**BUDT 758T Class Project**

**Project Report : Airbnb.com Prediction Contest**

**Group 30: Nishi Intwala, Arkav Banerjee, Kritika Panjiar**

**Introduction:**

This project aims to predict the popularity of an Airbnb Listing. We have used the Airbnb dataset which has 70 attributes. This prediction will help Airbnb sort out newly added listings available into potentially popular listings based on the given attributes. This will also help potential homeowners who are looking to rent out their homes to Airbnb, to predict whether their listing will be profitable or not.

We can partition our project into three main categories:

1. Exploratory Data Analysis and Feature Engineering

2. Model Evaluation

3. Modeling

**Exploratory Data Analysis and Feature Engineering:**

We started off our initial analysis with some basic EDA. We tried to figure out the attributes which might help us predict the popularity. So, we took a look at various variables such as number of days Airbnb is available for, the price of an Airbnb listing for each night, the most popular states for Airbnb listings, so on and so forth.

Some of the interesting insights we came up with are as follows:

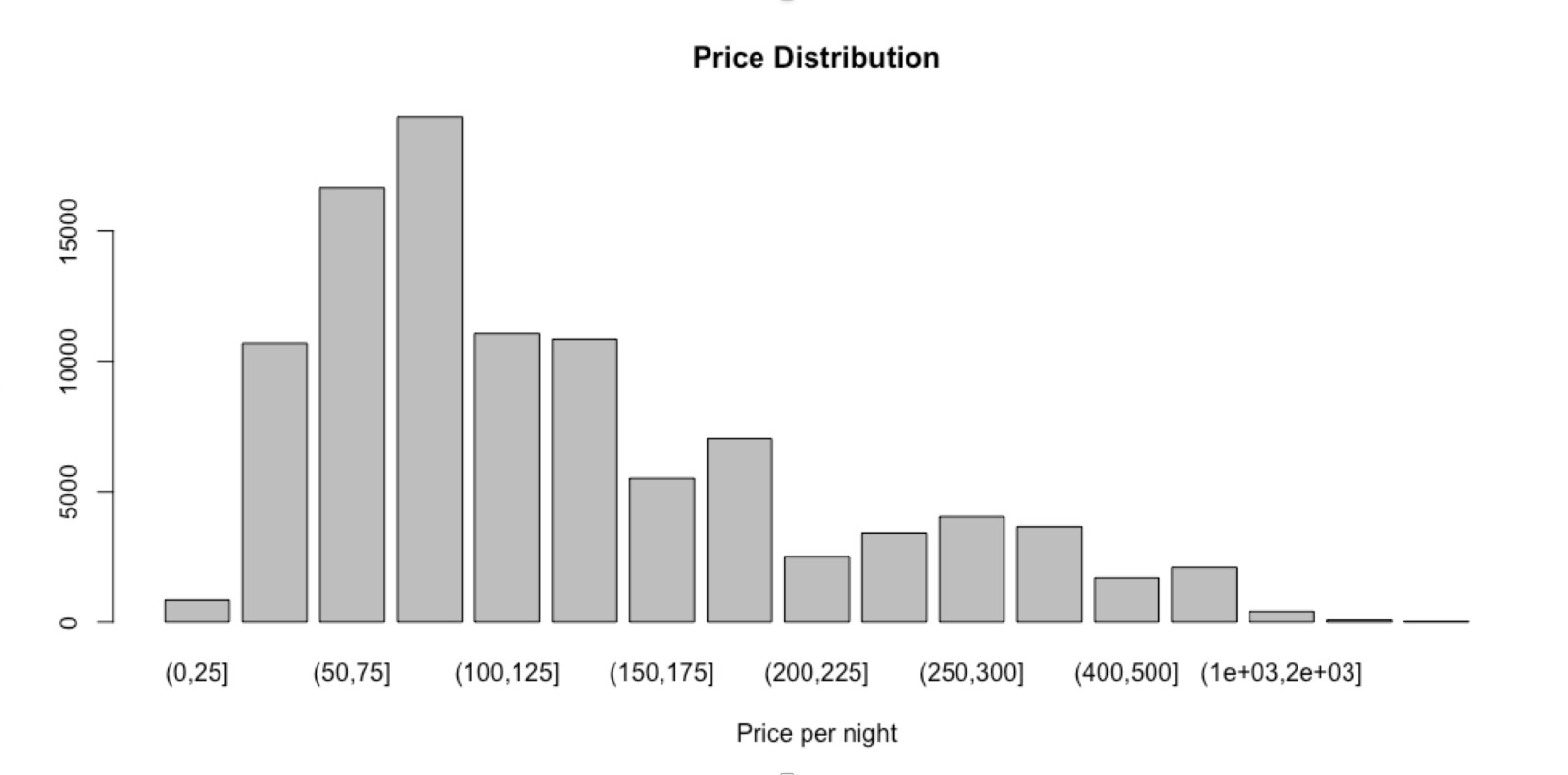


Figure 1 : Price Distribution of Airbnb Listings by **Kritika Panjiar**

The price per night was more common between $75 and $150. But the interesting point is that there are many listings with prices as high as even $800 per night! This indicates that people will pay very high prices for a good Airbnb listing. There were a few outliers with $10,000 per night.

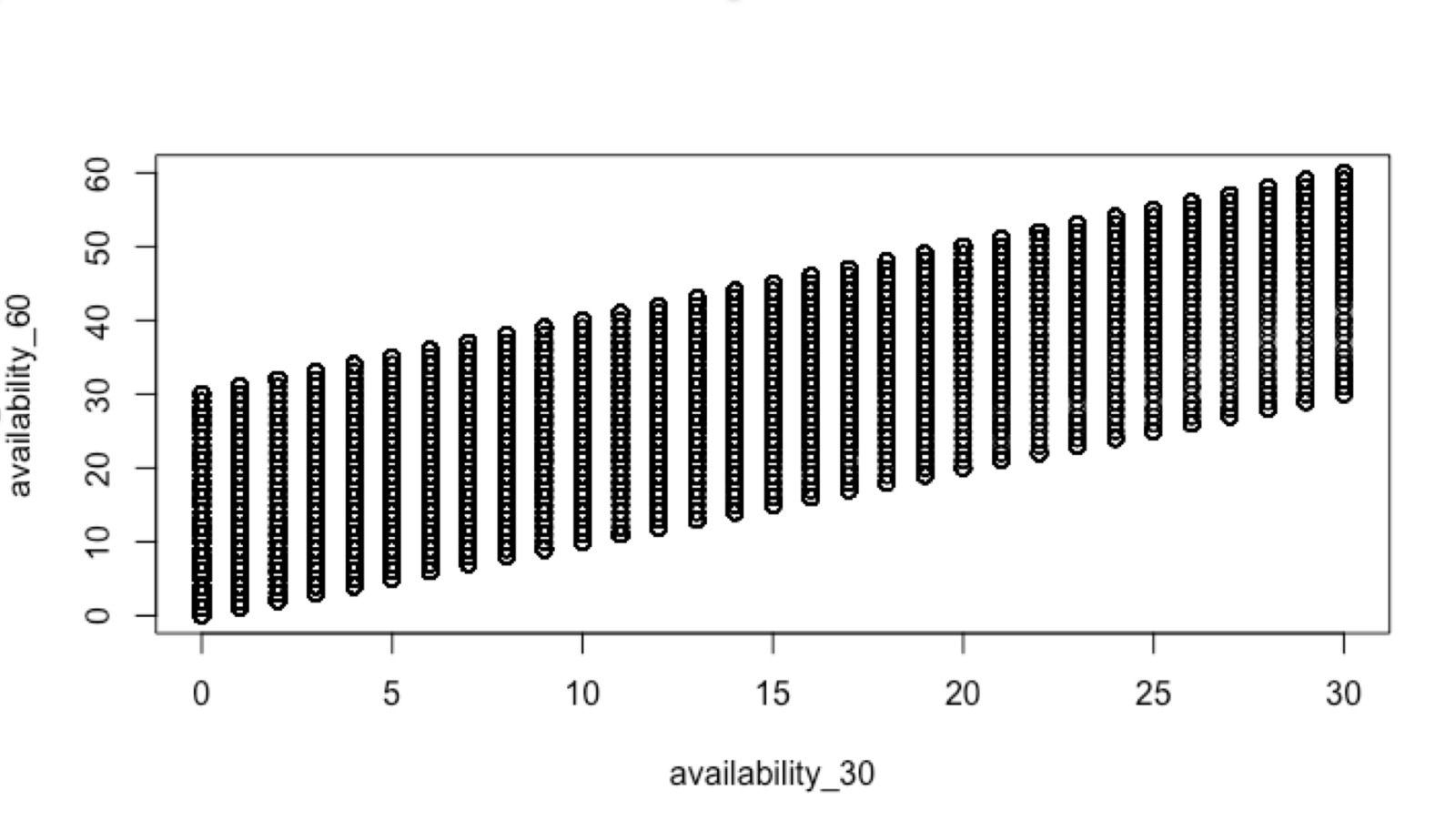


Figure 2 : Correlation between listings available for the next 30 and 60 days by **Nishi Intwala**

We can see in the above plot that the availability of Airbnb for the next 30 days and next 60 days is highly correlated. Their Correlation is also represented by their high correlation percentage which is 92%.

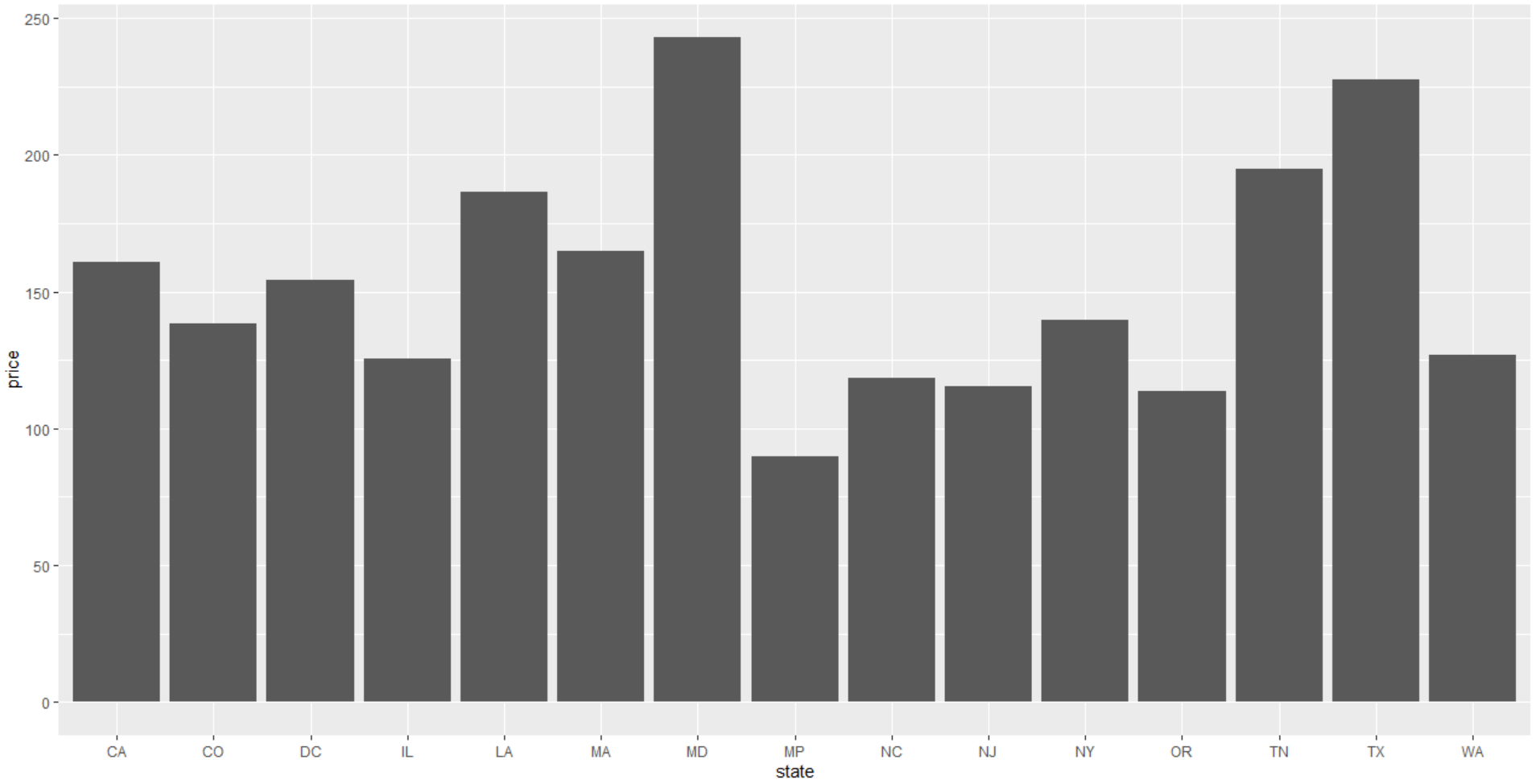


Figure 3 : Average Price for each state by **Arkav Banerjee**

We also took a look at the average price for each state. We expected that the average price per night for states such as California and New York would be much higher than others, but we were surprised to find states such as Maryland and Texas had higher average prices. It would be interesting to dig deeper.

After we were done with our initial analysis, we moved on to the Data Processing and Feature Engineering. Both these tasks went hand in hand as we tried to determine attributes that might be important for our models. First, we checked the nulls for each column and eliminated all columns which had more than half the values missing. We removed 19 rows from the training data which had jumbled data across all the columns, 2 rows belonging to Mexico and a state row belonging to Mexico. Therefore, we removed 22 rows in all. We imputed missing values for cleaning fee and security deposit to $0 because we assumed that a missing value indicates that the listing might not have these fees. The zip code column had over 16000 nulls. To impute these nulls, we sorted the entire data in ascending order by the latitudes and then imputed missing zip codes using forward fill. We believed that similar locations would then get clubbed under the same zip code. For the remaining numeric columns, we took the mean to impute missing data. We moved on to categorical columns, where we cleaned the states column which had different codes for the same state and imputed missing values by looking at the city name. We imputed remaining categorical variables using the mode. We created 2 new columns amenities\_count and host\_listings\_count which had a count of all the values in each list for amenities and host\_listings. We felt that a count of these lists would be a significant attribute in determining high\_booking\_rate. Once our dataset was ready for analysis, we decided to look at the MeanDecreaseGini for selecting our attributes. Table 1 shows the Gini Indices for the top 10 attributes. (refer to appendix)

We also used text mining to come up with a variable that would show the count of public transportations available near the listing, but that did not help us in our analysis.

**Model Evaluation:**

We used a random seed of 46747 to partition our training data into a 90:10 split. We believed that this would give our models more rows to train on and then we would still get 9998 validation rows to test our models on. Since our test data also had 12208 rows, we felt that 9998 rows for validation was sufficient. Our next step was to choose the baseline accuracy. Our validation data had 7484 values as 0 and the remaining as 1, thus giving us a baseline accuracy of 74.86%. We decided to work on less complex models first and then move higher up the ladder. We decided on a common list of attributes as deciphered by the Gini index and we each ran different models on the same set of attributes. We chose accuracy as our mode of comparison between each model. We kept adding and dropping attributes for all models till we reached the highest accuracy possible(We used our first 2 submissions to evaluate our feature selection and check accuracy on the test data) To achieve higher accuracy, we added and dropped various attributes and tuned parameters for each model to achieve higher accuracy. For each leaderboard submission, we trained our models on 100% of the data using the same parameters that we finalised before submitting our predictions. This would enhance our accuracy since the model could train on more data.

**Modeling:**

This was our main focus of the analysis. We tried out many different models for our analysis such as Logistic Regression, Decision Trees, KNN, Ridge and Lasso Regression. But we narrowed down our main modeling on Ensemble Methods since they kept giving us significantly higher accuracies than the rest.

Our modeling process can be split into two parts – Initial Models and Final Models. Our modeling process was in parallel with our feature engineering and data processing tasks. Hence, initially we ran some models which had minimum pre- processing done. Our Initial Models did not include any of our feature engineered attributes or text mining attributes. We also did not perform any hyper parameter tuning for our Initial models. Table 2 describes the different types of models we tried. (refer appendix)

The following steps will briefly describe our modeling process:

* We first ran a Logistic Regression as we believed this would be the easiest to execute and would give us a current standing.
* We then ran Decision trees where we used the same attributes as Logistic Regression. We could run this without much pre-processing. Unfortunately, we ran into multiple errors and hence decided to move ahead.
* Our next model was K-Nearest Neighbors which gave us a low accuracy
* We also ran a Ridge and Lasso Regression, but this gave us a lower accuracy as well
* We then decided to test out Random Forests for bagging and boosting models. Bagging models gave us the highest accuracy of 83.82% on the test data set during our third submission and was our breakthrough model. We saw that Boosting gave us a similar accuracy on our validation data. We affirmed our hunch that Bagging/ Boosting would be the best models to go forward with.

After the third submission, we confirmed our attributes and worked only on hyperparameter tuning to increase our accuracy further for bagging and boosting simultaneously.

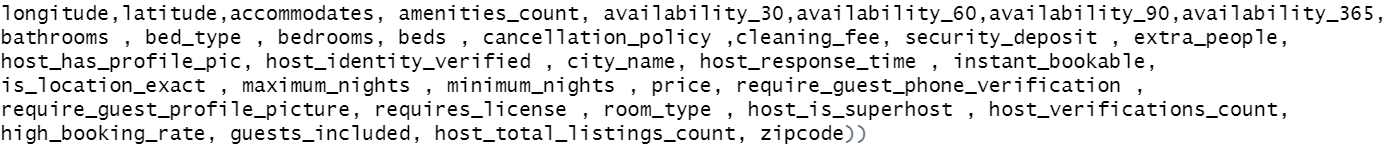


Figure 4 : List of final attributes used by **Arkav Banerjee**

We used TuneRF functions and custom loops to choose the best n.tree and mtry values for Bagging models. For Boosting we had more hyperparameters we could fine tune to get higher accuracy. Table 3 summarises our tuning process.

**Conclusions and Further Scope:**

Since Boosting took less time to train and had more hyperparameters to tune, we chose Boosting as our final model. This gave us a final accuracy of 84.1% on test data and we came 4th in the contest. Since we had only 3 team members, all of us had to put in complete effort. Arkav took a lead for the Data Processing Tasks while Nishi took a lead on Feature Engineering. Kritika was responsible for the EDA, documentation and presentation. We all contributed towards making the report and modeling as the latter required fresh ideas and multiple testings. Overall, each task was thoroughly participated in by all the team members to give a full joint effort. As future scope, we can play around with more interaction variables in our model and we can perform clustering to understand correlations between variables. We can also make use of the FSelector library as another way to select attributes for our model and create more attributes that may improve accuracy.

**Appendix:**

Table 1 : Gini Indices for top attributes

|  |  |  |
| --- | --- | --- |
|  | **Variable** | **MeanDecreaseGini** |
| **1** | Availability\_90 | 2291.869776 |
| **2** | Availability\_60 | 2257.896672 |
| **3** | Availability\_30 | 2212.768571 |
| **4** | Availability\_365 | 2127.249651 |
| **5** | Minimum\_nights | 1846.450608 |
| **6** | Longitude | 1845.680497 |
| **7** | Price | 1820.429033 |
| **8** | Latitude | 1779.921171 |
| **9** | Zip code | 1673.387616 |
| **10** | Extra\_charges | 1630.947425 |

Table 2 : Different models and accuracy

|  |  |  |
| --- | --- | --- |
| Models | Pre-Feature Engineering Models | Post Feature Engineering Models |
| Logistic | 76% | 78.83% |
| KNN | 75% | 75.5% |
| Ridge | 78% | 78.13% |
| Lasso | 76.1% | 76.70% |
| Bagging | 83.51% | 83.8% |
| Boosting | 83.84% | 84.4% |

Table 3 : Hyperparameter Tuning

|  |  |  |
| --- | --- | --- |
| Models | Hyperparamters tuned | Final values |
| Bagging | n.trees , mtry | n.tree=5 ; mtry=7 |
| Boosting | n.trees, interaction.depth, distribution, cut off | n.trees=1500 ; interaction.depth= 7 ; distribution = Bernoulli; cut off = 0.49 |