Modeling Business Success and Failure Using Yelp and Median and Mean Household Income Data

Taylor Gunter
University of Colorado- Boulder
taylor.gunter@colorado.edu

Devin Arnold
University of Colorado- Boulder
dear0350@colorado.edu

Matthew Coker
University of Colorado- Boulder
maco9370@colorado.edu

ABSTRACT

This is the final paper for group 18-- Yelp Us! for Spring 2018 CSCI 4502 Data Mining. The paper outlines the semester project, and includes sections describing the project motivation and objective, information about the data sets, and a summary of the previous work done on the proposed topic. This paper will also summarize the methods implemented, the results the methods produced, and and the application of the results. Works cited for code tutorials can be found at the end of the paper.

KEYWORDS

Data Mining, Yelp, Data, Income, Mean, Median, Modeling, Regression, Classification, Clustering, Cleaning, Processing, Python, Visualization, Failure, Success, K Nearest Neighbors, Naïve Bayes, Association Rules, Decision Tree Induction.

MOTIVATION AND OBJECTIVE

Client-service businesses are notoriously difficult. Margins are slim, risk is high, and customers can always find another establishment that offers the same service, but over time many businesses do find success. We are setting out to determine what makes a business successful, and conversely unsuccessful. Are there objectively better ways to run a business to increase the probability of success or is the secret just in the service?

Our objective for this project is to prove that successful businesses [1] have more in common than just good service, and unsuccessful business have more in common than just bad service. In order to accomplish this, we must determine what successful businesses have in common with other successful businesses; likewise, we must do the same for unsuccessful businesses. Additionally, we will integrate mean and median household income for all zip codes in the United States and use that data as an attribute to help in our discovery process.

DATASET SUMMARY

The primary data sets we are using are provided by Yelp and Kaggle. The data set that comes directly from Yelp is downloaded to Taylor Gunter's computer as a set of SQL tables. The data set from Kaggle is a set of .csv files and is also downloaded to Taylor Gunter's computer. Yelp also provides the data in a JSON format, which we will use to implement map visualizations if the project timeline permits.

We will be working with the .csv version of the data from Kaggle. There are seven files in the set, they

include: business info, business hours, reviews, tips, user data, check-in values, and a summary table that contains Boolean values of the attributes that each tuple contains. Our primary data will come from the following files: yelp_business.csv, yelp_business_attributes.csv, yelp_checkins.csv, and yelp_business_hours.csv. The secondary data set we are using is provided by the University of Michigan and is titled MedianZip-3.xlsx. The MedianZip-3.xlsx file has been integrated with the yelp_business.csv file to create a file called yelp_money_merge.csv. The following is a screenshot of the data organization, which illustrates the data attributes and file sources for this project. In this visualization the yellow highlighted rows are matching keys, and red highlighted rows are matching keys. The yelp_business_attributes.csv table is truncated to save space.

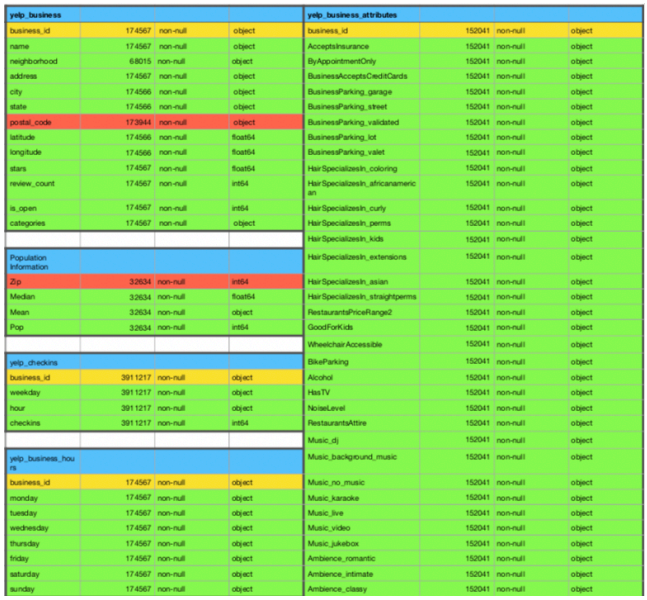

Figure 1: Data Summary

The data sets can be found at:

- https://www.kaggle.com/yelp-dataset/yelp-dataset
- https://www.yelp.com/dataset'
- https://www.psc.isr.umich.edu/dis/census/Features/tract2zip/

RELATED WORK

Since a source for our data set is Kaggle, previous work is abundant. We will summarize two articles in this paper. The first article is *Using Yelp Data to Predict Restaurant Closure* by Michail Alifierakis, who is a PhD Candidate at Princeton University. The second article is from Stanford University. It is titled *Predicting New Restaurant Success and Rating Using Yelp,* and is written by Aileen Wang, William Zeng, and Jessica Zhang. Additionally, there are several presentations on data mining techniques used specifically for this data, and we will summarize notable techniques found in the presentations. The first presentation is titled, "Data Crackers on Yelp Dataset". It was created by Prashanth Sandela, Vimal Gorijala, and Parineetha Gandhi. The second presentation is titled "Business Analysis Based on Location and Category". It was created by Keyur Mandani, Mikaelian Ovanes, and Hemanth Reddy.

Using Yelp Data to Predict BusinessClosure was written with the motivation to provide banks and investors with a model to predict whether loans and investments in a particular business are a good idea. The author initially attempted to use the Yelp rating system to predict business success. Yelp's rating system proved to not be useful however, because the data was not easily normalized. For this reason, the authors of the paper chose attributes including whether a business was a part of a chain, the density of local restaurants in the area, the number of reviews, star rating, price relative to other restaurants in the area, and finally the age of the restaurant. The primary issue the author faced in performing his analysis was strong (accurate) prediction of a business staying open, and weak prediction of a business closing. The author found the most important attribute leading to success was whether a business was a part of a chain, and the most important attribute leading to failure was business density in an area, where restaurants in restaurant-dense areas were more likely to close. The attribute the author wishes he would have used is population demographic, which is something we will be able to

determine with our secondary data set describing mean and median household incomes.

The second article is *Predicting New Restaurant Success and Rating Using Yelp* by Wang et al. This paper is by far the most academic, and describes many machine learning methods we had never heard of, but they do use chi-squared testing which was introduced in class. An unexpected benefit of this paper is it's analysis and use of previous work, so we multiple sources we were unaware of. Overall, this paper is very impressive, and will make sense of multi-characteristic attributes, and how to concatenate them. For example, there are attributes that can be converted to binary, then converted to decimal numbers to allow for east encoding. This paper found parking, attire, and ambience to be the most heavily weighted attributes in their chi-squared testing, and the one's that deserved the most attention.

The "Data Cracker on Yelp Dataset" presentation employed Naïve Bayes on user review data and classed the data into positive, neutral, or negative reviews. They also used decision tree induction, which they found to be useful for all types of attributes including, nominal, numerical, and ordinal. The decision tree was implemented in WEKA, which is not something we considered using, but will require further investigation, and will probably be implemented in our project. The final strategy this presentation covered was KNN and was also implemented using WEKA. This presentation provided good directional support for our project, but it will not be helpful as a results confirmation tool, because the work analyzed user review data using NLP techniques to extract information, which is outside of the proposed scope of our project.

The second presentation titled, "Business Analysis Based on Location and Category" focused on geo data and provided insight into ways to categorize the data. The group who created this presentation worked with SQL data, so their analysis was query based, but they were able to categorize the type of business more easily than pushing the data through a python parsing package. The most interesting outcome of this presentation is a line graph visualization that shows average 'star rating' by business category. There are two distinct low-rating outliers—acupuncture, and bingo halls, and two distinct high-rating outliers—auto detailing, and bartenders. Sentiment analysis based on user- review would be an interesting application to apply to outliers like these. If nothing else, the potential hypotheses that may be generated are interesting, but may not be testable, and could lead to unfounded speculation.

It is interesting to see the differing methods the papers and presentations employ, how they approach the problem, and how their results are affected. The most significant commonality between the papers is that the Yelp data set is a poor training platform, and classification (open or close) accuracy is low for both papers.

MAIN TECHNIQUES APPLIED

Cleaning

The first technique we had to apply to the Yelp data set, was data cleaning. Overall, the data was easy to work with since everything came prepackaged in .csv files. Because we needed to work with a large dataset, i.e. over one million rows, we integrated the yelp_checkin file with the yelp_business file. We joined on business_id, which resulted in the same business information being redundantly represented, but the information was tied to check ins now, so it would be good training data for clustering, and classification. In addition to integrating check-ins into the yelp_business data we deleted multiple attributes from the yelp_business data. The deleted attributes all contained information on location. We decided to focus on classification by stars, review count, and check-in counts.

The file that required the largest amount of cleaning, was yelp_business_attributes. Upon initial inspection of the data from that file we decided we wanted to use it for association rules-- primarily support, lift, and confidence, so we wanted Boolean values for each attribute in the data set. In its raw form the data contained 82 attributes with mixed types. The first

thing we did was fill NaN value with zero's. We made this decision, because if an attribute is not important enough to fill in, then it is assumed to be False or No. After replacing Nan elements with zero's. We filtered the data set for attributes that met minimum qualifications, which were that they had to contain values that could be converted to truth values that made sense. For example, if an attribute contained 'nan' or zero, and True, then the attribute was converted to 0 and 1. If an attribute contained 'nan' or zero, and False, then it was deleted because it would have contained 0, 0 for its elements , which would have been useless to us. We used the .unique() function to find values for attributes, and .replace() to replace elements with Boolean values. In addition to Boolean string values, there were ordinal string values. One example was the all ages allowed attribute which contained 0, True, 18, 21. We converted 18, and 21 to 0, and True to one. We made this decision by considering the attribute as a question, "Are all ages allowed?", if the return answer was 18, or 21, then the answer was "no, all ages are not allowed", therefore the answer is False, and set to 0. Nominal attribute elements also existed in the data set. An example of nominal data is the Alcohol attribute, which held 0, full_bar, beer_wine, and True; full_bar, and beer_wine were both converted to 1. Once we had all types converted to Boolean values we went through the data set, and found a strange discrepancy, which was attribute labels were swapped between neighbors. For example, GoodForKids, and RestaurantPrice, contained each other's data, so we swapped labels in the code, and not in the actual files.

The data cleanup we performed was integrating median, mean, and population data into the yelp_business data set, and yelp_business_attribute data set. We did this in the code by merging with zip code as the primary key.

In order to successfully apply the data mining techniques taught in class we wanted to create a visual summary of the data so that we could develop a high level understanding of what the data contained, and so that our results made sense when they were produced. (The following information happened during the same time as the data cleanup process). The information we wanted to summarize visually was on the distribution of the businesses across North America, and the distribution of open and closed businesses. Additionally, we wanted to see what the mean, and median income data, and population data looked like when integrated with the Yelp data. We chose to use a tool called Carto to visually summarize our data. The first visualization we produced was generated by the zip code attribute. Zip code became the primary key that we used to join different data sets together. A visualization of the businesses contained in the data set can be seen in Figure 2.

Figure 2: Business Location Summary

The distribution of businesses in this summary was surprising to us. We expected a more diverse sample size with less dramatic clustering. As a concession, many major geographic areas are represented, so the data does represent diverse populations. The yellow squares are the most densely populated areas, and the dark green squares are the least densely populated areas. Visually, Phoenix, Pittsburgh, Chicago, Montreal, Toronto, are all heavily represented. After getting a clear picture of what the data contains we wanted to see the distribution of open versus closed businesses, so we zoomed in on Phoenix, and produced Figure 3.

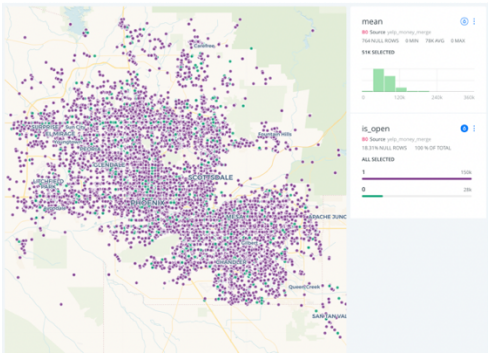

Figure 3: Business Open or Closed Map—Phoenix

This map shows approximately 85% of the businesses represented in the data are open, and 15% of the businesses represented in the data are closed. The final visual summary we wanted was to see how the income metrics distributed when projected over the data. Figure 4 shows the mean income distribution for the Phoenix area. The map space is the same as Figure 3.

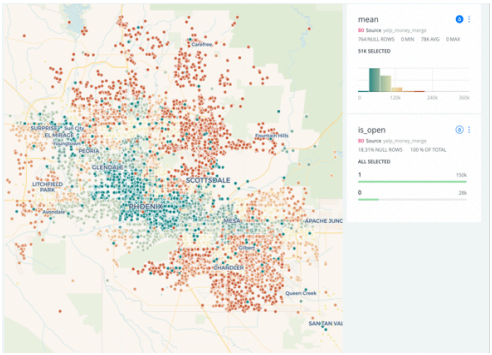

Figure 4: Mean Income Distribution-- Phoenix

Figure 4 shows clear lower income clustering around the Phoenix area, and higher income clusters in the surrounding metropolitan, and suburban areas.

Now that we had a visual summary of the data we wanted a statistical summary of the data grouped by business. The attributes we are focusing on primarily are stars, review count, average number of check-ins per business, the average of median income of the zip code the business is located in, and the average population surrounding the businesses. The average number of stars given to a business is 3.67; the average number of reviews a business has is 34.20; the average number of check-ins a business has is 1.83; the average median income by the zip code businesses are located in is $58,398.02; the average population by the zip code businesses are located in is 31,343.

All of the data was cleaned using Pandas, and cleaned data sets were produced and saved locally, so the original files did not need to be processed again. The new files are stored in our yelp-dataset folder, but can also be re-created because df.to_csv(<file>) are included in the data cleaning code.

Preprocessing

Our data preprocessing consisted of grouping data to get aggregated value averages and binning the data in preparation for decision tree modeling. The file produced by this technique is labeled average_groupby.csv Our group by technique was to group by business_id, and category, and take the mean of the other attributes, which included stars, review_count, is_open, Median, Pop, and checkins. We dropped the Mean attribute from data set, because Mean income data can severely skew results. For example, if 20 people are in a bar, and the average income of that population is 70k, and Bill Gates walks into the bar, then the average income of the population of the bar is over 1 billion dollars. Mean income is sensitive to outliers, so we dropped it. The average_groupby file was produced by merging checkins to yelp_business. Merging created redundant instances of each attribute from yelp_business, so using mean() in the groupby function, only actually produced the mean number of checkins for each business. The other preprocessing component was to bin the data in preparation for decision tree induction. We binned ‘review count’, checkins, ‘Median’, and ‘Pop’, with varying ranges, and twenty bins. We did this to match the decision tree data set taught in class. In the code it was done by creating unique ranges for all attributes that needed to be binned. The ranges were from 0 to the next whole number nearest to the max. Then we used pd.cut() to move the data into the created bins. This data set is labeled decision_tree_data.csv

With all of the data preprocessed we wanted to look at the distribution of review_count to stars, checkins to stars, median income to stars, populaton to stars, and review_count to checkins. The following figures show the graphical summaries of this data.

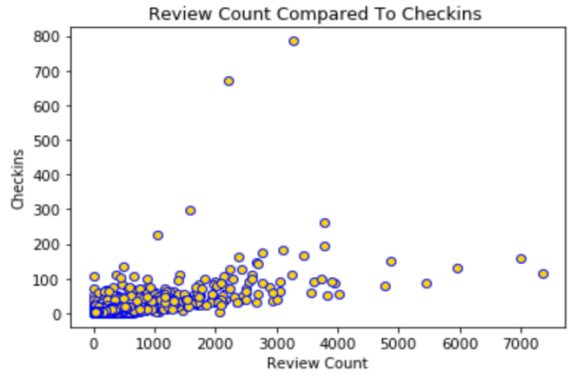

Figure 5: Review Count Compared to Check ins

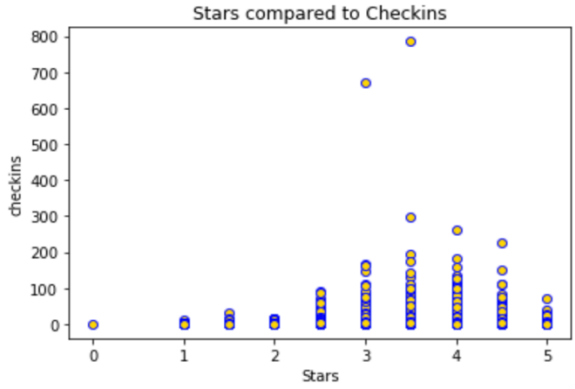

Figure 6: Stars Compared to Check ins

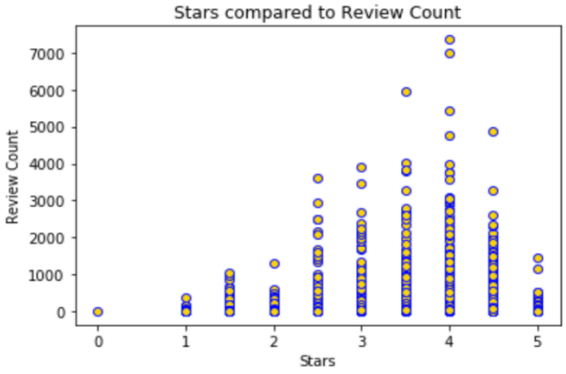

Figure 7: Stars Compared to Review Count

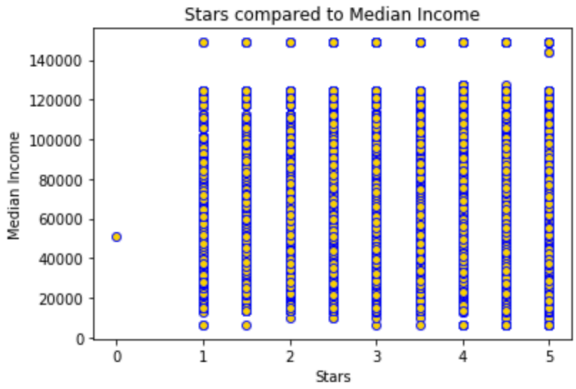

Figure 8: Stars Compared to Median Income

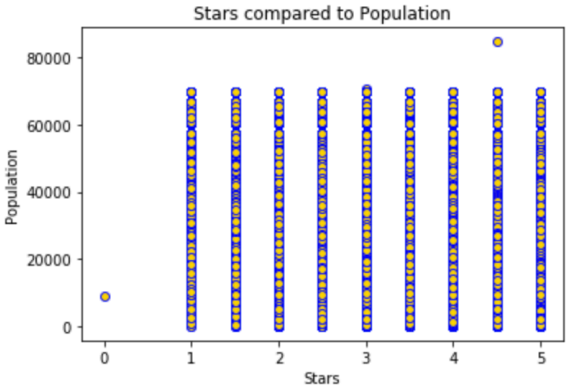

Figure 9 Stars Compared to Population

The x-axis in four of the five graphs is review count, so the plotting is discretely partitioned on the reviews 0 to 5 with .5 increments.

The first thing that can be seen from the scatterplots is that Figure 5 has strong clustering around the bottom left corner, but there is slightly positive correlation in the graph. Figure 6 and Figure 7 both show unimodal shapes with outliers at the tops of the graphs. The other notable observation is that population and median income are uniformly distributed across all review ranks. The single scatter plot points at the (0,0) values are an indexing marker, and are a reflection of the actual data.

Clustering

After cleaning, summarizing, and preprocessing the data we moved into the clustering and classifying of the data. The order of work was to create a decision tree model and use the attribute that was split on first as one

of the parameters in K Nearest Neighbors. The decision tree model produced a confusion matrix, and KNN also produced a confusion tree matrix. We also produced Naïve Bayes models, and Association Rules models. The decision tree modeling, KNN, and Naïve Bayes were created using ScKit Learn tools. The Association Rules model was creating using mlxtend tools. We were unfamiliar with how to use the tools properly, so we closely followed tutorials found online. The tutorials are cited at the end of this paper.

The decision tree model was done using the SciKit Learn library, and pandas. We use the checkin_money_merge.csv file, and set the decision features by extracting the attributes from the read in csv file into a new data frame. The features are 'stars', 'review_count', 'Median', 'Pop', and 'checkins'. Y was set as the target which is 'is_open', and X was set to the features dataframe. A training set was split off using entropy as the criteria for splitting. The model was fit using x and y training data. After that the remaining data was pushed through the trained model, and an accuracy score was calculated by dividing the y test by the y predicted. A confusion matrix was also generated. The results of the decision tree modeling will be shared in the key results section. The best split attribute is number of stars, so that is the base attribute we used for the KNN model.

The KNN model was built using SciKit Learn's KNeighborsClassifier library. The data set used for this model is checkin_money_merge.csv. The data was read into a dataframe, and a few additional cleaning and preprocessing operations were performed such as confirming there were not any null elements in any of the attributes. The data was then grouped by non-numeric attributes and spilt into a training set. The training data was modeled, and then the model was applied to the other half of the data, and a confusion matrix was produced, that showed the accuracy of the predictive model. From there a function that determines the optimal number of neighbors was implemented, so that over or under fitting does not occur. Finally, KNN is graphed for checkins versus reviews, and checkins versus stars. Both of these graphs show how similar data points will be classified i.e., whether they will be classified as is_open = True, or is_open = False. Both graphs, show five KNN clusters.

The third classification method we used was Naïve Bayes. The model was built using the SciKit Learn Naïve Bayes library, and is similar to the Decision Tree and KNN methods for generating the models. The data was spilt into training and testing sets, with the same numeric attributes pulled as features for the previous models, and 'is_open' set as the target attribute. The Naïve Bayes model predicted the success of a business, and the failure of a business, and broke out the mean stars and standard deviation for successful businesses, and unsuccessful businesses.

The final data mining method we used was Association Rules. We decided to use Association Rules because we are interested in what attributes are commonly found together, and in what frequency. We performed this technique using the MLxtend tool with the frequent patterns libraries with association rules, and apriori packages imported. We used the clean_test.csv file, which is a variant of the yelp_business_attributes.csv file. After some cleaning to fit the MLxtend parameters, we plugged the data frame into the tool. Because the data is so sparse we only looked at single antecedent and single consequent associations. Another consequence of working with a sparse dataset is that we had to tune the apriori algorithm way down to find frequent item sets with support of 0.001. If we went any higher the returned results were negligible. The frequent item sets table was plugged in to the association rules package and we got support, confidence, and lift for all available frequent item sets.

Key Results

This section will explain the results of the clustering and classification methods applied.

The Decision Tree model produced a confusion matrix, with the following results:

	Predicted Failure	Predicted Success
True Failure	11331	140710
True Success	697	840114

Figure 10 Decision Tree Confusion Matrix

The model was trained on half of the data, and the confusion matrix was produced from the untrained data. The accuracy of the model ranges between 84% and 86%. The best attribute to split on according to the decision tree model is star greater than 1. This is followed by review count greater then 31, Median income greater than $22,812 and Population greater than 34,253.

Results for KNN are based in splitting on number of stars a business has. The KNN model was used to predict whether a new business will fall into the open or closed (success or failure) category if it is added to the data set based on number of checkins and stars. We show how how a business would be classified based on review count and checkins. The following graphs show the KNN visualization. The right y-axis displays the classification, while the left y-axis, and x-axis are the attribute whose distances are being computed. The KNN SciKit library defaults to minkowsi distance, which is good for our data because the points are so close together.

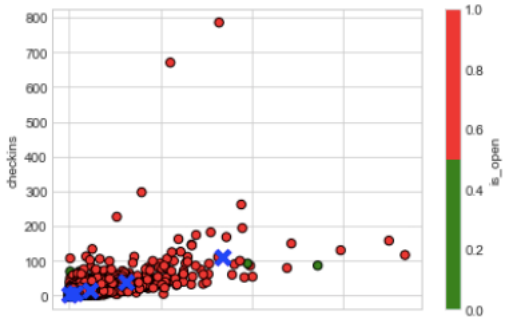

Figure 11: KNN Clustering of Checkins v.Review Count

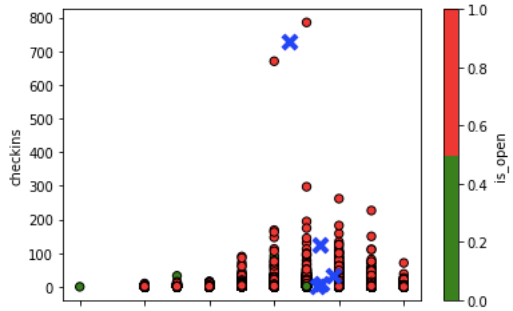

Figure 12: KNN Clustering of checkins v. Stars

The blue ‘X’s marks the location of the centroids in the graphs. The is_open = False classification is rare in the graphs as far as is visible, but it is probable that more exist. Unfortunately, with such large data sets overlap of data points is a consistent issue. The final result from the KNN is the optimal number of neighbors per cluster. This can be seen in Figure 13.

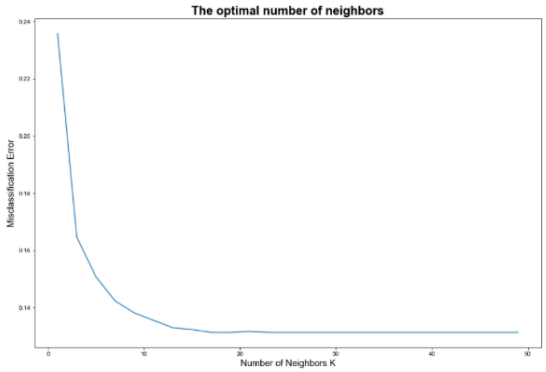

Figure 13: Misclassification Error and Number of Neighbors.

We can see from the line graph that the misclassification error decreases as the number of neighbors increases. The optimal number of neighbors according to the data is 17. Any more than 17, and the reduction in misclassification is insignificant and the data becomes too generalized. Any more than 17 and the training data is overfit, and misclassification of the test data increases.

The third set of results is from the Naïve Bayes model, which was used in classification to predict restaurant success or failure. The model indicates that out of 1,985,703 test elements 302,148 were misclassified. This is a 84.75% success rate, and a 15.25% failure rate. Since stars was the best attribute to split on according the Decision Tree model we used average stars as the metric for success and failure. Figure 14 shows the results for this measure.

```
mean_stars_success= 3.58
std_stars_success = 0.79
mean_stars_failure = 3.32
std_stars_failure = 1.06
```

Figure 14: Average stars for successful businesses, and failed businesses with standard deviation included.

Figure 14’s measure can lead to several conclusions. The first conclusion is that there is not actually any difference between successful and failed businesses.

Null hypothesis testing could produce a confidence interval that would tell us if there is actually a difference. The other conclusion we could reach, which we believe is more likely is that it is very difficult to get over 3.5 stars on average and it is actually a good marker for success because only successful businesses can get there.

The final set of results is for Association Rules. We learned that sparse datasets need lower support thresholds to produce enough results to make comparisons. The data shows low support for all relationships, but strong confidence, and extremely strong lift for all relationships that make the cut. Low support with high confidence and lift can show hidden relationships that may not be immediately obvious. The lift in all of the Association Rule relationships produced by the model is over 100, which gives pause, because in order for lift to be valid the features must be independent. An example of such a relationship with high lif is {BestNights_Monday, GoodForMeal_Breakfast} -> {GoodForMeal_Dinner}. Are good breakfast, and good dinners independent? It's hard to tell. The thing that is promising though is that a place that is both good for breakfast and good on Mondays probably does go together and does point to a successful restaurant because people are running late on Mondays and may have eaten out over the weekend, so they are less prone to eat out again early in the week. The other relationships commonly seen together are coat_check and valet_parking, which indicates a fancier business in a busy area, or a densely populated area.

Application

We are treating the application section of this paper as if we are a consultancy delivering a list of recommendations to a business to help they gauge whether they will be successful or not, or what they need to do in order to not fail. The first thing we will tell a business is that their star rating really matters. It is the first criteria that is split on in a decision tree, and the difference of star rating between a successful business and unsuccessful business is smaller than one might think. A business needs at least 3.5 stars to push them toward success. Less than 3.5 stars categorizes a business as one that shares attributes with businesses that fail. Another key application is that the location of a business matters, but not in the way a business owner may think. The criteria is oriented more towards what someone doesn't want than what someone does want for a location. That being stated, a business owner wants to stay away from zip codes with populations under 34, 253 and zip codes with Median incomes of less than $22,812 dollars Other criteria that a business owner may want to consider if they have to develop a business from scratch is that coat check service and valet services go together. This could indicate a fancier establishment that meets all their customers needs and is big on hospitality. Another combination items to consider is 'good for breakfast', 'good for dinner', and 'good for Mondays'. A possible reason for this is that the business isn't too fancy, is convenient, and does a lot of different things well. Possible businesses like this may be a Diner is a densely populated are. Finally, if a business owner wants their business to succeed they should plant the business in a densely populated area with higher median incomes. This may seem like common sense, but the data is there to back it up.

Conclusion

The yelp data set is interesting to work with and allows for different data sets to be joined easily. Any data set that contains zip code information can easily be appended to many of the tables.

As a group, we left a large portion of the packaged data set untouched because it contains reviews by customers, and customer ids. The reviews portion of the data would be good for sentiment analysis, and natural language processing,a nd could be used to predict star rating.

A major drawback of not integrating any NLP into this project is that we were unable to do any data mining on the categories of business. It would have been beneficial to see how certain groups of businesses rate amongst each other, and then normalize that within the data.

Overall, we were able to create general insights that will apply to all types of businesses in all geographical locations. The generality of our models will be beneficial because a business could use them to no matter how many attributes they want to analyze, or if they fit in the categorical datasets.

Sources

The Decision Tree Model was made with SciKit Learn's tree library. Our code and tool implementation is based on Ben Alex Keen's blog post "Decision Tree Classifier In Python Using SciKit Learn".

The KNN model was made with SciKit Learn's KNeighborClassifier. Our code and tool implementation is based on Robert Miller's blog post "K nearest neighbors, it's purpose and how to use it".

The Naïve Bayes model was made using SciKit Learn's Naïve Bayes library. Our code and tool implementation is based on Martin Müller's blogpost "Naïve Bayes Classification with Sklearn".

The Association Rules model was made with MLxtend's Frequent Patterns library. Our code and tool implementation is based on Chris Moffitt's blogpost "Introduction to Market Basket Analysis in Python".

Works Cited

Keen, Ben Alex. "Decision Tree Classifier in Python Using Scikit-Learn." *Ben Alex Keen*, 31 May 2017, benalexkeen.com/decision-tree-classifier-in-python-using-scikit-learn/.

Miller, Robert. "K Nearest Neighbors, It's Purpose and How to Use It." *Towards Data Science*, Towards Data Science, 3 Apr. 2017, towardsdatascience.com/k-nearest-neighbors-its-purpose-and-how-to-use-it-36fa927acc64.

Moffitt, Chris. "Introduction to Market Basket Analysis in Python." *Practical Business Python Atom*, 3 July 2017, pbpython.com/market-basket-analysis.html.

Müller, Martin. "Naive Bayes Classification with Sklearn – Sicara's Blog." *Sicara's Blog*, Sicara's Blog, 28 Feb. 2018, blog.sicara.com/naive-bayes-classifier-sklearn-python-example-tips-42d100429e44.

Rundell, Michael. "Data Mining in Python: A Guide." *Springboard Blog*, 25 Mar. 2018, www.springboard.com/blog/data-mining-python-tutorial/.