Dataset
We used the Yelp Dataset Challenge business.json which included information on businesses.

PREPREOCESSING

Unnesting/Expanding and Grouping Data
The 'hours' and 'attributes' columns' data were json files nested within the dataframe. Thus, we needed to unnest them. We then looked at the number of observations by region in order to see which regions we wanted to focus on. We deleted the observations of the states with less than 500 businesses, because that had too few data. We wanted to see data by region and wanted to have at least 5000 observations for each region, so we grouped states that were similar in region together: South Carolina + North Carolina → Carolina and Wisconsin with Illinois.

Deleting 90% Missing Data and Identifying "Anomalous" Patterns
Knowing that we didn't need the observations that had more than 90% of the data missing, we deleted all those from the dataset. We also looked at the ratios of data that was missing to see if some columns were basically missing the same data. We did it because we thought that if there were multiple columns that had the exact same percentage of data missing that it could be an indication that anomalies existed. However, upon analysis, we concluded that the columns missed the same percentage of data because they were filled out for the same observation, but the values within each observation for each column was different. Thus, we were not able to group or delete columns to reduce the dimensionality of the data.

Dividing the Data by Region (primarily states)
The next step was to divide the dataframe by region. Our plan is to do analysis by region. However, we also wanted to create some models just to see what were some of the complications of our dataset and see via experience what was necessary to clean parts of the data. As a test run, we arbitrarily chose the Ohio Dataset which we call ohDfPractice.

Dropping Dimensions/Columns of the Ohio Dataset
We decided to drop ['postal_code','city', 'business_id', 'state', 'name', 'neighborhood', 'categories', 'Friday', 'Monday','Tuesday', 'Wednesday','Thursday', 'Saturday', 'Sunday'] for a couple of reasons. We did not need postal_code or state because we already had the latitude and longitude and new the dataset was on Ohio. We dropped city and neighborhood and name because we really didn't want that "fine grain" of a detail, especially because we are looking things by region. We dropped categories because it would complicate our model. We would need to regroup each category and find any repeating categories, but also do it for the rest of the regions because we want things to be consistent. Also, we didn't care much about the type of business because the aspect of the "type" of business we cared about was the region mainly. We also deleted the days which included the open hours of the businesses, because we initially thought it was too hard or computationally intensive to parse. However, I think we will make separate columns (i.e. Friday-Breakfast, Friday-Lunch, Friday-Dinner) and use binary numbers to indicate whether the restaurants were open during those time frames.

Filling NaN values of the Ohio Dataset
We also made a lot of categorical data into binary formats for 'Wifi', 'Alcohol', 'Noise Levels', 'Restaurants Attire' and 'Restaurants Price Range'. For 'Wifi', 'Alcohol', 'RestaurantsAttire', and 'NoiseLevels', I filled it with a 0, because the assumption was if there was no information available for it, then we thought the consumer would assume it did not have those attributes. We also grouped 'paid' and 'free' of 'Wifi' together as a 'free' or binary 1 value. We grouped 'beer_and_wine' and 'full_bar' as 1 for 'Alcohol'. For 'Restaurants Attire' we grouped 'dressy' and 'formal' as 1. For 'NoiseLevel' we grouped 'average' as 'quiet' (value 0) and 'very_loud' as 'loud' (value 1). We also binarized 'stars' column. If a business had more than ⅗ stars it was considered as high (1), otherwise (0). However, in retrospect this binarization scheme may be unnecessary. I think in the future we will not necessarily group data in such an arbitrary manner and not binarize them but keep the categories as they are.

Additional Parsing of the Ohio Dataset

I concluded that we didn't need the ['index', 'latitude', 'longitude'] columns because we didn't really care about 'latitude' and 'longitude' information, particularly because we aren't visualizing the data with that fine grain of a detail geographically. I then looked at the ratio of missing data and deleted all the columns that had more than 60% of the data missing.

Merging Parking Data

To reduce the number of dimensions and reduce repetitiveness, I combined all the columns regarding car parking together. Thus the new 'Parking' column would indicate whether parking was available, regardless of whether one had to pay for valet or park on the street or not.

Filling NaN Values for data with less than 40% of missing

For the columns ['BikeParking', 'BusinessAcceptsCreditCards', 'GoodForKids', 'RestaurantsTakeOut', 'Parking'] we filled all the NaN with false because we assumed that the customers should assume there 'No' for the attributes if they were not present. However, I think for the future we will change this to a separate category as 'Unknown' rather than False.

INITIAL MODEL & PRELIMINARY RESULTS

Variables for Our Initial Model

We have 13 Dimensions in the dataset. 12 of which are the predictors: is_open, review_count, Alcohol, BikeParking, BusinessAcceptsCreditCards, GoodForKids, NoiseLevel, RestaurantsAttire, RestaurantsPriceRange2, RestaurantsTakeOut, WiFi, and Parking. The 'stars' is our target in this case and it is binary info**.** We split the dataset into 80% training and 20% test data randomly to check the performance of the data.

Decision Classification Tree

We wanted to create a decision classification tree to identify what factors or questions could potentially be important. We want to suggest to businesses what are important features for a successful business. The training data error was 0.81347209515096064 and test data was 0.8005489478499542. So it seems that our performance is great. However, when looking at this one model of the tree, something seems to be "off" because the first question asked is Parking. So, this seems to show that our model is not necessarily the best. However, the top questions asked were: 'Parking', 'review_count', 'Noise_Level', 'BikeParking', 'RestaurantsPriceRange2', and 'Alcohol'. The max_depth was 8, because 10 went to granular and had all its leaves have 1 for its samples, but we plan to override this by setting the min_samples_split to a higher value, rather than its default, 2.

Random Forest Model

We ran a random forest model with the same dataset to see the variable importance of the categories. The Mean Squared Error (MSE) resulted to be 0.173300937507 and the sqrt MSE was 0.41629429194572937. The top five variables were review_count, RestaurantsPriceRange2, GoodForKids, is_open, and WiFi. By doing a side by side comparison with our decision tree, it seems that review_count and RestaurantsPriceRange2 are important variables.

Future Plan/Complications

We plan to refine our dataset by not binarizing all the categories and labeling the NaN values as an 'unknown'. We also plan to add the hours information by days. However, we fear that that would complicate our model, so we may have to reduce the number of columns and focus on specific ones. We will also try a 'reverse' model by setting the target as review_count. We think we will run multiple trees to see what are some commonalities each tree has which could perhaps reinforce the idea that some factors are more important than others. If time provides, also may look at each region or combine all the region to do a comparative analysis of the models. We ran a regression Tree Model in case we wanted to use it in the future. In this case it was ignored/not helpful because our target was binary (as can be seen at the y_test:stars vs predRegr plot at the end). However, if we do not use binary categories we believe that this model could be useful in the future and plan on using this as well.