**BUDT 758T: Group 8 Final Project Report**

**Group Members: Xiaoyou Zhou, Haibo Yu, Luying Lou, Qian Jiang, Praneeth**

**1. Basic Data Cleaning & Model Making**

**1.1 Cleaning**

For the initial cleaning part, we kept things simple. We concentrated mainly on two things:

**1.1.1 Handling NULL Values:**

For the columns with less than 20 Null records we imputed the missing values with the majority value or the “mode”. For the columns with 100-200 Null records we removed these rows from the training data. For the columns with too many Null records we couldn't just delete or impute the majority class. So, we decided to impute the values using Logistic Regression and Predictive Mean Method from the MICE package. We also implemented a few techniques like using the missingness, replacing with 0 etc.

**1.1.2 Data Type Consistency**

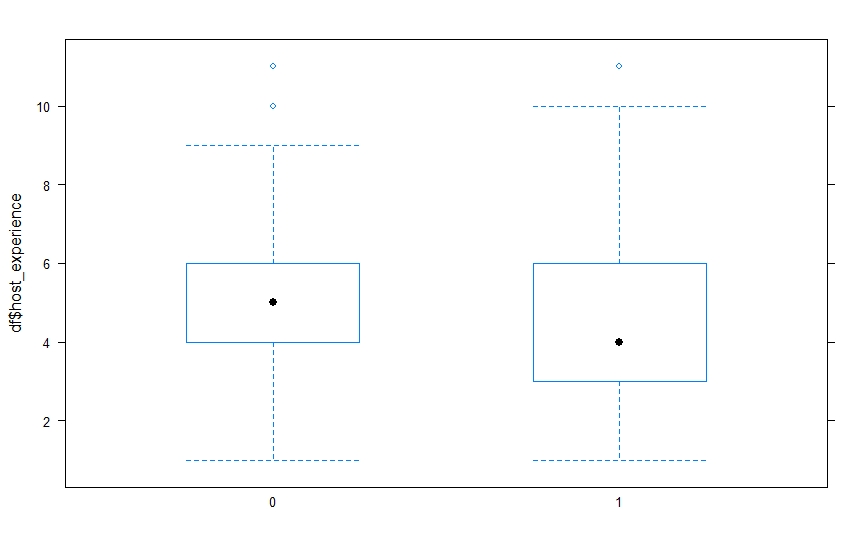
The data consisted of Categorical, Numerical and Text Features combined.

For the Categorical Features like property\_type, host\_response\_time etc. we removed any outliers and converted them to factor variables.

For the Numerical Features like price, host\_response\_rate, cleaning\_fee etc. we converted them to numeric variables by removing the “$”,”%”,”,” from the strings.

**1.2 Feature Engineering**

1. host\_experience: From the host\_since column, we have extracted the years in each record and calculated the host\_experience by subtracting it from 2020. We felt like this would be one of the ways that we could use the date metric.



(Praneeth Yaramosu)

2. first\_review: from the first\_review column, it was modified to “OLD” or “NEW” to use the date metric.

3. availability: There were 4 availability columns; 30,60,90 and 365. All the 4 were combined together to the availability column as “HIGH” and “LOW”.

4. Binning: Columns like property\_type, cancellation\_policy etc. had a lot of factors with very little frequency. So, we binned the low frequency factors into the higher frequency ones.

**1.3 Modelling**

Firstly, we started with basic modelling using logistic, lasso and ridge regression. We fine tuned the model using grid search with lambda values. The best cutoff was then determined using the ROCR curve. The model that produced the best accuracy was the Lasso model with 77% accuracy.

**2. Second Data Cleaning, Random Forest and Variable Importance**

**2.1 2nd Data Cleaning:** based on the concatenate of both train data and test data to avoid issues we had before.

**2.1.1 Add new features:**

* Amenities\_count: We decided to use this text variable as a numeric feature, by counting how many amenities each listing has.
* Description\_length: We believe the Description can reflect the hosts’ passion and level of responsibilities. So, we split each word by blank and then count the unique values.
* Weekly\_cheaper, Monthly\_cheaper:We used the ifelse function to check if the weekly and monthly prices are cheaper than the (daily) price.

**2.1.2 Delete some features:**

The features we deleted were Availabilities and Accommodates. It is necessary to keep the availability\_30/60/90/365 indicates availability in different time ranges. Accommodates have similar functions as guests\_inclueded, so we only keep guests\_inclueded.

**2.1.3 Change Cleaning Method**

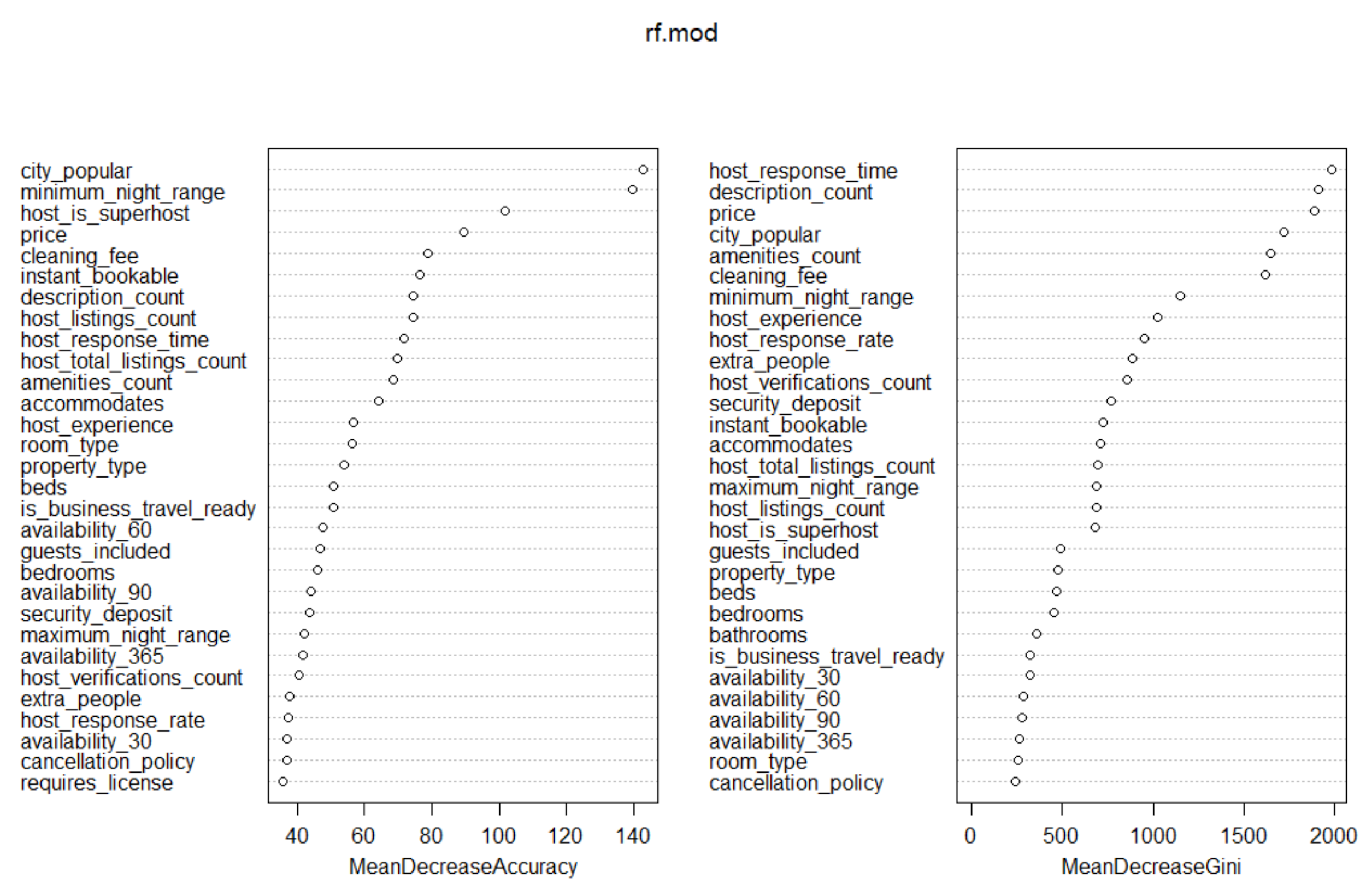
We corrected this mistake by leaving the bathroom number unchanged as floats. For the Cleaning\_fee, instead of drop NA, we impute the mean.

**2.2 Feature Engineering**

* Property Type: grouping into : Big house, Tiny house, APT, Fun living, Resort, Economic Living.
* Host\_experiene: We change the data type to date and extract the year, then use 2020 to minus the first review year.
* Popular City: We filtered and ranked the top 10 cities of all high booking rate listings as Popular\_City list, making it as a binary variable.

**2.3 Modeling**

We used the Random Forest Model, adjusted my mtry to 7, and gained a 81% ACC. I generated a Variable Importance plot for our nest phrase



(Qian Jiang)

**3. Feature Engineering & XGBoost**

**3.1 New features**

First of all, we add some new numeric columns: Average number of beds per person, average price per person and average bathroom per person. More specifically, the average number of beds per person can be a useful metric to measure whether a hotel is big enough for customers. Then, I also introduced the average price per person, because the number of persons a house can hold is quite different. Also, I replaced the share\_bathroom with a new feature called the average bathroom per person.

Second, I also explored some text columns, such as the amenities. I picked two important amenities, Air conditioning and WIFI. Then I created two dummy variables, wifi\_ornot and air condition. Another feature I created is transit, which describes the transportation of the house. If the hotel has transit descriptions then I set the transit variable to 1 because the house is more likely to have a good location.

**3.2 Data cleaning**

For train data, after creating new variables price per person, I noticed that there is ‘Inf’ in the new feature columns we created. I replaced them with some meaningful values. After that, I also checked the null values in the train dataset and used a subset function to remove all the rows containing null values.

For test data, I used a table function to see all the unique values and then I found that there is an invalid character ‘t’ in this numeric column. I also found sentence value in the city\_name column and host\_listing\_count column. I manually replaced them with random values in that column.

**3.3 Model**

I chose XGBoost, which is an algorithm good for data with a large number of variables, as my final model. First of all, I separated train data and train labels. After that I used a function to change train data and train label to xgb matrix so that they can be processed more efficiently. In terms of the model parameters, I chose ‘gbtree’ booster, a maximum depth of 5, learning rate of 0.001 and minimum loss reduction of 3. Using these parameters, I then trained the xgb model with 10000 rounds. After I fit the model, I tested the model on the validation data and got an accuracy of 79% when choosing a cutoff of 0.5.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

(Luying Lou)

**4.** **Feature Engineering and Modeling**

**4.1 Feature Engineering**

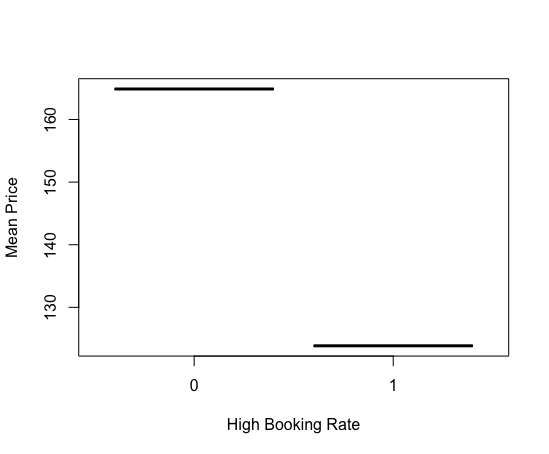
* ‘availability\_30’, ‘availability \_60’, ‘bedrooms’, ‘bathrooms’, ‘accommodates’, we filled the null values with the median, because we found there were some outliers.
* ‘maximum\_nights’, ‘minimum\_nights’, if the values of them were higher than 365, we just changed these values to 365, and then filled the null values with mean, because they do not have extreme values anymore.
* ‘Minimum\_price’, we got it from the price multiplied by minimum nights, we think it might be helpful, because it was the value customers must pay, so we tried it.
* ‘weekly price’, we filled the null values with price multiply by seven. And like states, first we make the character to lowercase and then divide them by the listing with most states, others just changed to “others”.

**4.2 Modeling**

* Random Forest: tuning hyperparameters such as minimum nodes, number of trees, sample fraction and mtry. Accuracy: 82%

**4.3 Visualization**

Mean price in the listing of high booking rate and not high booking rate



(Haibo Yu)

As we can see from above, the price of listings with high booking rates are commonly low.

**5. Finalize the model**

**5.1 Create more features**

To further improve the accuracy of our model, we tried to add more useful text feature variables, in other words, find out features which will affect people’s decisions on whether to book the house or not.

First we counted the number of words for 8 text columns and added them as new variables. And they appeared to have high importance among all the explanatory variables. Besides, we counted the word frequencies of each text column for houses whose high\_booking\_rate equals to 1 and extracted some keywords from them. For example, “kitchen” appears a lot in the “amenities”column and “metro” appears a lot in the “Transit” column. so we wrote a function to find out whether each row of the text column contains that keyword and make dummy variables equal to 1 if it does. Adding these variables does result in a higher accuracy.

Fig 5: Word Cloud for “amenities” Column



(Xiaoyou Zhou)

**5.2 Select the best model**

We chose random forest as our final model because it gives us the highest accuracy among all the 7 models including Ridge, Lasso and Logistic Regression, Boosting, XGboost, Bagging, and Random Forest that we tried. Based on that we used grid search to tune the hyperparameters of the Random Forest model and pick up the best values of “Number of variables available for splitting at each tree node” and “The number of trees” .The final model achieved 83.5% accuracy on the test dataset which was the highest we got throughout the project.

**BUSINESS RELEVANCY:**

As we gained classification accuracy for a high booking rate, the business intelligence we could bring to Airbnb is mainly in two aspects.

1. Airbnb can use our model to predict if a listing will be popular. This prediction will help Airbnb to arrange their website list sorting.
2. Airbnb hosts can use this model to predict if their rooms could be one with high booking under current situations. So they may predict their income from Airbnb and find the relevancy of their project.
3. For investors who are interested in hotels and estate, the predictive model can help them identify houses more likely to be booked. It’s a great opportunity for them to make profit by investing in high booking hotels.