly less than those of the QDA and SVM models. Our QDA model gave an overall accuracy of 0.6944, and had the highest F1 values. The true negative rates were slightly higher than those of the other models, but were generally similar. Finally, the SVM model gave an overall accuracy of 0.6349 with generally higher F1 values; the true negative rates were also slightly higher than the first two models but mostly similar to those of QDA.

Because we want high accuracy, high true negative values, and low F1 values, the QDA and SVM models generally performed better than the KNN and LDA models

With our model using the variables 'Bus_Ave_Star', 'Negative', 'Positive','meanS' and 'sdS', it was extremely interesting to recognize that the amount of negative words in a review ('Negative') had not only a higher increase to our model's accuracy than 'Positive', but it also had one of the most significant increases in general. This variable can be linked to a clear distinction of a reviews general trajectory; the more negative words, the more likely the amount of stars a review would get be lower. Conversely, this assumption may hold for the variable 'Positive'. Other variables such as 'Funny' having significant accuracy improvement were extremely interesting findings. In all these cases, indicators of a review's general meaning help developed our model's accuracy.

5.2 Conclusion and Final Thoughts

We concluded that quantifying the reviews by using TF-IDF and creating positive and negative sentiment scores played a huge role in determining and predicting how many stars a business would receive. While some supervised learning methods definitely outshined others, they all generally were capable of accurately predicting reviews to a certain degree,

affirming that these methods have become widely used and famously utilized because of their reliability and versatility. This project helped to demonstrate to us the significance of text mining; this was our first time experimenting with TF-IDF, and it helped us boost the model significantly. The language that one uses when describing their business is typically very crucial to the rating that they will give, and by quantifying this data, we were able to see a more clear and complete picture of the Yelp dataset.

6 Appendix

6.1 Appendix A: Data Dictionary

The dictionary of the variables defines the 17 variables in the subset of the data provided.

Variable	**Description**
User_id	22 character unique user id, foreign key to user.
Bus_id	22 character unique business id, foreign key to business.
Star	What the user rated the business in this given instance.
Useful	How many people found a given review useful.
Cool	How many people found a given review cool.
Funny	How many people found a given review funny.
Review	The text input from a given review.
State	The state where the business is located.
City	The city where the business is located.
MeanS	The average star rating a user gives.
SdS	The standard deviation of a user's star ratings.
Bus_Ave_Star	The business' average rating (rounded to the nearest half-star).
User_Review_Count	The number of reviews a given user has written.
User_Useful_Count	The number of reviews a given user has labeled 'Useful'.
User_Funny_Count	The number of reviews a given user has labeled 'Funny'.
User_Cool_Count	The number of reviews a given user has labeled 'Cool'.
Elite	Which years a given user was an Elite member.
User_Fans	The number of fans a given user has.
User_Ave_Star	The average star rating of all the reviews a given user has left.

Note: Only users with 150 or more reviews were considered. All businesses in the subset are located in Santa Barbara County.

References

Nabiha Asghar. Yelp dataset challenge: Review rating prediction. 2016. URL https://www.researchgate.net/publication/303331726_Yelp_Dataset_Challenge_Review_Rating_Prediction.

Tola Alade. Tutorial: How to determine the optimal number of clusters for k-means clustering. 2018. URL https://blog.cambridgespark.com/how-to-determine-the-optimal-number-of-clusters-for-k-means-clustering-14f27070048f.

Aman Kharwal. Word cloud from a pandas dataframe in python. 2021. URL https://thecleverprogrammer.com/2021/11/11/word-cloud-from-a-pandas-dataframe-in-python/#:~:text=A\%20word\%20cloud\%20is\%20a,based\%20on\%20natural\%20language\%20processing.

Jason Brownlee. A gentle introduction to k-fold cross-validation. 2018. URL https://machinelearningmastery.com/k-fold-cross-validation/.

Zulaikha Lateef. Knn algorithm: A practical implementation of knn algorithm in r. 2022. URL https://www.edureka.co/blog/knn-algorithm-in-r/.

Sunil Kumar Dash. A brief introduction to linear discriminant analysis. 2021. URL https://www.analyticsvidhya.com/blog/2021/08/a-brief-introduction-to-linear-discriminant-analysis/.

Benyamin Ghojogh and Mark Crowley. Linear and quadratic discriminant analysis: Tutorial. 2019. URL https://arxiv.org/pdf/1906.02590.pdf/.

Rohith Gandhi. Lsupport vector machine — introduction to machine learning algorithms. 2018. URL https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.