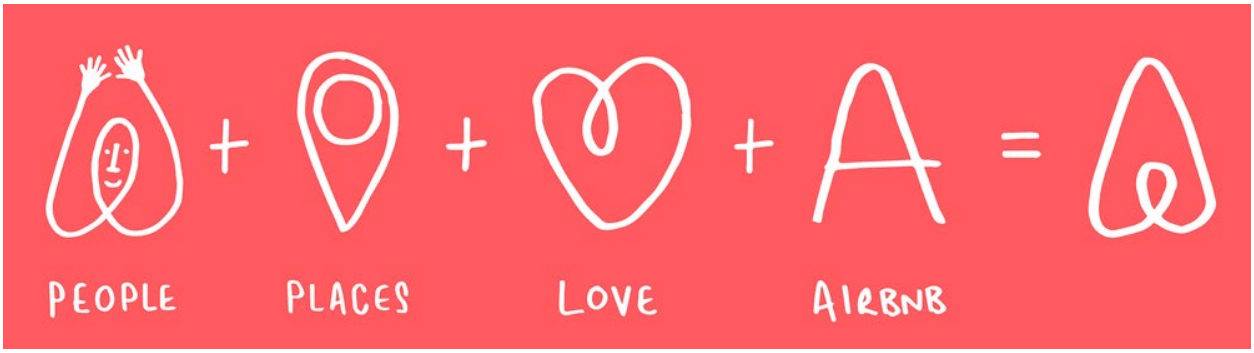
Data Mining and Predictive Analytics
BUDT 758T

Group Project - Airbnb Dataset



Group Members

Lakshita Garg
Satvik Narang
Vincent Brown
Shweta Salelkar
Jihan Shen

Table of Contents

Team member names and contributions

Vincent Brown: Cleaning original dataset, Cutoff Selection, Ridge and Lasso Models, Logistic Regression Model
Lashita Garg: Cleaning original dataset, Ranger implementation for random forest model
Satvik Narang: Cleaning original dataset, Linear regression, Variable plotting and insights
Shweta Salelkar: Cleaning original dataset, External dataset, Logistic Regression Model
Jihan Shen: Cleaning original dataset, report writing

Business Understanding

This project presents the findings of our analysis conducted on a dataset comprising various features of Airbnb listings. We built six predictive models, including linear regression, logistic regression, Random forest, Bagging, Ridge and Lasso, to identify the features most strongly related to perfect rating score (having a 100% rating). The goal of this analysis is to provide valuable insights for individuals and businesses seeking to optimize their Airbnb listings and increase booking rates. Our models offer actionable information that can be utilized to attract potential guests and maximize the revenue potential of Airbnb listings.

Target Audience and Potential Users:
The predictive models we developed cater to a wide range of individuals and businesses within the Airbnb ecosystem. The target audience includes:

1. Individual Airbnb hosts: Individual hosts can leverage the predictions to gain insights into the factors most strongly associated with perfect rating score. By identifying these features, hosts can optimize their listings, improve their descriptions, and make targeted improvements to their properties to increase bookings.

2. Property management companies: Companies specializing in managing multiple Airbnb listings can utilize the models to identify common characteristics of high-performing properties. This information can guide their acquisition strategies, property improvement investments, and marketing efforts to maximize their revenue potential.

3. Real estate investors: Investors looking to purchase properties for Airbnb rentals can benefit from the predictive models by identifying the key features associated with perfect rating score. This enables them to make informed decisions about property selection and optimization, maximizing their return on investment.

4. Airbnb platform itself: Airbnb can leverage the insights derived from our models to provide recommendations to hosts. By offering tailored suggestions for optimizing listings, Airbnb can enhance the guest experience and increase overall booking rates, resulting in increased revenue for both hosts and the platform.

Business Actions and Value Generation

The predictions generated by our models offer actionable insights that can drive strategic business actions and generate substantial value for the target audience. Here are some potential actions and value propositions based on the output of the models:

1. Listing optimization: Hosts can leverage the model predictions to identify the features that have the greatest impact on perfect rating score. By emphasizing these features in their listings, hosts can increase their attractiveness to potential guests and ultimately boost their booking rates and revenue.

2. Property improvements: Hosts and property management companies can invest in targeted property improvements based on the features identified as significant predictors of perfect rating score. For example, if amenities such as swimming pools or outdoor spaces are found to be strong predictors, hosts can consider investing in these areas to enhance their property's appeal.

3. Pricing strategy: Hosts can adjust their pricing strategies based on the model predictions. If certain features are strongly associated with perfect rating score, hosts can consider charging a premium for listings that possess those features, maximizing their revenue potential.

4. Marketing and guest targeting: Property management companies and hosts can utilize the identified features to tailor their marketing efforts and target specific guest segments. By understanding the preferences and priorities of guests associated with perfect rating score, targeted marketing campaigns can be developed to attract similar individuals or demographics.

Data Understanding and Data Preparation

Feature description:

ID	Feature Name	Brief Description	R Code Line Numbers
1	perfect_rating_score	Target Variable	23
2	availability_30	Original feature from dataset	32-39
3	availability_90	Original feature from dataset	32-39
4	bedrooms	Original feature from dataset	48-59
5	city_name	Original feature from dataset	48-59
6	price	Original feature from dataset	110-128
7	host_response_time	Original feature from dataset	110-128
8	market	Original feature from dataset	110-128
9	neighbourhood	Original feature from dataset	110-128
10	room_type	Original feature from dataset	135-136
11	host_identity_verified	Original feature from dataset	75-83
12	host_response_rate	Original feature from dataset	75-83
13	host_listings_count	Original feature from dataset	75-83
14	Transit	External dataset feature	196
15	availability_365	Original feature from dataset	32-39

16	availability_60	Original feature from dataset	32-39
17	bathrooms	Original feature from dataset	32-39
18	bed_type	Original feature from dataset	48-59
19	cancellation_policy	Original feature from dataset	48-59
20	instant_bookable	Original feature from dataset	110-128
21	is_location_exact	Original feature from dataset	110-128
22	accommodates	Original feature from dataset	104
23	extra_people	Original feature from dataset	110-128
24	property_category	Original feature from dataset	110-128
25	requires_license	Original feature from dataset	135-136
26	host_acceptance_rate	Original feature from dataset	75-83
27	host_is_superhost	Original feature from dataset	75-83
28	first_review	Original feature from dataset	72
29	host_since_days	Original feature from dataset	75-83
30	above_avg	For a given listing, if the price is greater than median price in that market	110-128
31	price_per_person	Feature created from price and accommodates column. Describes per person cost for a night	110-128
32	ppp_ind	Feature created from 'per_person_price'. 1 if value is value is median of per person price	110-128
33	has_cleaning_fee	Feature created from cleaning_fee column. 1 if the listing charges a cleaning fee	48-59

34	amenities_count	Feature created from amenities column. Gives the count of amenities provided	32-39
35	lockbox	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
36	oven	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
37	stove	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
38	microwave	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
39	bed.linens	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
40	refrigerator	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
41	indoor.fireplace	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
42	pets.allowed	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
43	buzzer.wireless.intercom	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
44	self.check.in	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
45	lock.on.bedroom.door	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
46	translation.missing..en.hosting_amenity_50	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
47	wifi	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
48	free.parking.on.premises	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
49	family.kid.friendly	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165

50	internet	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
51	laptop.friendly.workspace	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
52	hair.dryer	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
53	carbon.monoxide.detector	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
54	shampoo	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
55	air.conditioning	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
56	coffee.maker	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
57	cooking.basics	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
58	elevator	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
59	breakfast	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
60	dishes.and.silverware	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
61	private.entrance	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
62	hot.water	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
63	pets.live.on.this.property	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
64	safety.card	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
65	translation.missing..en.hosting_amenity_49	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165

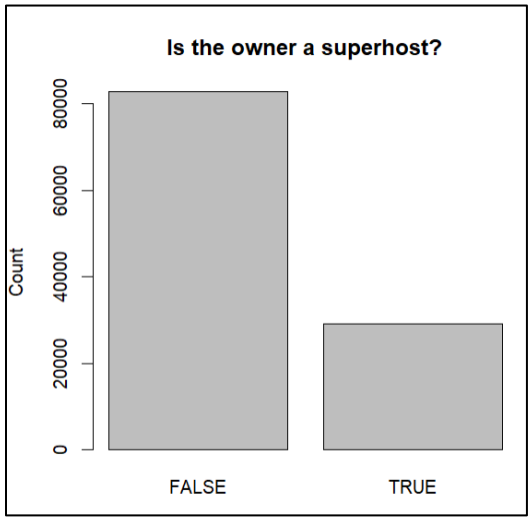
66	X24.hour.check.in	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
67	cable.tv	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
68	first.aid.kit	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
69	fire.extinguisher	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
70	wireless.internet	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
71	iron	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
72	dryer	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
73	washer	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
74	hangers	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
75	tv	Feature created from amenities column by text mining. Value is 1 if this amenity is available	143-165
76	OtherTransp	External dataset feature	198
77	Drive	External dataset feature	195
78	Walk	External dataset feature	197
79	Income	External dataset feature	199-200

We also attempted to clean the description feature but it did not help us in our analysis

Interesting Insights:

List of features:

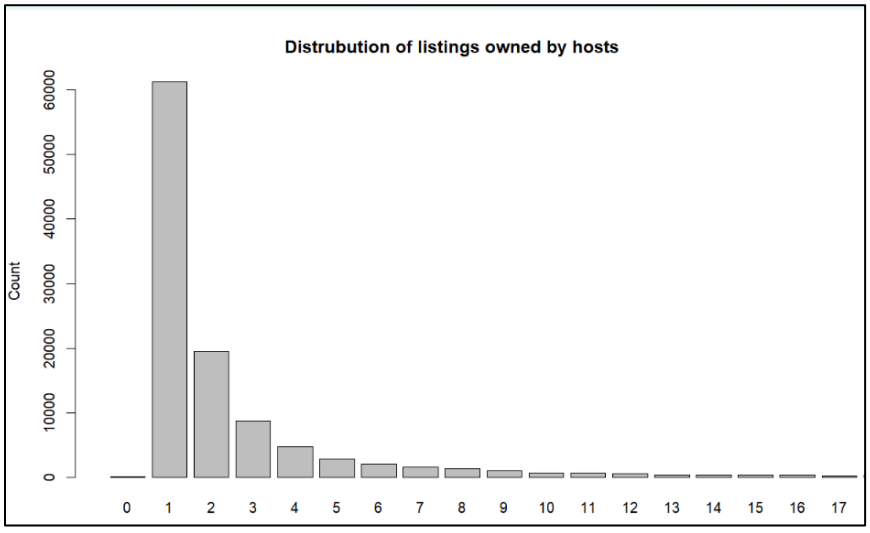
1. **host_is_superhost**




```
FALSE  TRUE
82818 29216
```

This feature describes if the host is a superhost (an experienced and highly rated host who consistently provides exceptional hospitality to guests on the Airbnb platform). Only 25% of the total listings have a superhost.

2. **host_listings_count**



Most people own property which they have listed on Airbnb. Fewer people own more than a single listing

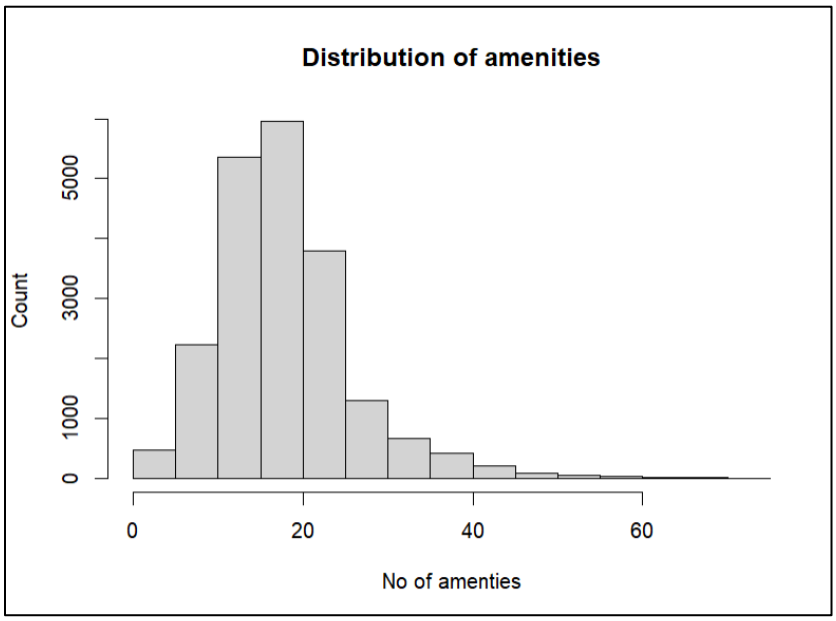
3. **city_name:**

```
> table(total$city_name)

    Asheville        Austin        Boston       Chicago       Denver   Los Angeles
          742          5942          3944          4469          3393         23718
    Nashville   New Orleans      New York       Oakland      Portland     San Diego
         4557          4694         36941          1300          4187          4481
San Francisco    Santa Cruz       Seattle Washington DC
         4367           699          3171          5581
```

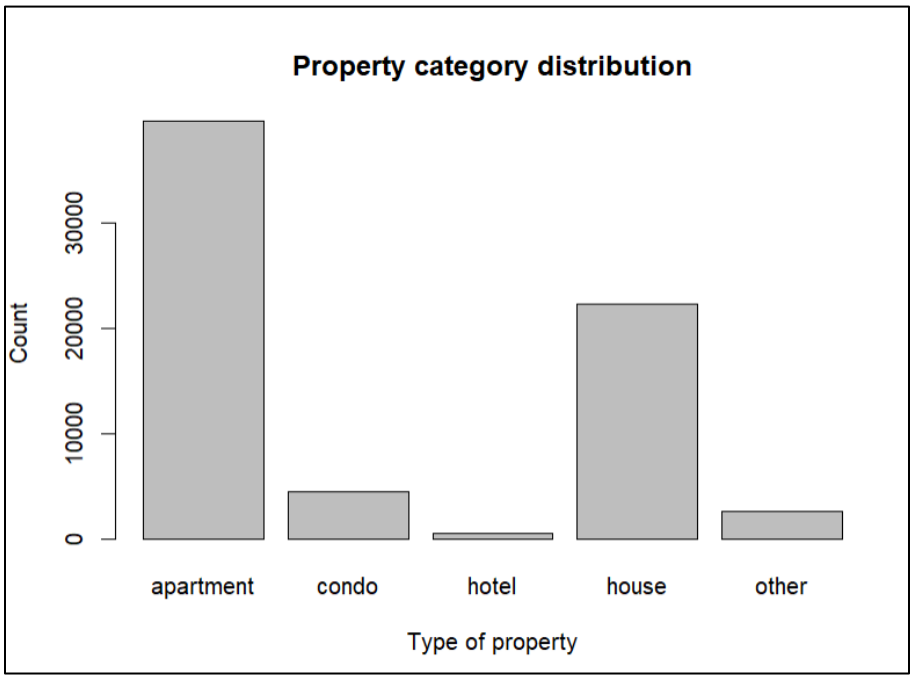
Given airbnb listings are spread across 16 geographical areas. New York has the highest number of listings, followed by LA and Austin. Demand in these areas would be high due to the presence of a lot of young population.

4. **amenities_count (feature created from 'amenities')**



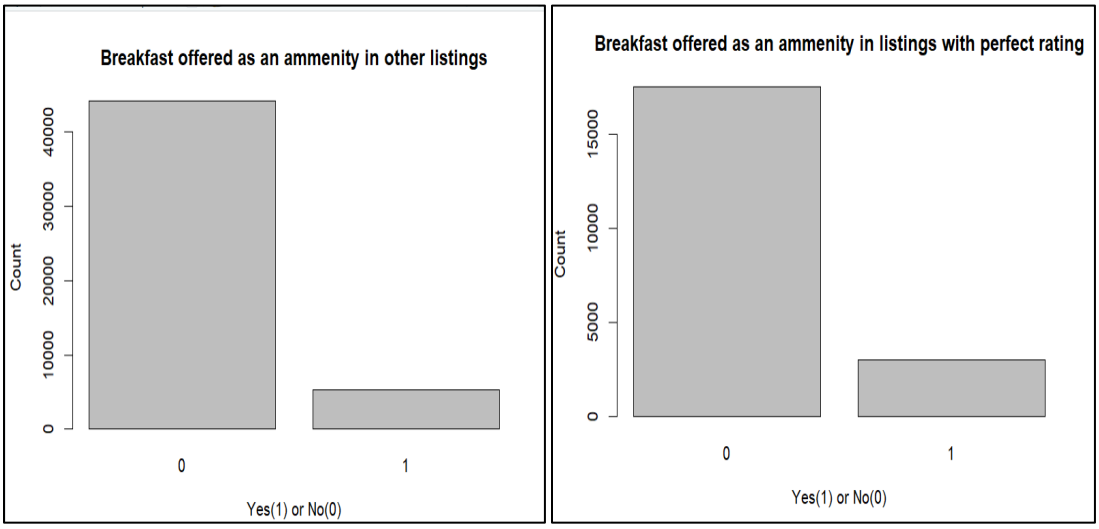
amenities_count provides the total number of amenities provided by an Airbnb property. The distribution is skewed to the right showing that there are a few listings that provide a large number of amenities. A majority of properties provide around 20 amenities.

5. **property_category**



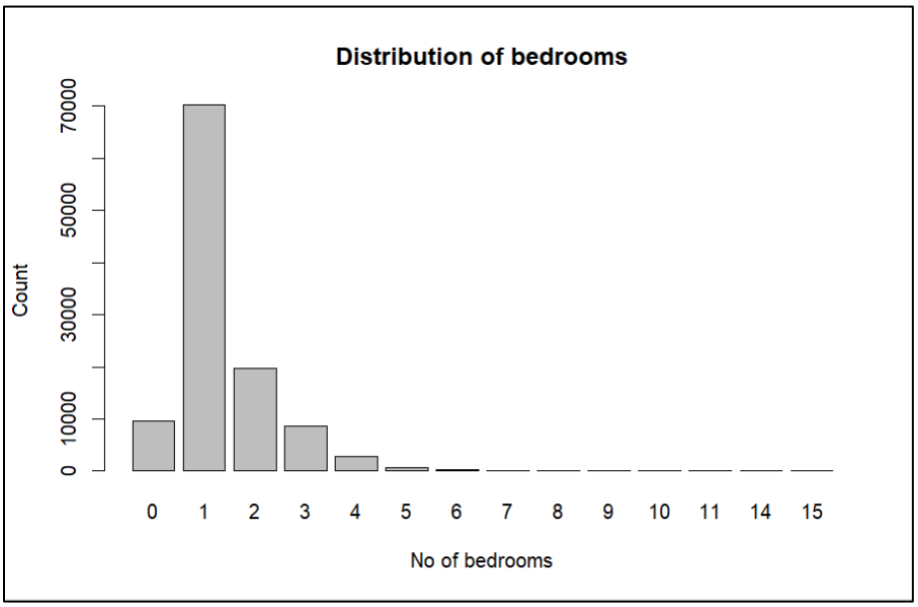
Out of all the available listings on Airbnb, apartments and houses are the most popular property categories, followed by condos. Hotels are least preferred and have been listed the least.

6. **breakfast - as an amenity**



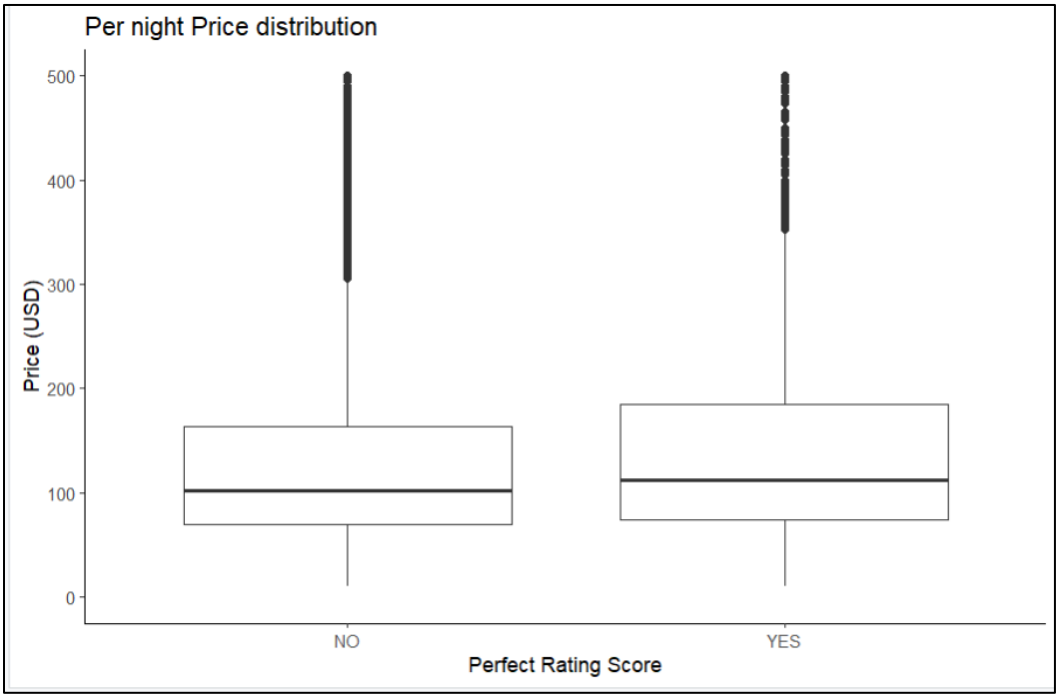
Breakfast being provided as an amenity acts as a factor in achieving a perfect rating score. From the second chart, a higher percentage of listings have a perfect rating score serving breakfast (1) as an amenity.

7. bedrooms



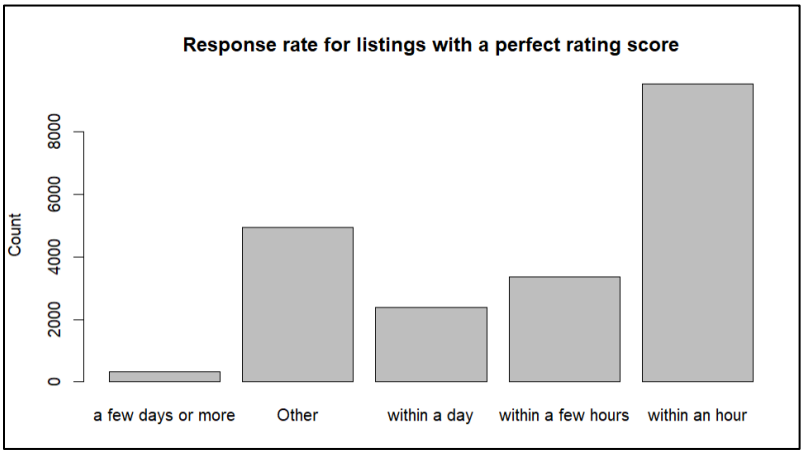
Properties with a single bedroom have been listed the most on Airbnb based on the given dataset, followed by 2 bedrooms and 0 bedrooms (studios).

8. Price



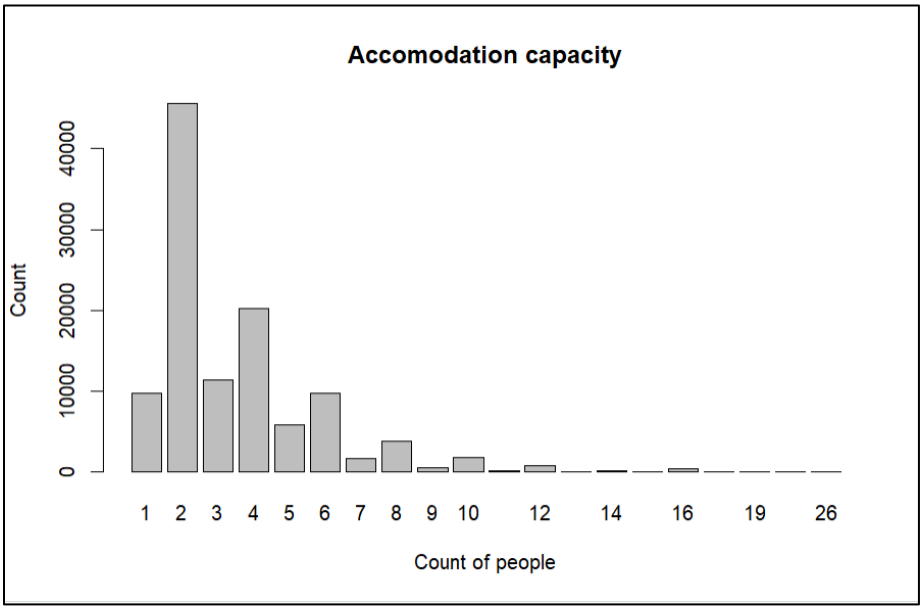
Price (per night for the listing) distribution across the 2 categories shows median prices are approximately similar around 100$. Listings with a perfect rating score have a marginally higher median price.

9. host_response_rate



Response provided by the host is captured by host_reponse_rate varies from within an hour to few or more days. To be a good host and achieve a perfect rating score, it is important to have a quick response rate. For listings with a perfect rating score, response is majorly within an hour.

10. accommodates



Majority of the listings have a capacity for 2 or 4 people. There are relatively fewer options available which have a high capacity to host more than 4 people.

External Dataset:

To increase the performance of our model, we added data from the Census dataset (https://data.census.gov/):

- The DP03 table, also known as the "Selected Economic Characteristics," contains data related to income, employment, commuting patterns, occupation, and other economic indicators. It provides insights into the economic well-being and labor market of specific areas. From this Table, we added the median **household median income** of a zipcode and **mode of transportation** that people for commute to our current dataset (this would give an idea of transportation availability in the zipcode).

The data from this table were joined at a zip code level to our original dataset and all 5 columns used were cleaned by replacing the null values with the mean.

Metric	Prior to using the external dataset	After using the external dataset
TPR	0.4187	0.4212
FPR	0.0911	0.0917
Accuracy	0.7638	0.7638

We notice that while the accuracy remained similar, our TPR did increase and this dataset would be a valuable addition to our model.

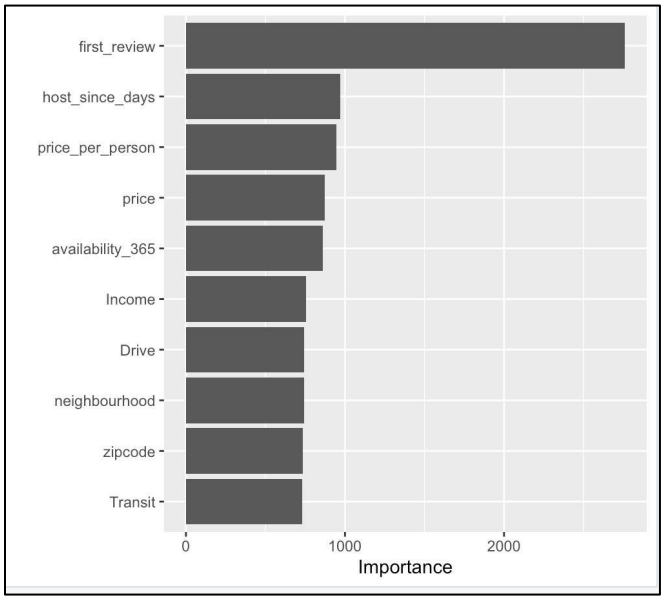
Evaluation and Modeling

The winning model - Ranger:

The logistic regression function **ranger()** is taken from the **ranger and vip library**. The winning model in our analysis was the Ranger model, which is an optimized implementation of random forest in R. This model considered a wide range of variables from the Airbnb dataset, including listing attributes, amenities, location factors, and pricing information. To start with, we split our training data in the 70:30 train to test ratio and then worked on training the model and building a confusion matrix on the testing data. We filtered out our metrics results for an FPR lesser than 0.092. On applying this filter we got the following metrics in the training dataset -

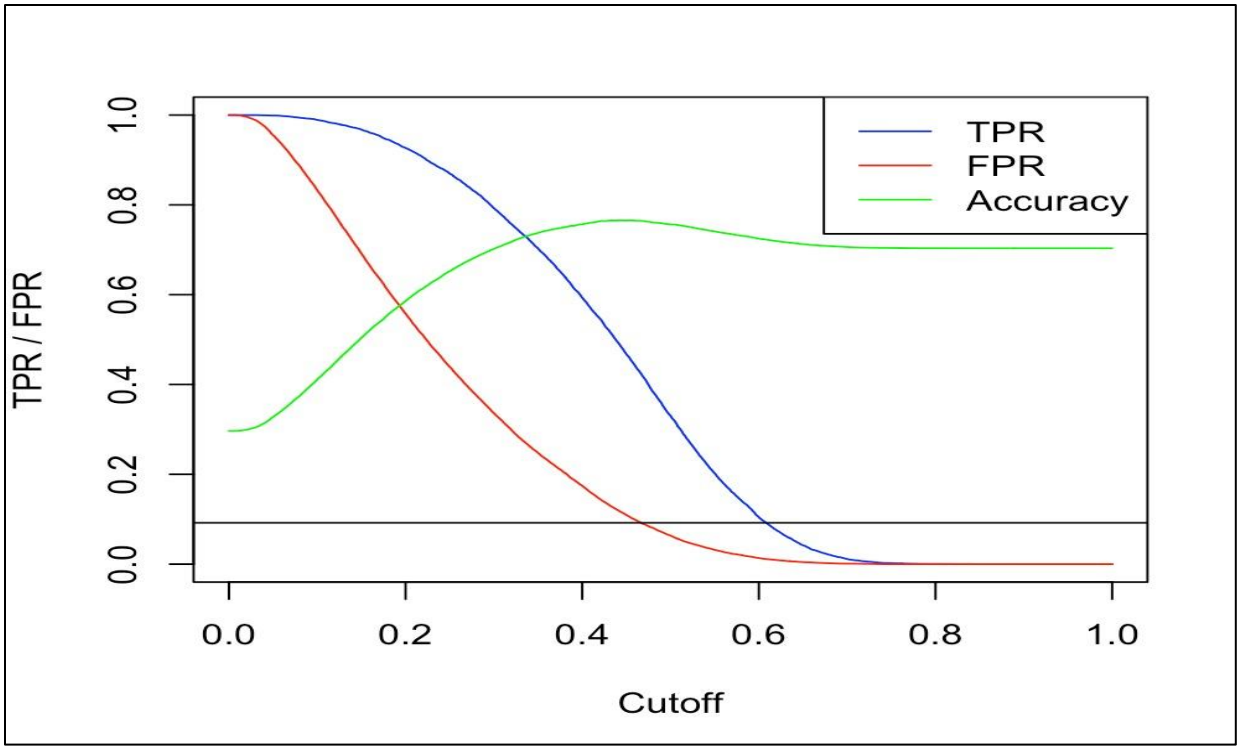
Metric	Training data	Validation data without amenities and external dataset	Validation data with amenities and external dataset
TPR	1	0.4187	0.4212
FPR	0.0917	0.0911	0.0917
Accuracy	0.93	0.7638	0.7638

We noted the best performing features by plotting a vip chart:



Here we see that features Income, Drive and Transit are from our external dataset and this shows us that the external dataset is indeed improving our model. The **predictions** were generated using this model (**Refer line 1119**) and **generalization performance** is calculated on line no **1015-1017**

We then plotted a fitting curve:



The ntrees hyperparameter was tuned by keeping the range from 100 to 1000 with a step of 100. To determine the Ranger model as the winning model, we compared its performance with other models using various evaluation metrics such as TPR, FPR and accuracy. After comprehensive analysis and comparison, the Ranger model consistently demonstrated superior predictive performance and robustness. In our R code, the final predictions generated by the Ranger model can be found on line 931 of the script.

Other Models attempted:

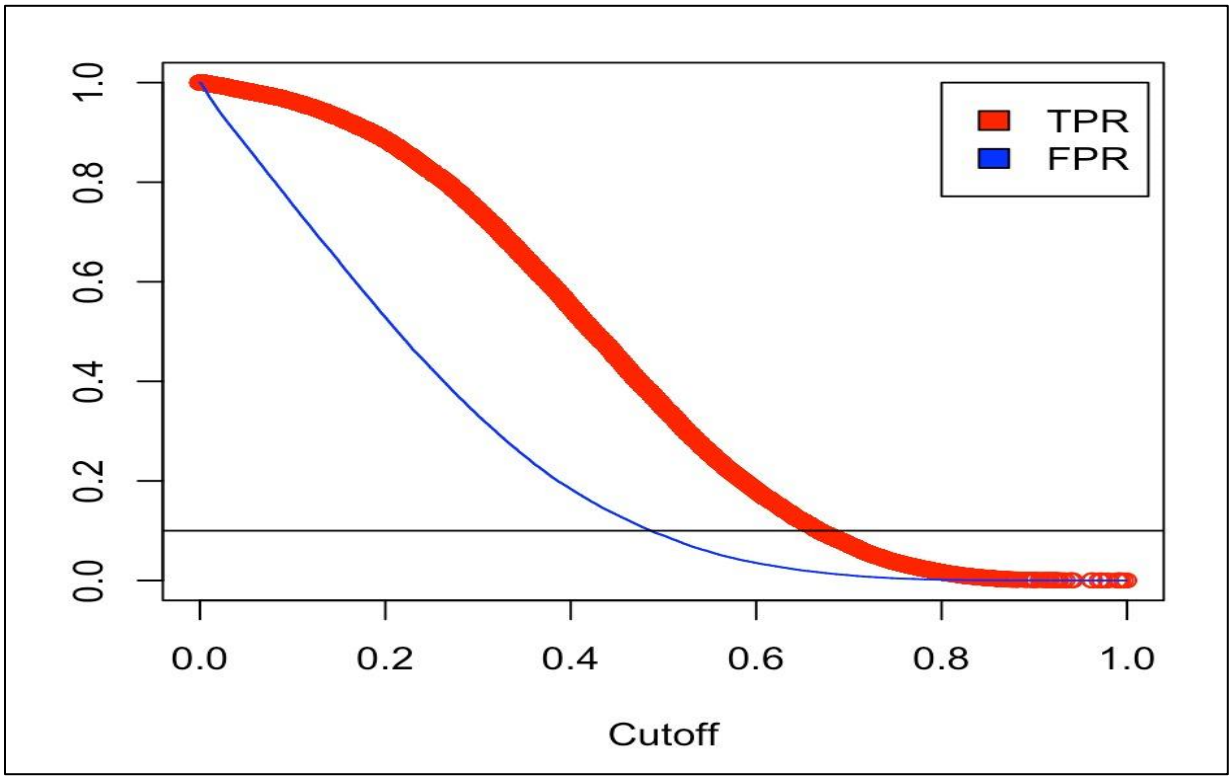
1. **Logistic regression:**

The logistic regression function **glm()** is taken from the **caret and boot package**. A simple train/validation split in a 60:40 ratio was used to estimate our generalization performance, however we then used 5 fold cross validation. The performance was as follows:

Metric	Training data	Validation data
TPR	0.355	0.36
FPR	0.092	0.0919
Accuracy	0.74	0.74

The cutoff hyperparameter was tuned by filtering the FPR to less than 0.092 and the cutoff was found to be 0.4975. The best performing set of features were:
"city_name: Santa Cruz"
"market: Monterey Region"
"market: South Bay, CA"
"cancellation_policy: no_refunds"
"city_name: Nashville"
"market: Nashville"
"city_name: Boston"
"host_response_time: within an hour"
"city_name: New Orleans"
"city_name: San Diego"
"host_response_time: within a few hours"
"city_name: Washington DC"
"market: Portland"

In our code, logistic regression **modeling** can be found on **line 512**, **generalization performance** is calculated on line no **600-602** and a visualization of the performance of both models can be found in the charts below:



The accuracies from the 5 fold cross validation was:

Fold 1	0.747
Fold 2	0.749
Fold 3	0.7366
Fold 4	0.7399
Fold 5	0.7431
Average Accuracy	0.7431

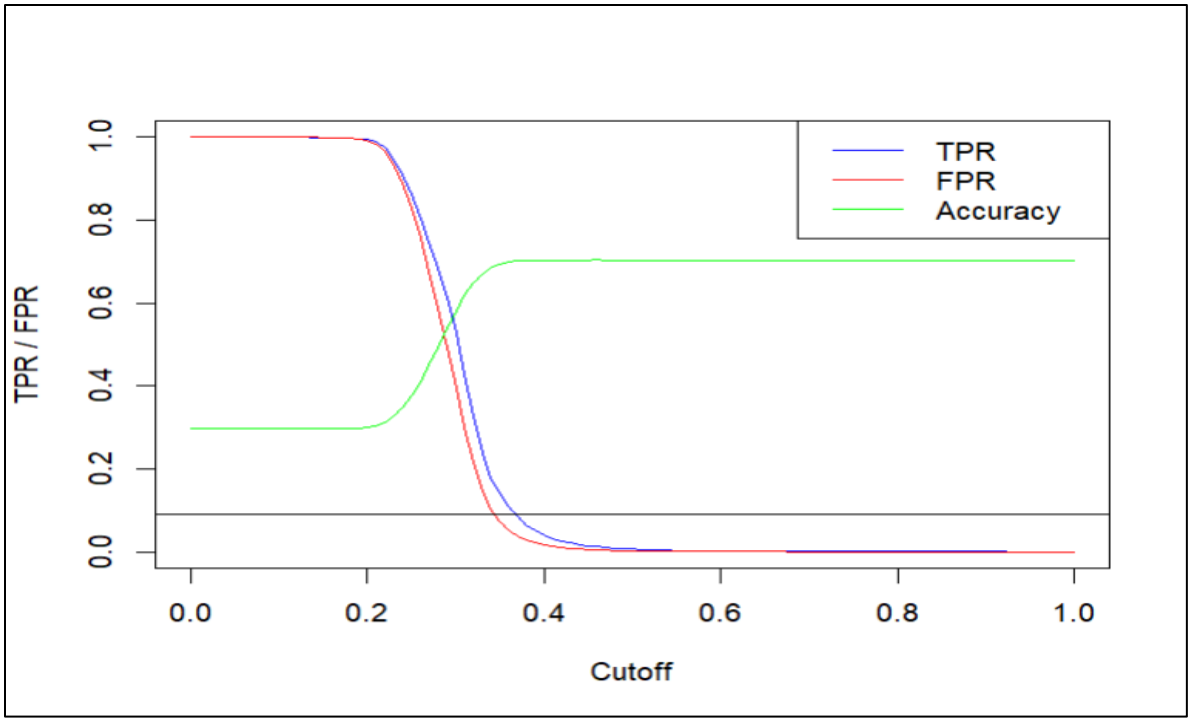
2. Linear Regression

The function lm() is used for linear regression and is taken from the tidyverse library. A simple train/validation split in a 70:30 ratio was used to estimate our generalization performance. The performance was as follows:

Metric	Training data	Validation data
TPR	0.070	0.133
FPR	0.021	0.073
Accuracy	0.7035	0.6932

The cutoff hyperparameter was tuned by filtering the FPR to less than 0.092 and cutoff was found to be 0.35. Linear regression is used to predict a continuous numerical target variable and hence gives below average performance when used for classification of a binary variable. The best performing set of features was as follows:
"Availability_30"
"Bathrooms"
"Amenities_count"
"Bedrooms"
"Price"
"Price_per_person"
"Host_response_rate"

In our code, linear regression **modeling** can be found on line **403**, **generalization performance** is calculated on line no **476-478** and a visualization of the performance of both models can be found in the charts below:



3. Ridge Regression

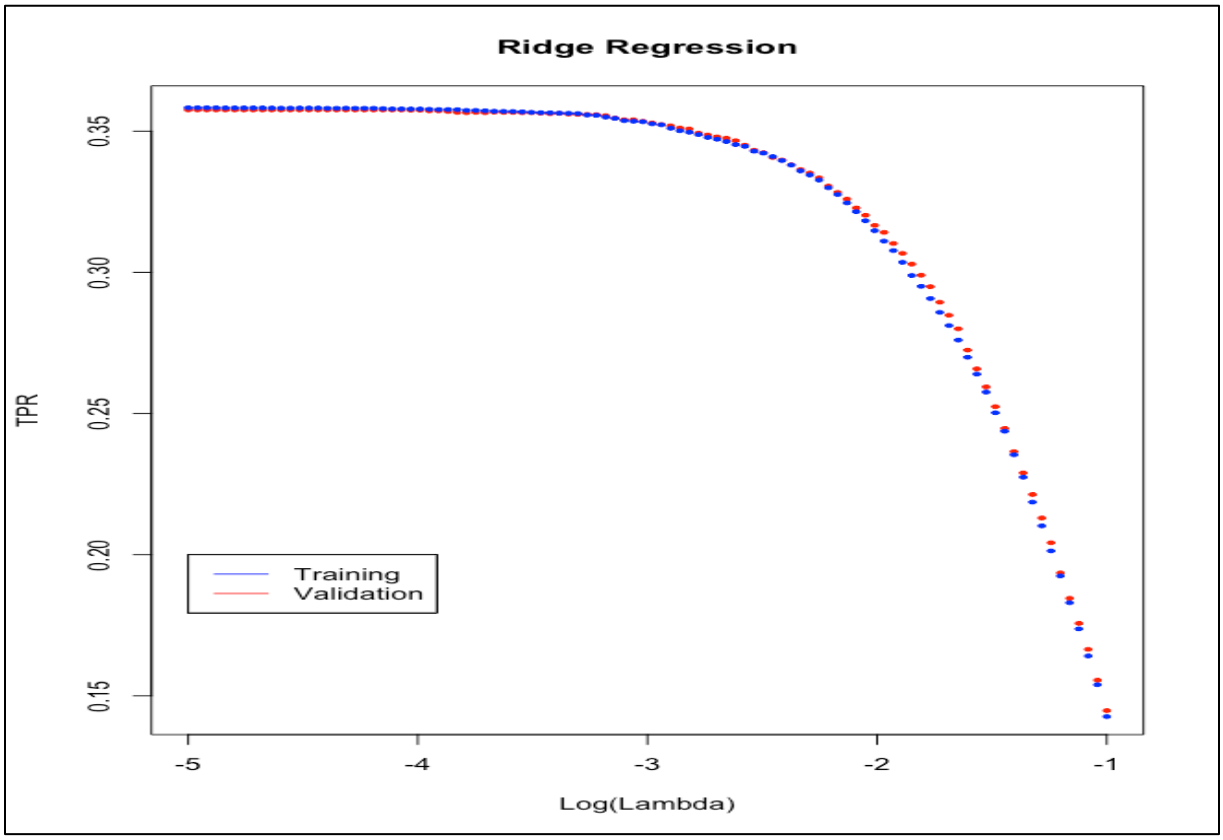
The Ridge regression function **glmnet()** is taken from the **tidyverse, caret and glmnet** package. A simple train/validation split in a 60:40 ratio was used to estimate our generalization performance. The performance was as follows:

Metric	Training data	Validation data
TPR	0.232	0.12
FPR	0.04	0.051
Accuracy	0.732	0.7121

Tuning lambda within R, we used grid search to sample hundreds of possible values and evaluated the model's predictive performance on the validation subset of our training data. Doing so, we found that the optimal lambda for ridge is 0; therefore, the ordinary least squares model would be a preferred model due to decreased complexity. The best performing set of features were:
"City name"
"Market"
"Bedrooms"
"Price"
"Price_per_person"

"Host_response_rate"

In our code, ridge regression **modeling** can be found on line **801**, **generalization performance** is calculated on line no **840-842** and a visualization of the performance of the model can be found in the chart below.



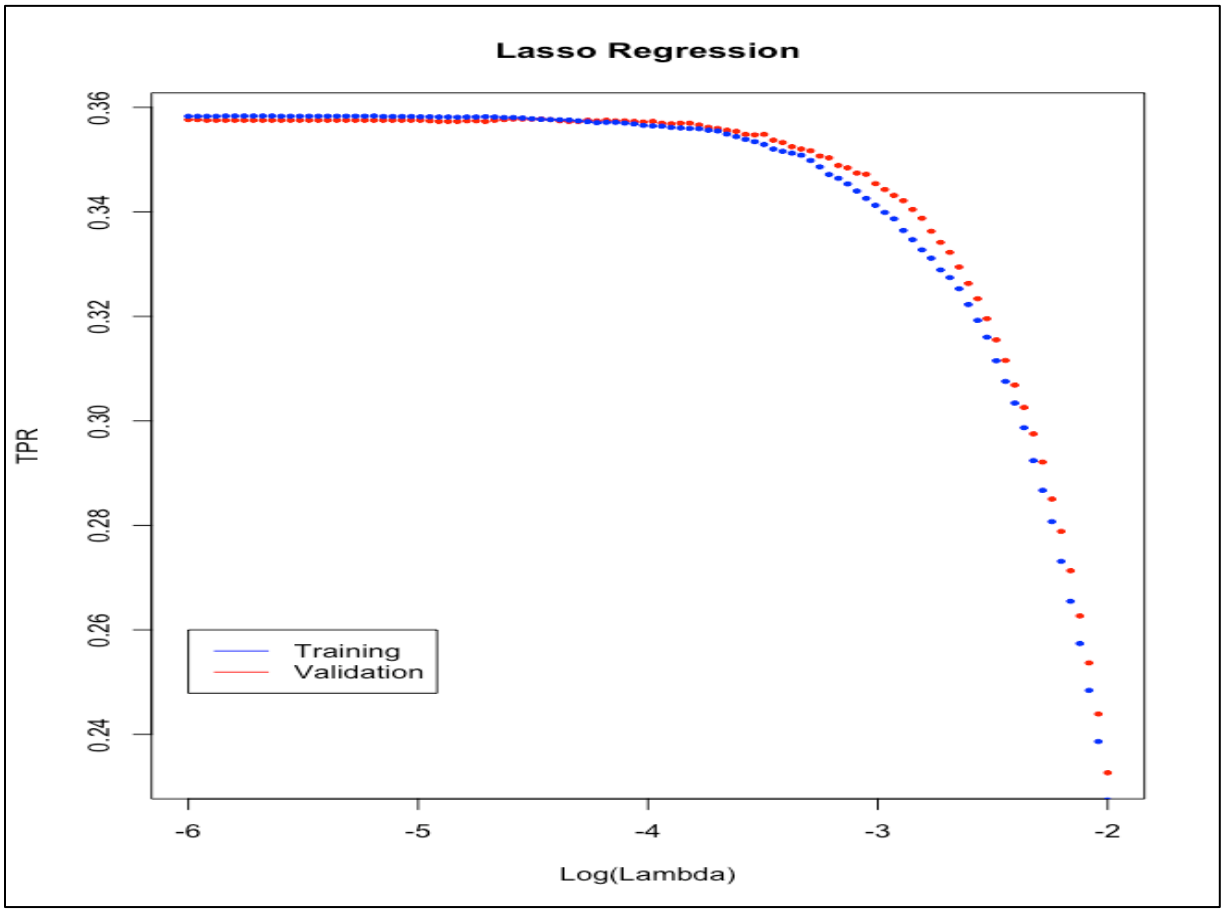
4. Lasso Regression

The Lasso regression function **glmnet()** is taken from the **tidyverse, caret and glmnet package**. A simple train/validation split in a 60:40 ratio was used to estimate our generalization performance. The performance was as follows:

Metric	Training data	Validation data
TPR	0.4232	0.3012
FPR	0.059	0.062
Accuracy	0.797	0.701

Tuning lambda within R, we used grid search to sample hundreds of possible values and evaluated the model's predictive performance on the validation subset of our training data. Doing so, we found that the optimal lambda for lasso is 0; therefore, the ordinary least squares model would be a preferred model due to decreased complexity. The best performing set of features were:
"Cancellation_policy"
"Market"
"Availability_30"
"Bathrooms"
"Price_per_person"
"Host_response_rate"

In our code, lasso regression **modeling** can be found on **line 856**, **generalization performance** is calculated on line no **895-897** and a visualization of the performance of the model can be found in the chart below.

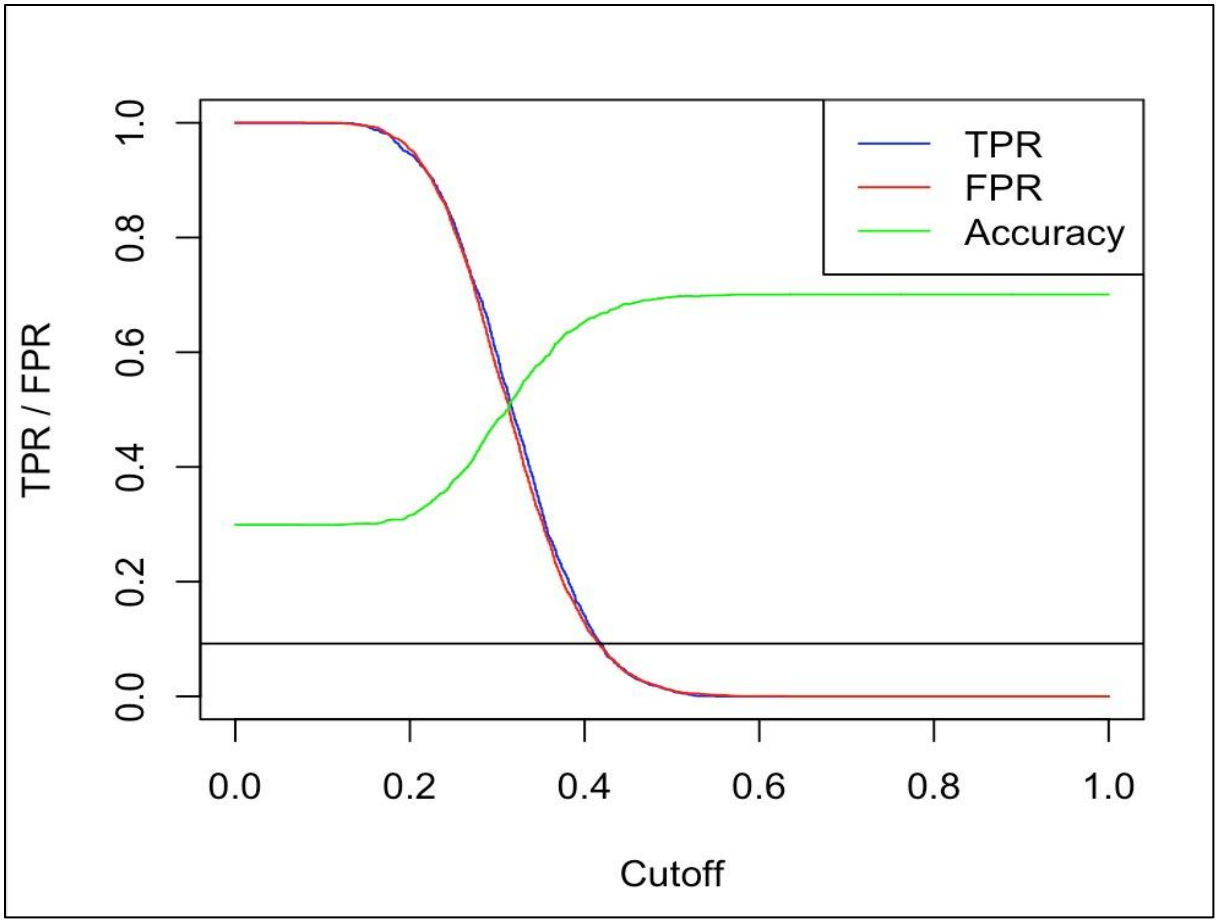


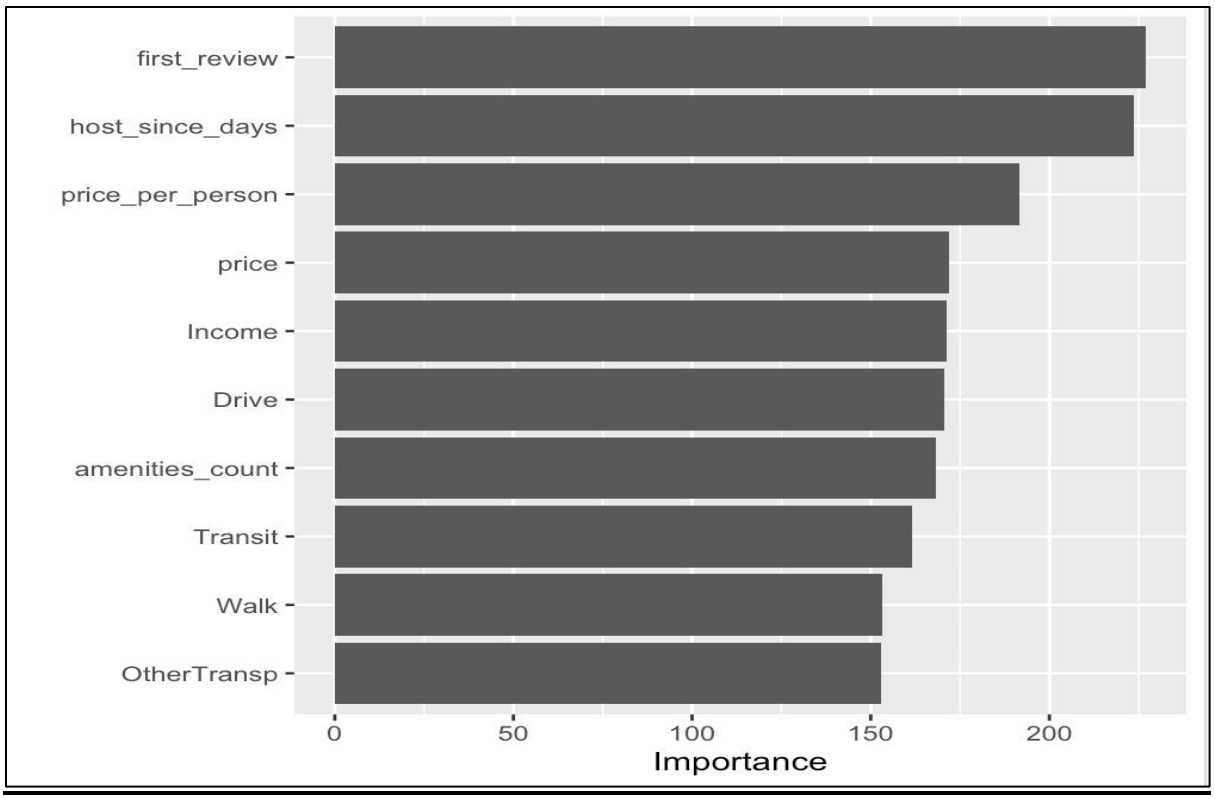
5. Bagging

The Bagging regression function bagging() is taken from the random forest package. A simple train/validation split in a 60:40 ratio was used to estimate our generalization performance. The performance was as follows:

Metric	Training data	Validation data
TPR	1	0.0954
FPR	0.090	0.9090
Accuracy	0.936	0.6655

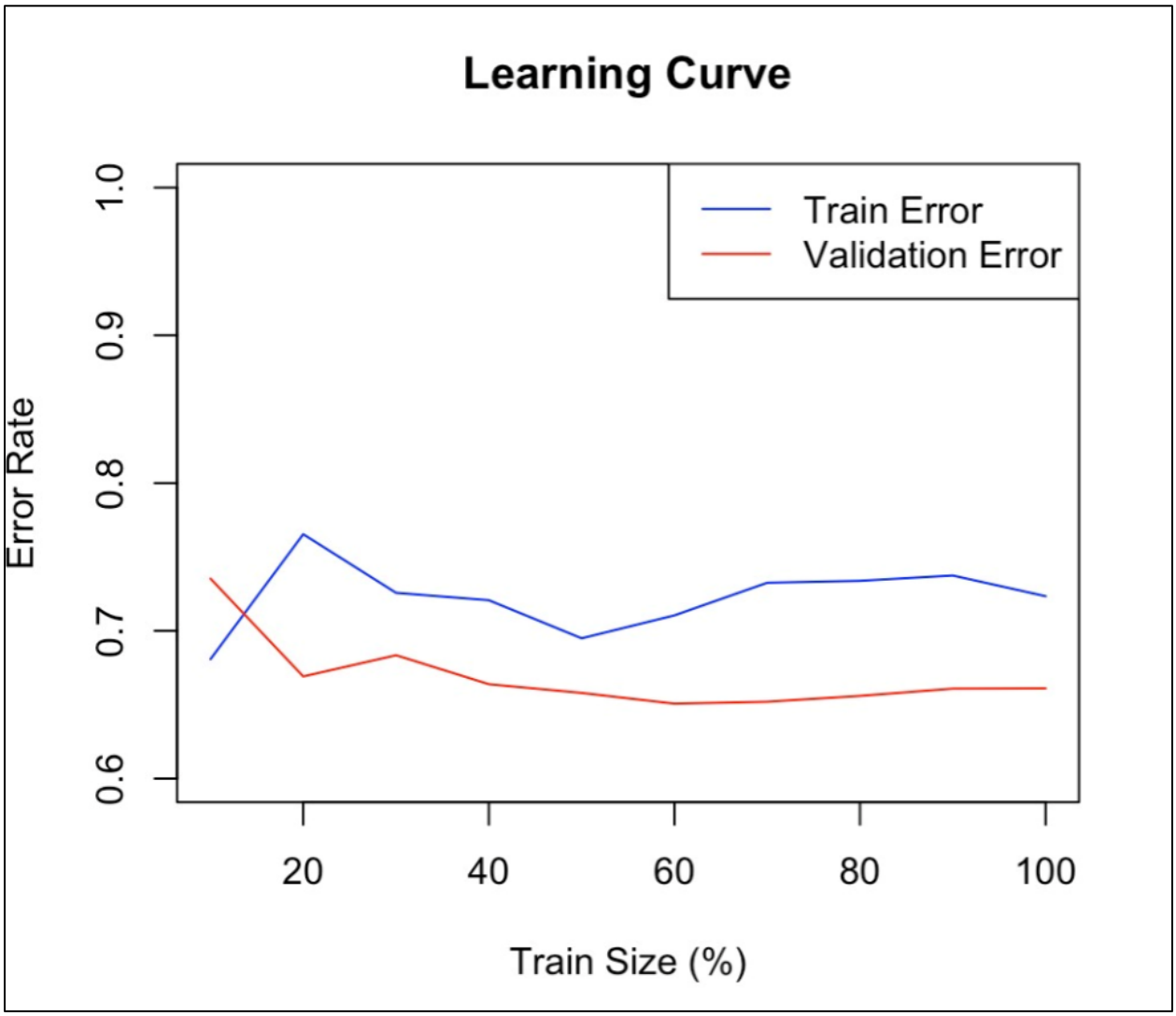
Bagging was performed on a smaller subset of data, which led to imbalance between positive and negative classifications sampled. The cutoff hyperparameter was tuned by filtering the FPR to less than 0.092 and cutoff was found to be 0.32. In our code, bagging regression **modeling** can be found on **line 678**, **generalization performance** is calculated on line no **758-760** and a visualization of the performance of the model can be found in the chart below.


TPR
FPR
Accuracy
TPR / FPR
0.0
0.2
0.4
0.6
0.8
1.0
Cutoff


first_review
host_since_days
price_per_person
price
Income
Drive
amenities_count
Transit
Walk
OtherTransp
0
50
100
150
200
Importance

Learning curve:

The following learning curve was created for random forest model implemented using ranger. The training size based on the below learning curve was set to be 70% as both the training and validation error starts to increase after that point.



Reflections on the Data Mining Project:

Our group undertook a comprehensive analysis of a dataset containing various features of Airbnb listings with the aim of identifying the factors that contribute most significantly to perfect rating score. Throughout the project, we built and evaluated six predictive models, including linear regression, logistic regression, Random forest, Bagging, Ridge, and Lasso. The project allowed us to gain valuable insights into the optimization of Airbnb listings and the enhancement of perfect rating score. In this section, we will reflect on our group's performance, highlight the main challenges we encountered, discuss what we would have done differently given the opportunity to start the project over again, and explore what we would have accomplished with additional time. Finally, we will provide advice for future groups undertaking a similar project.

What we did well:

One aspect in which our group excelled was the thoroughness of our analysis. We carefully selected a diverse range of models, ensuring that we covered various algorithmic approaches suitable for the dataset and problem at hand. This allowed us to explore different perspectives and gain a deeper understanding of the data. Additionally, our group demonstrated effective teamwork and communication. We divided tasks efficiently, collaborated closely, and maintained clear channels of communication throughout the project. This enabled us to make timely progress, resolve issues promptly, and produce a cohesive final report.

Our main challenges:

One of the primary challenges we encountered was data preprocessing and feature engineering. The Airbnb dataset was extensive and contained missing values, outliers, and categorical variables requiring transformation. It took significant effort and time to clean the data, handle missing values appropriately, and convert categorical variables into suitable representations for our models. Another challenge we faced was feature selection. With a large number of potential predictors, identifying the most relevant features for predicting booking rates was a complex task. We employed techniques like feature importance from ensemble models, and domain knowledge to guide our selection process, but it required careful consideration and experimentation.

What we could have done differently if we could start again:

If we could start the project over again, we would have focused more on exploratory data analysis (EDA) at the beginning. While we conducted some initial EDA, we realized later in the project that a more in-depth understanding of the dataset would have informed our feature engineering and selection processes. Additionally, we would have allocated more time to fine-tune hyperparameters of our models. Although we achieved reasonable results, optimizing model parameters can often yield better performance and generalization. Furthermore, we would have implemented cross-validation more rigorously during model evaluation to obtain more reliable estimates of model performance.

What we could have done differently if we had more time:

If we had a few more months, we would have explored additional modeling techniques beyond the ones we used. For example, we could have considered support vector machines (SVM), neural networks, or other ensemble methods. This would have allowed us to compare the performance of these models with the ones we implemented and potentially discover additional insights. Moreover, we would have expanded our dataset by incorporating external data sources, such as local events or tourist attractions, to capture more contextual information that could influence booking rates. Finally, with extra time, we would have conducted a more extensive analysis of the pricing aspect, investigating the relationship between pricing strategies and booking rates to provide further recommendations for listing optimization.

Advice for the next batch of students:

For future groups starting this project, we offer the following advice:

- During the project, our group considered the potential benefits of incorporating external datasets to enhance the predictive power of our models. We recognized that additional contextual information could provide valuable insights into the factors influencing Airbnb booking rates. If you were to undertake this project next year, we recommend exploring relevant external datasets, such as local tourism data, weather data, or neighborhood demographics. By integrating these datasets with the existing Airbnb dataset, you can enrich your analysis and potentially uncover new patterns and relationships. However, it is important to ensure that the external dataset aligns with the scope and quality of the Airbnb dataset and that appropriate data integration techniques are employed.
- Take ample time for data preprocessing and feature engineering. Cleaning the data, handling missing values

References:

https://data.census.gov/