
# SUMMATIVE PROJECT REPORT

# AIRBNB REVIEWS - PREDICTION MODELS

## EXAM CANDIDATE NUMBERS -

**39719**
**36493**
**27373**

Course: MA429 Algorithmic Techniques for Data Mining

April 29, 2022

## PART 1 - EXECUTIVE SUMMARY

Data Mining has found its use case in a wide variety of industries. One such field is the travel and tourism industry, where different type of data is gathered for each traveller, whether from booking a cab or from booking hotel accommodation. Our research in this report analyzes trends on data from the popular homestay and lodging website Airbnb for listed properties in London.

Exploratory data analysis of the raw data reveals some interesting insights into the London lodging market. Most properties listed in London are from Central London, particularly from regions like Westminster, Camden and Tower Hamlets. An analysis of the price per property reveals that Sutton is the cheapest and Kensington is the most expensive place to rent an Airbnb.

We implement traditional machine learning techniques to predict a customer's review scores for a property based on various attributes related to customer experience, price, amenities and property owner. The bagging and boosting model worked well for most predictions but other methods aren't far along. We conclude that our models have scope for further improvement if given access to more computational resources and are prone to the curse of dimensionality for some models.

## PART 2 - EXPLORATORY DATA ANALYSIS

### 2.1:- Data Source and Description

Our dataset is sourced from Inside Airbnb [1], an open-source project that provides data and advocacy about Airbnb's impact on residential communities. This data is originally scraped from publicly available information from the Airbnb site. The last scaping for the data of London was done on December 7, 2021, and we use the listings data for our analysis.

The raw data contains over 66000 instances for 74 attributes, ranging from host details to listing specifics. Each instance of data is a listed property on Airbnb and contains all the information mentioned on the property webpage. There are 7 columns in the data that store the reviews of past customers on factors like - overall rating, communication, location, check-in experience, value, accuracy and cleanliness.

A brief description of the attributes can be found in the table below:-

| Attribute Name | Attribute Description |
|---|---|
| host_response_time | Nominal; host_response_time type:<br>0 – NA; 1 – within an hour; 2 – within a few hours; 3 – within a day; 4 – a few days or more |
| host_response/acceptance_rate | Numerical; range 0-1; NA is replaced by the median on the training set. |
| host _is_superhost | Nominal; 1– superhost, 0 –, not superhost |
| host_total_listings_count | Numerical; the number of listings each host owned |
| host_has_profile_pic | Nominal; 1– yes, 0 – no |
| host_identity_verified | Nominal; 1– yes, 0 – no |
| neighbourhood_cleansed | Nominal; the neighbourhood of listings: westminster, camden…. |
| latitude, longitude | Numerical; |
| Property_type, room_type | Nominal; property/room type of listings |
| accommodates, bathrooms, beds | Numerical; count the number of people/ number of bathrooms/number of beds in one listings |
| price | Numerical; NA is replaced by the mean price according to room type on training set |
| minimum_nights/maximum_nights | Numerical; unit in days; |
| Has availability | Nominal; 1– yes, 0 – no; availability of listings |

| | |
|---|---|
| availability_30/availability_60/availability_90/availability_365 | Numerical; number of availability days in period of 30/60/90/365 days |
| Number_of_reviews | Numerical; total number of reviews of host |
| Number_of_reviews_ltm/Number_of_reviews_l30d | Numerical; total number of reviews last 12 months/ last 30 days |
| Review_scores_rating/accuracy/cleanliness/checkin/communication/location/value | Numerical; ranged 0-5; Rating given by customers for each host from total experience,accuracy, cleanliness, checkin, communication, location, value; predicted |
| Instant bookable | Nominal; 1– yes, 0 – no |
| Calculated_host_listings_count | Numerical; |
| Calculated_host_listings_count_entire_homes/private_rooms/shared_rooms | Numerical; |
| Review per month | Numerical; |
| Join time | Numerical; unit in days;NA is replaced by the mean on the training set. |
| count_verification | Numerical; the number of distinct verification ways |
| count_amenities | Numerical; the number of distinct amenities host provided |
| 90 Dummy variables created from amenities | Nominal; 1– host provide, 0 – doesn't provide |

**Table 1: Variable Names and Description**

### 2.2:- Missing Values in Data

4 columns in the dataset ("neighbourhood_group_cleansed", "calendar_updated", "license", “bathrooms”) were entirely blank. 10 columns had 20-30% of the instances missing and the remaining had less than 10% missing values. The graph below shows a frequency distribution of missing values per column in our dataset -

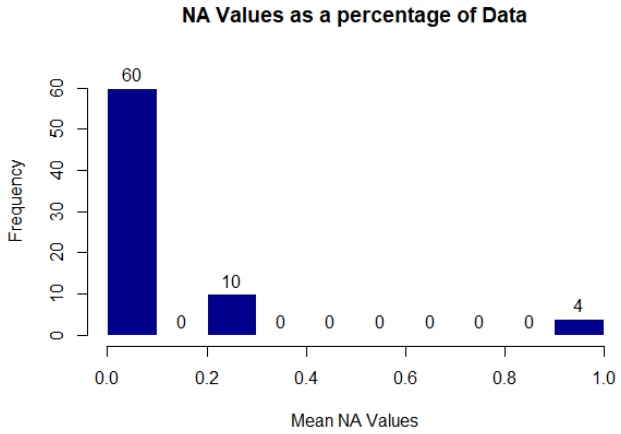



**Figure 1: Frequency distribution of NA values in columns**

After preliminary cleaning (discussed in Section 4.3.1), we were left with 45 variables. The missing values in the modified data have the following frequency distribution -

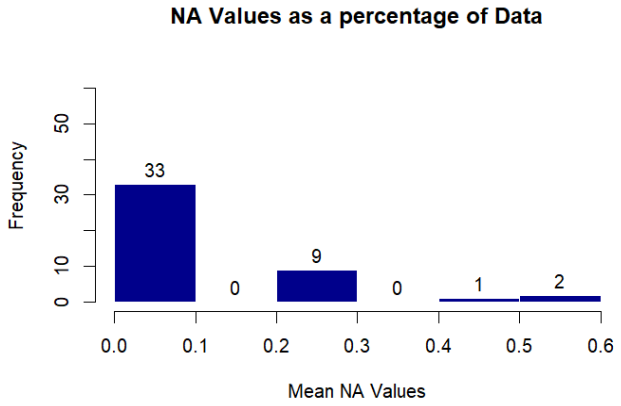



**Figure 2: Updated frequency distribution after data cleaning**

### 2.3:- Analysis of Neighbourhoods Listed

London is divided into several zones from Zone 1 (city centre) to Zone 6 (outskirts of the city). Since most travellers in the city are either tourists or business/government workers, we expect regions in central London to be the most popular areas to rent out a property.

The data testifies to this hypothesis as out of the 33 neighbourhoods, Westminster is the most popular region, which is followed by Tower Hamlets, Hackney, Camden, Kensington and Chelsea, Southwark, Islington, Lambeth, Wandsworth and Hammersmith & Fulham. The following graph helps us put this data aesthetically -

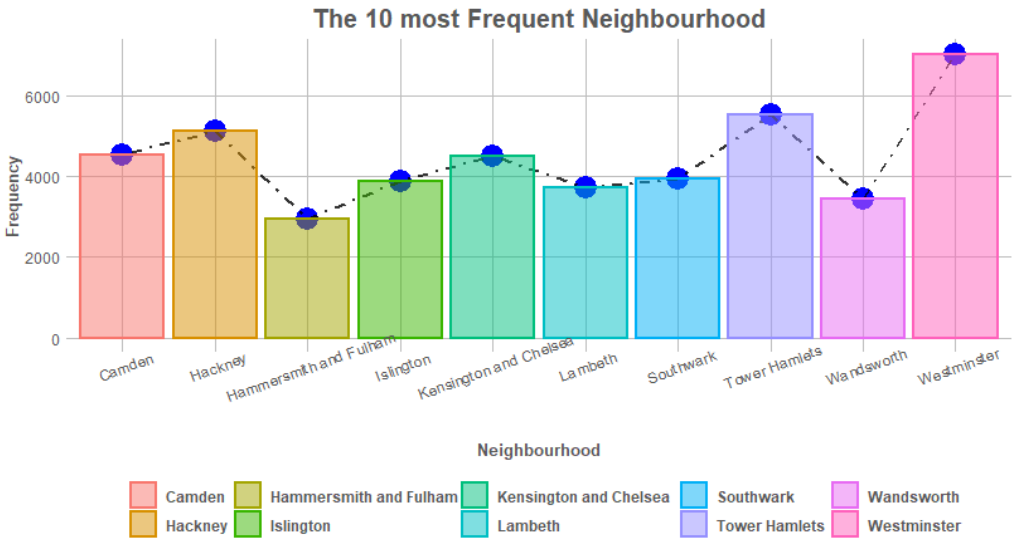



**Figure 3: Frequency distribution of the most common neighbourhood**

Trends in the price of the listed properties were also noted, with central regions having the highest fares and the regions on the outskirts having relatively cheaper options. The following polar charts highlight the neighbourhoods with the cheapest and most expensive properties listed on Airbnb -

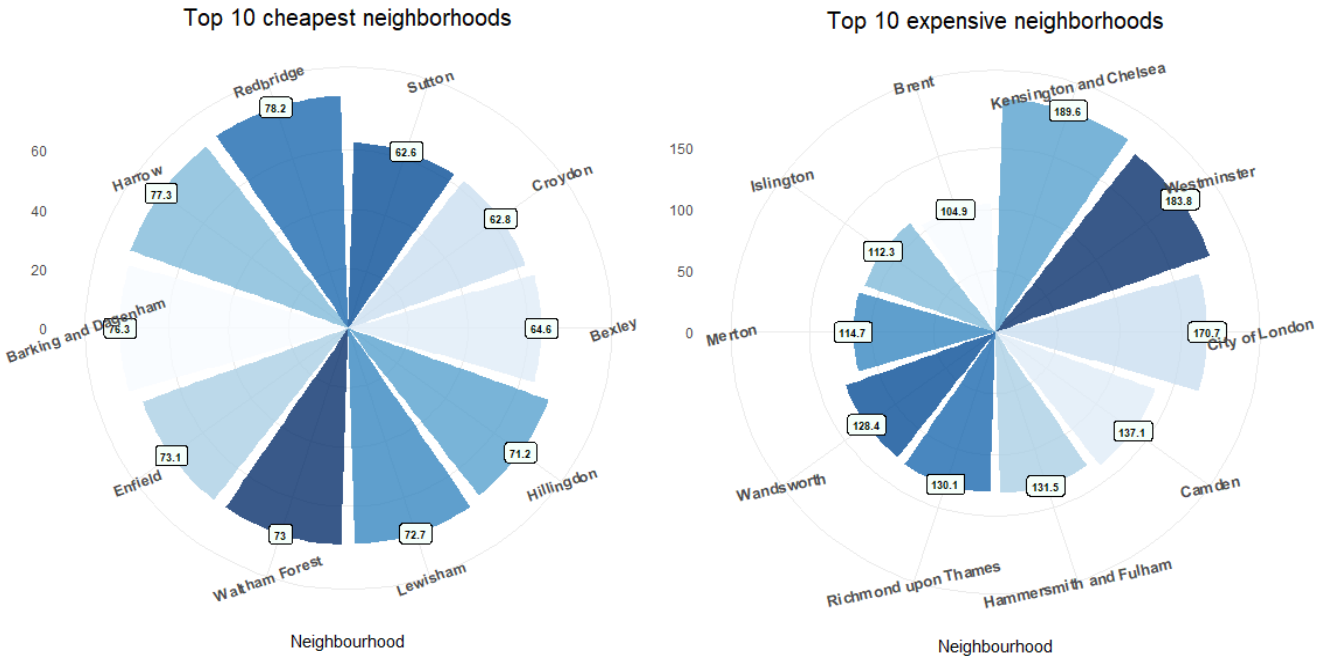



**Figure 4: Top 10 most cheap and expensive neighbourhoods**

## 2.4:- Analysis of Price and Property Types

Airbnb hosts properties of various types and other lodging options. The following graph shows the number of each of the different room types listed.

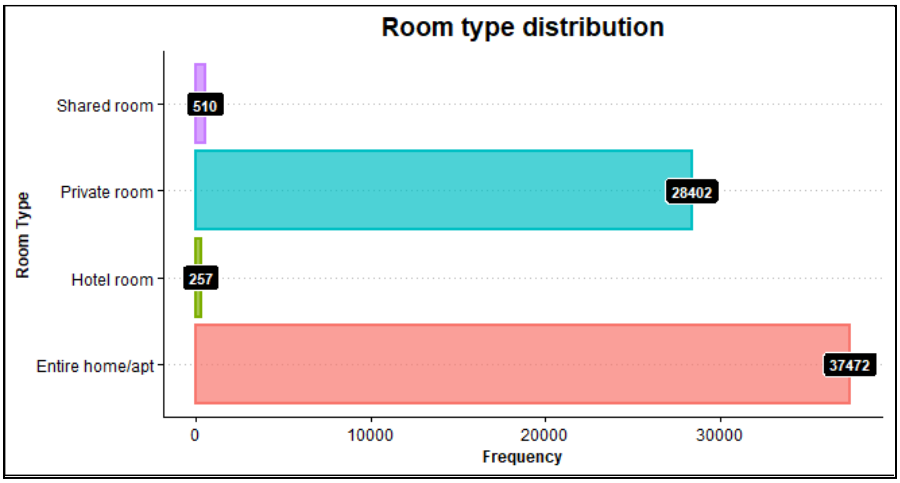



**Figure 5: Distribution of room type**

Interestingly, hosts are lending out their entire home more than an individual private room which was the original concept of Airbnb. There is also a small percentage of shared rooms and hotel rooms.

Next, we calculate the average price per listing based on the room type being rented out. Even though hotel room is a small percentage of the data, they have the highest price. This is followed by the prices of entire homes/apartments. Surprisingly, the price of a private room is slightly lower than the price of a shared room. This might be because there are more private rooms listed than shared ones, thus bringing the average price down. These insights are captured in the following graph -

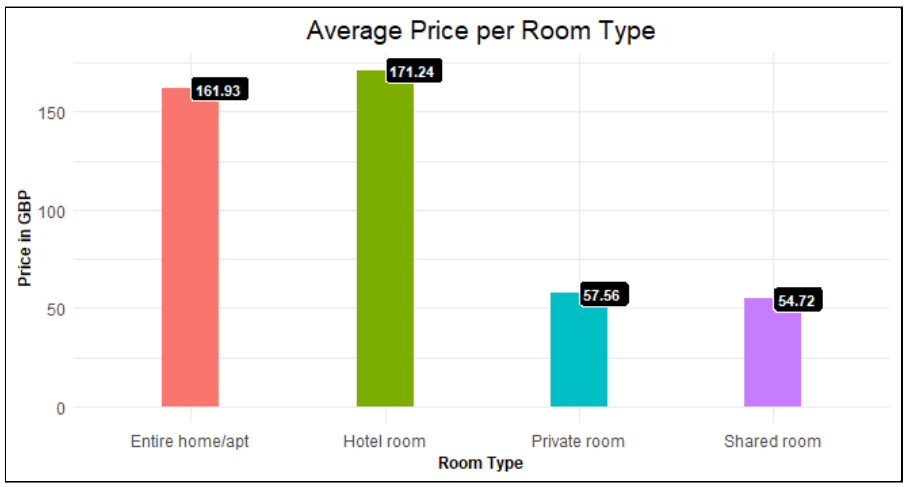



**Figure 6: Distribution of Average price per room type**

### 2.4:- Analysis of Minimum Nights of Stay required

Trends on the minimum nights required to make a booking show that over 90% of the properties require travellers to stay for 10 days or less at the least. The graphs below show a distribution of the minimum nights required by a guest to stay to make a booking. As expected, both the graphs are skewed to left -

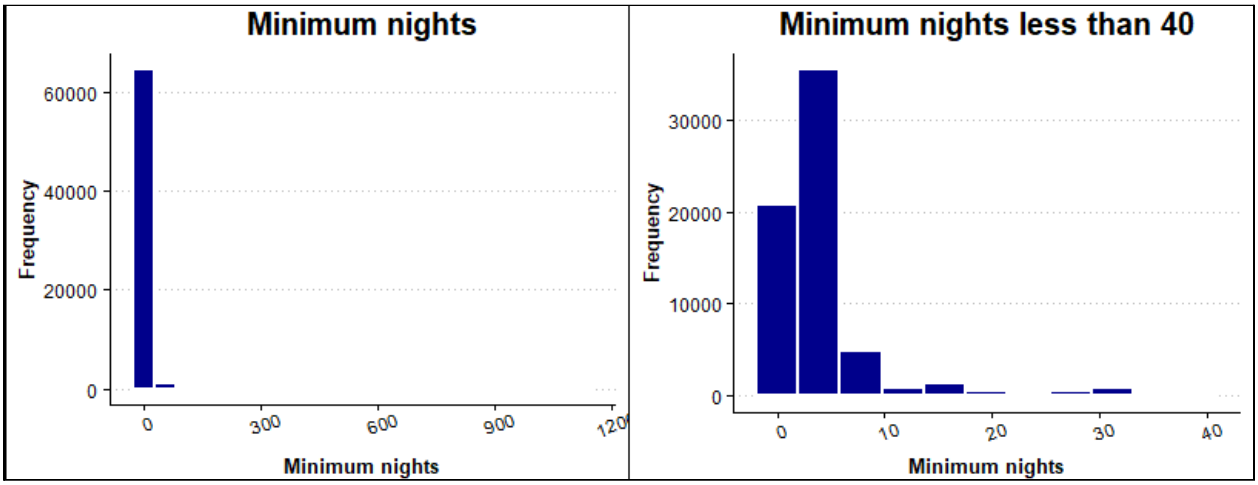



**Figure 7: Frequency distribution of minimum nights of stay**

### 2.5:- Correlation between Numeric Variables

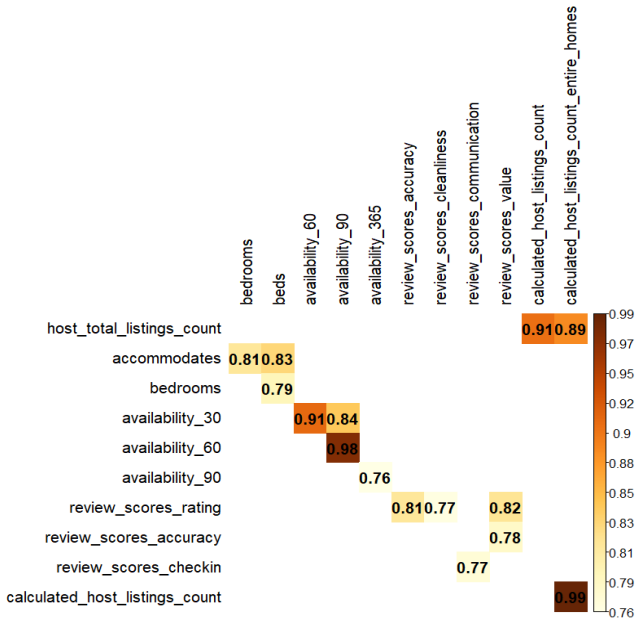
A simple correlation plot between the numeric variables reveals some intriguing insights. Due to a large number of variables, the correlations were put in baskets of 4 categories: High (>0.75), Moderate (between 0.25 and 0.75), Weak (between 0 and 0.25) and Negative(<0).

**Strong/Moderate Correlation:** The high correlation plot shows some expected relations between the number of accommodates and the beds/bedrooms. Availability in the last 30,60,90 and 365 days has a strong correlation between them as one is the subset of the other. The various review scores also exhibit a high to moderate correlation between them.

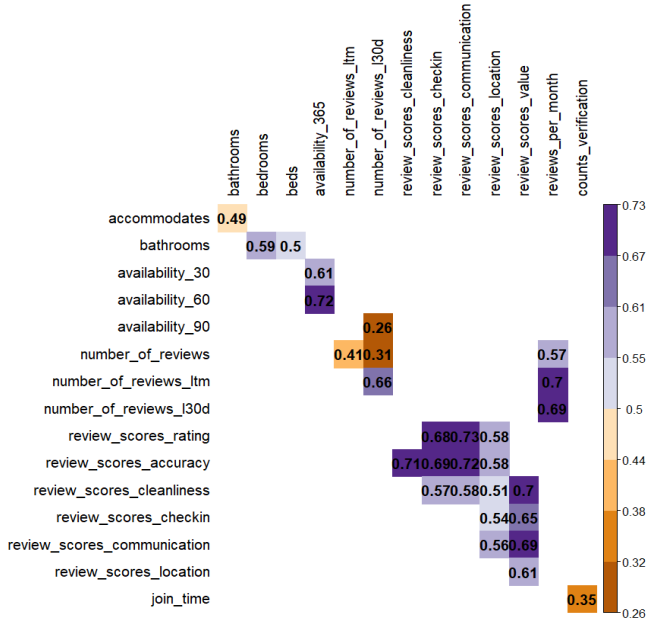
**Weak Correlation:** Bedrooms have a weak correlation with the number of guests (which is expected), the number of reviews with join time (implying that older hosts have more reviews), and availabilities with the total number of reviews.

**Negative Correlation:** Interestingly, the join time has a weak negative correlation with availabilities and reviews per month, meaning recent hosts are more likely to get more reviews and have more availability.

The correlation plots of various baskets are visualized on the following page -

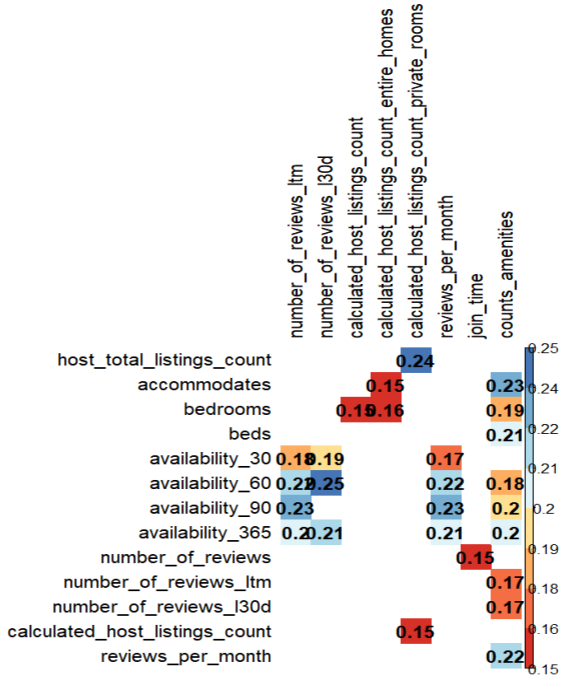



*Attributes with high correlation( >0.75)*

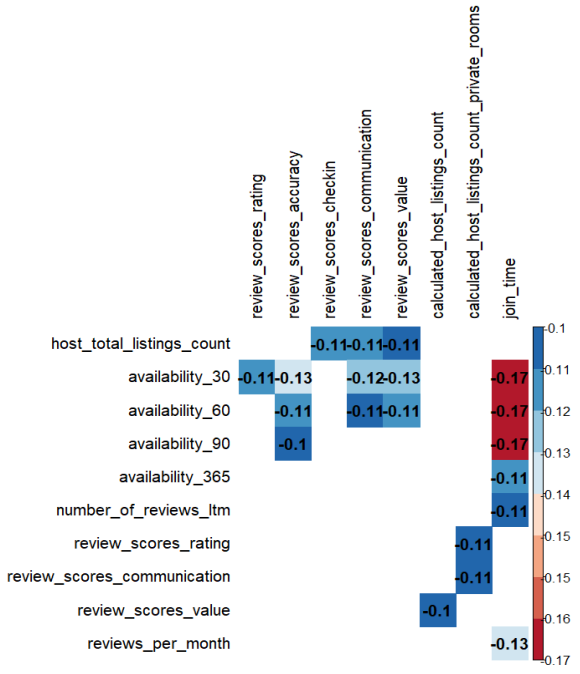



*Attributes with moderate correlation (>0.25 and <0.75)*

**Figure 8: Variables with High and Moderate Correlations**




*Attributes with weak correlation( <0.25)*




*Attributes with weak negative correlation*

**Figure 9: Variables with Weak Positive and Weak Negative Correlations**

# PART 4 - PREDICTION OF RATINGS USING ML TECHNIQUES

### 4.1: MODEL PERFORMANCE METRICS-

To evaluate the performance of each regression model for scores, we use the Root mean square error (RMSE). It is a commonly used measure for the difference between the predicted value and the observed value. It is calculated by

$$RMSE = \sqrt{\frac{1}{T}\sum_{i=1}^{T}(predicted\ value(i) - observed\ value(i))^2}$$

RMSE is initially calculated by the difference between the predicted value and observed value, then summed all the squared differences to aggregate the magnitude of errors. Then, we averaged squared errors and took a square root of it.

We divide the Airbnb dataset into 2 parts: the training set and the test set. The training set contains 70% of the data randomly selected from the Airbnb dataset. The test set contains the rest of the instances. The training set is used for building a regression model. Then we run the model on new data which is test data and calculate RMSE to evaluate its performance.

### 4.2: TARGET VARIABLES, PREDICTORS USED AND THEIR JUSTIFICATION-

- **Overall Rating**
  Rating score is a score for the overall experience given by customers after stays ranging from 0 to 5. A score of 5 means that the host was brilliant and a score of 0 means the host is bad. Future customers will select their host based on their rating score. We would like to use all attributes to build a regression model for this score since it reflects the overall feedback of the host. All attributes may have some possibility to correlate with rating scores.

- **Location**
  This score is based on the location of the property and a high score implies a good location with respect to connectivity, neighbourhood and region. We select latitude and longitude as attributes to predict the review score of location since latitude and longitude can locate a place accurately. We disregard the neighbourhood column for prediction as it can be captured by the coordinates themselves. From the correlation plot, Latitude is positively correlated with longitude. Latitude shows little correlation with the review score location, while longitude shows a negative correlation with the review score location. A position of low longitude will have a higher review score location.

- **Communication**
  The communication score is dependent on the guest's experience in communicating with the owner of the property. Our hypothesis is that host_response_time and response rate should be variables of importance along with instantly_bookable, host_is_superhost, host_total_listing_count. More listed properties and being a super host implies that the owner is a good host and exhibits excellent communication with guests.

- **Check-In**
  The check-in score is dependent on the guests' check-in experience when they enter the property. We believe that this score should depend on the instant_bookable attribute, host_acceptance_rate, host_is_superhost and host_response_time. Each of these variables is crucial to a certain extent in determining the check-in experience, although there might be other attributes of relevance which can be found from the decision tree on the entire dataset.

- **Accuracy**
  The accuracy score is for guests to mark how accurate the description and images of the property were with respect to what they received on arrival. Since there is no direct attribute with which accuracy can vary, we hypothesise that old hosts and super hosts with high amenities tend to have more accurate descriptions of their properties than others. Only hosts that have unusual/more amenities list them and guests can identify them leading to better scores, hence it should be an important attribute. This will be verified by the decision tree later.

- **Value**
  This score is based on the value of services the guests feel they get for the money they pay. Price should be a key attribute that this score depends on along with the host factors like host_is_superhost, instant_bookable, count of amenities etc.

- **Cleanliness**
  The cleanliness score is a metric for the guests to evaluate the cleanliness of the property upon their arrival. While there are no obvious metrics in the dataset that this score can depend on, a reputable host rating and other factors may have a slight correlation with this score. Again, this information can be extracted from the decision tree and a KNN algorithm can be applied from it.

### 4.3 DATA CLEANING

#### 4.3.1 Preliminary Cleaning -

- This step involved getting rid of 3 blank columns (neighbourhood_group_cleansed, calendar_updated, license) and extracting the number of bathrooms in the property from column bathrooms_text.
- Next, we calculated the number of days a host has been associated with Airbnb from the date type column host_since and then removed that column.
- We additionally removed over 30 columns that involved URLs / scrape IDs/ names and description texts
- Some columns had NA values as string literals and were hence converted to true NA values

#### 4.3.2 Data Processing for ML models -

- Remove listings with no stays or corrupted/incomplete reviews/ review scores
- Remove the "$" symbol and comma values and convert Price to numeric type
- Count the number of unique verifications and amenities required for each property and store them in a new column

- Conversion of top 90 most common amenities into their own dummy variables (0/1)
- Encode binary variables like host_is_superhost and other into 0/1 and categorical variables to 0/1/2/3 and so on.
- Remove irrelevant columns with too many missing values that couldn't be imputed (eg. host location)

#### 4.3.3 Data Splitting and Imputing values in Training set -

- Data was split into 70% training set and 30% testing set.
- Spaces had to be removed from column names as R throws various errors when they are there
- Imputation was done on values for beds, bedrooms and bathrooms based on the median values of each room type.
- Imputation was also done on host_acceptance_rate, host_response_rate and join_times calculated above using the median values.
- Character type variables were converted to factors and columns with more than 32 factors had to be omitted for decision tree models.

### 4.4 METHODS USED -

#### 4.4.1: Linear Regression-

Linear regression is used to model the linear relationship between attributes and predicted scores by the least-squares method. The least-squares method minimizes the sum of squared error. It assumes that the response variable has a linear relationship with the regressors. The model is formulated as, $y = \beta_1 x_1 + \ldots + \beta_p x_p$, where $\beta$ s are predicted from the training set.

Firstly, we convert all categorical attributes to numerical attributes by a function built-in R – as. numeric(). If attributes stay as categorical, in the linear regression model attributes would be less informative. For example, if we use superhost (binary attribute) as a regressor, the linear regression model will treat it as superhost1 and superhost0 and then give coefficients respectively. Knowing the coefficients of superhost1 and superhost0 will not give us any information. Then, we plot the distribution of each score by histogram. They all skewed to the right. It is reasonable since, in reality, most people are willing to give a high score like 5 for the host. We tried to build a linear model for each score and checked the residual plots. The QQ-plots are used for comparing two distributions graphically. Deviance from the dotted line shows that residuals are not normally distributed with heavy tails for each score and violates the assumption of linear regression. R-squared is a measurement of the good fit of a linear regression model. It can tell us how much variation is explained by the linear regression model. The table below summarizes the output of the regressions run -

| Target Variable | Training Data RMSE | Testing Data RMSE | R-Squared |
|---|---|---|---|
| **Rating** | 0.498 | 0.483 | 10.8% |
| **Location** | 0.421 | 0.433 | 0.0% |
| **Communication** | 0.451 | 0.437 | 8.1% |
| **Checkin** | 0.471 | 0.455 | 2.5% |

| | | | |
|---|---|---|---|
| **Cleanliness** | 0.565 | 0.542 | 9.0% |
| **Accuracy** | 0.490 | 0.478 | 9.1% |
| **Value** | 0.502 | 0.487 | 9.0% |

**Table 2: Training, Testing RMSE and R Squared values for Linear Regression Approach**

As we can see from Table 2, R-squared values are small for each score, especially for the location and check-in review scores. Low R- squared value implies that linear regression would not be a good method to predict review scores.
To fix it, We tried different transformations for scores (eg. taking the log, inverse, and square) however, none of them could give us a better prediction model.

### 4.4.2: Decision Tree Regression

A Regression Tree [4] is a tree where the target variable can take continuous values (typically real numbers). Decision trees are among the most popular machine learning algorithms given their intelligibility and simplicity. Early decision trees were only capable of handling categorical variables, but more recent versions, such as C4.5, do not have this limitation [5] and can handle continuous values too.

The hierarchy of attributes in a decision tree reflects the importance of attributes. [6] It means that the features on top are the most informative. [7] This information helps us extract variables to build our KNN model later on since running a KNN regression on over 100 independent variables and 40000+ instances is computationally expensive. We implement a 10 fold Cross-validation repeated 3 times and add training control parameters to generate trees with more leaves. The following figure shows us all the trees generated -

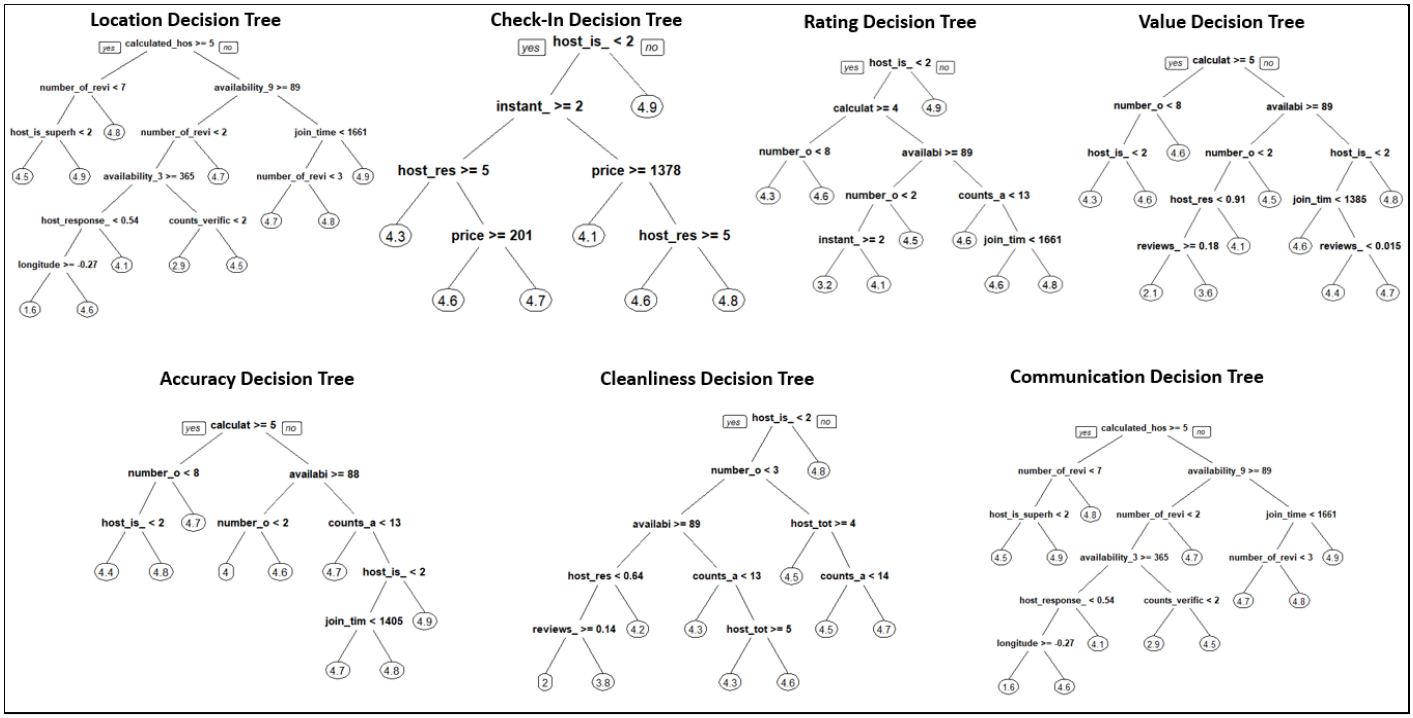



**Figure 10: Decision Trees for all the Target Variables**

Table 3 below summarizes the results of the decision tree regression ran for all target variables -

| Target Variable | Training Data RMSE | Testing Data RMSE |
|---|---|---|
| **Rating** | 0.498 | 0.482 |
| **Location** | 0.412 | 0.425 |
| **Communication** | 0.445 | 0.438 |
| **Checkin** | 0.219 | 0.203 |
| **Cleanliness** | 0.567 | 0.547 |
| **Accuracy** | 0.498 | 0.484 |
| **Value** | 0.487 | 0.499 |

**Table 3: Training vs Testing RMSE values for Decision Tree Approach**

Hence, we conclude from this section that the model is not overfitting on any of the scores and performs exceptionally good on Checkin scores and not much for others.

### 4.4.3: Random Forest Regression

Random forest regression is a form of ensemble learning. When a random forest is used for regression it can be thought of as trying to get a prediction from numerous models and averaging their answers. This method is popular due to the way this forest is “grown”. For each tree, a random sample of parameters is drawn. For regression empirical evidence seems to suggest that a third of the variables need to be chosen. Then this tree is constructed just like any normal decision tree and then added to the forest. It is this random sampling of variables that makes the trees uncorrelated and as a result, they tend to avoid overfitting and perform better than a single decision tree on unseen data. If the trees are allowed to be sufficiently deep then a random forest can capture complicated interactions. Furthermore, Random forests tend to be superior to bagging as bagging is not identically distributed but random forests are i.i.d. and therefore the whole forest doesn’t introduce bias. [3] Whilst these things make random forests attractive they do suffer from a lack of explainability and it is hard to work backwards from a result to figure out which collection of variables are causally linked to the output variable. My models tended to explain just over 10 % of the variance in the data which shows that these target variables, the scores, are quite complex, and things that do affect it, such as the host’s charisma, are probably not captured by the dataset. The software package I used defaults to the forest having 500 trees in it, however when I compared the mean squared error against the number of trees in the model, I saw that 100 trees perform just as good, but having too many didn’t result in lower performance either due to their nature to avoid overfitting. Looking at the table below our random forests seem to have test RMSEs within the range of training ones, perhaps with the location training results being just a little bit too overfitted.

| Target Variable | Training Data RMSE | Testing Data RMSE |
|---|---|---|
| **Rating** | 0.428 | 0.470 |
| **Location** | 0.355 | 0.412 |
| **Communication** | 0.391 | 0.430 |

| | | |
|---|---|---|
| **Checkin** | 0.400 | 0.438 |
| **Cleanliness** | 0.488 | 0.531 |
| **Accuracy** | 0.424 | 0.468 |
| **Value** | 0.434 | 0.478 |

**Table 4: Training vs Testing RMSE values for Random Forest Approach**

### 4.4.4: SVM Regression

Support vector machine regression is a non-parametric method, which doesn't assume any underlying distribution like decision tree-based algorithms and KNN. It works similar to the support vector classifier. In the support vector classifier, we want to find a maximum margin hyperplane to separate variables and thus classify new points. It relies on the choice of kernel. The kernel can help us to find the hyperplane in higher dimensional spaces. In the support vector regression, we don't care about predictions as long as their error is less than a certain value.[10][11] But it is computationally expensive and our instances are over 40000 in the training set, R only gives an output for location score with linear, radial and polynomial kernel and check-in score with the linear and radial kernel. In the case of location score, there is overfitting for linear, radial and polynomial kernels. When we plot the graph of longitude and latitude as x, y and denote the location score by colours, we found that longitude and latitude with a high location score will form a circle. Then we would like to use a radial kernel to predict the location score. The overfitting is caused by the tuning parameter. We should relax the epsilon. In the case of check-in review score, we performed SVM regression with the linear and radial kernel and showed a better result with the radial kernel. It may suggest a non-linear relationship between the check-in score and attributes. It is confirmed with the R-squared value in linear regression, suggesting check-in and location score doesn't have a linear relationship with attributes.

| Target Variable | Training Data RMSE | Testing Data RMSE | Best Performed Kernel |
|---|---|---|---|
| **Location** | 0.426 | 0.438 | Radial |
| **Checkin** | 0.494 | 0.477 | Radial |

**Table 5: Training vs Testing RMSE values for SVM Regression**

### 4.4.5: Bagging and Boosting

Bagging is a form of bootstrapping and is sometimes known as bootstrap aggregation. It is somewhat similar to random forests. It hopes to reduce variance by taking several samples of the target variable and averaging them. It is a collection of decision trees, however instead of using a square root for classification or a third of the variables for regression, we use all of them. Also unlike a single decision tree, which is usually pruned to reduce overfitting, bagging doesn't and lets the trees grow all the way. The rationale is that this is bad for a single tree but gets averaged out by bagging. However, as bagging is allowed to choose any of the variables they will tend to choose the same couple of variables as their top predictors in the first split, therefore resulting in them looking quite similar, not being independent of one another to a sufficient degree. This results in bagging not being the best way to reduce variance.[2] Bagging did result in slightly higher RMSE than random forest but surprisingly by not that much as can be seen in the table below.

Boosting can be used to generate good estimators using weak learners which on their own could have low performance. Therefore boosting is a method that can use any learner but here we used decision trees. When the decision trees used have depth one, they are called stumps, and result in an additive model. These models therefore are actually explainable and aren't black boxes.[2] However the R package I used let me set data aside for validation. This allowed me to tune the parameters, and unfortunately using decision trees of about 3 seemed to perform better but again at the cost of explainability. Also as it was easier to tune the number of trees used in boosting, I was able to choose the number of trees where validation loss is at its lowest, and therefore training and testing RMSE are a bit tighter together. I had to do this as boosting is sequential in nature where subsequent trees are trained and scaled on the residuals of the previous trees, and as this isn't done in a bootstrapping manner, overfitting can occur, especially as some trees, or in case of stumps, some variables will be favored by the algorithm. Lastly a training rate parameter was used to slow down the training and prevent overfitting but at the cost of having to train more trees.

| Target Variable | Training Data RMSE (Boosting) | Testing Data RMSE (Boosting) | Training Data RMSE (Bagging) | Testing Data RMSE (Bagging) |
|---|---|---|---|---|
| **Rating** | 0.481 | 0.475 | 0.427 | 0.473 |
| **Location** | 0.389 | 0.415 | 0.353 | 0.414 |
| **Communication** | 0.431 | 0.435 | 0.390 | 0.433 |
| **Checkin** | 0.443 | 0.444 | 0.400 | 0.441 |
| **Cleanliness** | 0.533 | 0.541 | 0.486 | 0.534 |
| **Accuracy** | 0.470 | 0.471 | 0.422 | 0.471 |
| **Value** | 0.475 | 0.482 | 0.432 | 0.480 |

**Table 6: Training vs Testing RMSE values for Bagging and Boosting**

### 4.4.6: k-Nearest Neighbours Regression

The k-nearest neighbours' algorithm (k-NN) is a supervised learning methodology that is non-parametric in nature.[8] The output of k-NN regression is the object's property value. This value is the average of the k closest neighbours' values. To avoid the effects of the curse of dimensionality [9], dimension reduction is usually performed prior to applying the k-NN algorithm to high-dimensional data (e.g., with more than ten dimensions). In our work, we utilize variables of most importance extracted from Decision Tree regression in section 4.4.2 above to make predictions on the scores.

We implemented a 10 fold cross-validation repeated 3 times for alternate values of k from 1-59. Surprisingly, all the best results were achieved for k = 59, implying that a higher k value can further improve the results obtained. But this could not be implemented due to computing and time limitations. The following table summarizes the best training and testing RMSE values achieved -

| Target Variable | Training Data RMSE | Testing Data RMSE |
|---|---|---|

| | | |
|---|---|---|
| **Rating** | 0.505 | 0.489 |
| **Location** | 0.409 | 0.420 |
| **Communication** | 0.452 | 0.440 |
| **Checkin** | 0.470 | 0.450 |
| **Cleanliness** | 0.574 | 0.551 |
| **Accuracy** | 0.493 | 0.480 |
| **Value** | 0.506 | 0.491 |

**Table 7: Training vs Testing RMSE values for KNN**

Therefore, from this method, the best results were obtained for the prediction of Location score. The variables on which the regression was run were limited due to computational constraints. Oftentimes, the cases, where the regression could be run, were giving warnings for unbreakable ties (presumably due to the Curse of Dimensionality) and led to NA values for RMSE, hence more variables were eliminated to achieve these results.

### 4.4.7: Polynomial regression

Polynomial regression is used to model the relationship between the dependent variable (review score) and the attributes in pth degree. A higher degree of the polynomial regression, the more flexible the model is. In this case, the location score only has two attributes and it is easy to perform a polynomial regression. The best model is when the degree is 7. For other scores, they have a large number of attributes and it is hard to find the most suitable degree in r. In this case, there is no overfitting.

| Target Variable | Training Data RMSE | Testing Data RMSE |
|---|---|---|
| **Location** | 0.427 | 0.415 |

## PART 5 - SUMMARY OF RESULTS AND CONCLUSIONS

| Method\Score | Rating | Location | Communication | Checkin | Cleanliness | Accuracy | Value |
|---|---|---|---|---|---|---|---|
| **KNN** | 0.489 | 0.420 | 0.440 | 0.450 | 0.551 | 0.480 | 0.491 |
| **Bagging and Boosting** | 0.473 | 0.414 | 0.433 | 0.441 | 0.534 | 0.471 | 0.480 |
| **Random Forest** | 0.470 | 0.412 | 0.430 | 0.438 | 0.531 | 0.468 | 0.478 |
| **Decision Tree** | 0.482 | 0.425 | 0.438 | 0.203 | 0.547 | 0.484 | 0.499 |
| **Linear Regression** | 0.483 | 0.433 | 0.437 | 0.455 | 0.542 | 0.478 | 0.487 |
| **SVM** | | 0.438 | | 0.477 | | | |
| **Polynomial Regression** | | 0.415 | | | | | |

PCA analysis shows that the rating score has an overwhelming large eigenvalue of over 5 compared to other scores which are less than 1, which is the most important score variable we need to predict and explain over 70% variance. The second and third largest eigenvalue is for cleanliness and accuracy scores. Their eigenvalues are both around 0.5. In the linear regression, rating score is mainly related to communication with the host, whether the host is superhost, description and amenities of the listings, listings' availability, number of listings host had and join time. If the host has WIFI, essentials, kitchen, TV, lockbox and backyard and provides breakfast and access to the gym, they will get a higher review score. It is highly coincidental with the criteria when we select the host in Airbnb. In the decision tree, whether the host is a superhost, total listings host had and availability in 90 days will play important roles in determining the scores.

**Ethical Implications:**
Our discovery of scores correlating with latitude/longitude could result in hosts buying property in certain locations, reducing access to housing in those communities. It could also lead to hosts in this neighbourhood increasing their costs, charging more from travelers to the city.

**Strengths and weakness:**
We use non-parametric algorithms to build models for each score, which does not assume any underlying distribution for variables. And we encode the amenities into dummy variables, thus making our models more accurate and close to reality. Tree-based algorithms were the best models for all scores and then our models are easy to explain.

Number of reviews plays an important role for the main prediction review rating. In the model, more reviews will lead to a higher review rating score. But in reality, the best host may have a slightly lower score than expected because they have a large number of reviews. For example, some hosts may have a rating score 5 but with one review. Some hosts may have a rating score at 4.8 with 100 reviews.The customer may tend to the later. Number of reviews may raise causality problems. In the current model, treating it as an attribute would cause some problems.

## REFERENCES

[1]Data : http://data.insideairbnb.com/united-kingdom/england/london/2021-12-07/data/listings.csv.gz

[2] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. (2013). An introduction to statistical learning : with applications in R. New York :Springer

[3] Hastie, T.; Tibshirani, R. & Friedman, J. (2001), The Elements of Statistical Learning, Springer New York Inc., New York, NY, USA.

[4] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software. ISBN 978-0-412-04841-8.

[5] Rokach, Lior; Maimon, O. (2014). *Data mining with decision trees: theory and applications, 2nd Edition*. World Scientific Pub Co Inc

[6] Provost, Foster, 1964- (2013). Data science for business : [what you need to know about data mining and data-analytic thinking]. Fawcett, Tom. (1st ed.). Sebastopol, Calif.: O'Reilly. ISBN 978-1-4493-6132-7. OCLC 844460899.

[7] Piryonesi S. Madeh; El-Diraby Tamer E. (2020-06-01). "Role of Data Analytics in Infrastructure Asset Management: Overcoming Data Size and Quality Problems". Journal of Transportation Engineering, Part B: Pavements. 146 (2): 04020022. doi:10.1061/JPEODX.0000175.

[8] Fix, Evelyn; Hodges, Joseph L. (1951). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. USAF School of Aviation Medicine, Randolph Field, Texas.

[9] Beyer, Kevin; et al. "When is "nearest neighbor" meaningful?". *Database Theory—ICDT'99*. **1999**: 217–235.

[10] https://www.kdnuggets.com/2017/03/building-regression-models-support-vector-regression.html

[11] https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/