# QBUS3820 Machine Learning & Data Mining in Business (2022 Semester 1)

Group Assignment: Airbnb Regression Project

SIDs: 490032125, 490388169, 500202034

## Contents

# 1 Introduction

Airbnb rental is an online platform for home-stays, vacation rentals, and tourism activities that links hosts of the housing with potential rental customers. In 2020, Airbnb had over 5.9 million active listings across over 100,000 cities, a 9% increase from the previous year (AllTheRooms, 2021). The purpose of this report is to analyse and develop market advice to hosts, property manager, real estate investors and other stakeholders, through the use of data analytics techniques.

Our project follows closely to the Cross-Industry Standard Process for Data Mining (CRISP-DM), starting with business understanding, followed by data understanding, data preparation and modelling, and finally evaluation (IBM, 2021). This report seeks to define the business problem, followed by a preliminary investigation of the provided Airbnb dataset before cleaning and preparing the data. Next, different machine learning models will be fit and compared to assess the prediction accuracy on the response variable, the rental price of an Airbnb listing in Sydney. Finally, we pick the 'best' predictive model and provide an analysis and conclusion of result.

In addition, as investors in the real estate market, property owners or hosts are definitely keen on understanding what drives the rental income. This report also seeks to address the insights uncovered from our process of data mining of the Airbnb dataset, such as what are some of the most important features in determining rental prices and what are some of the best hosts doing.

# 2 Problem Formulation and Objectives

To achieve our goal of developing an advice, the main focus of our project is divided into two parts - building a predictive model for rental prices, and discovering insights from the provided dataset that can help hosts to make better decisions.

A statistical model using supervised learning, or commonly known as "machine learning", can meaningfully help hosts to increase revenue and reduce costs. Such model is able to study the existing data, and output predictions based on a combination of predictors. Through a machine learning model, the host is able to understand what predictors do not contribute to the rental price much, as well as improve the property and host behaviour accordingly to most efficiently increase the rental price.

Airbnb rental price can be influenced by many factors, such as the size of the property, location, availability, online reviews of that property, host responsiveness. It can be formulated into a regression problem where the response variable, `price`, is regressed the potential predictors from the dataset so a predictive model can be built to predict the rental price. One of the primary objectives of this project is to find the "best" predictive model that models the rental price with highest prediction

accuracy based on the given dataset.

The analysis of the data can help us inform our recommendation to hosts and potential investors in the real estate market. Investors are likely to be keen on knowing what factors drive rental prices before making a decision. While for investors who already has a property, it is almost infeasible to change any attributes of the property (e.g., size and location) to increase the rental price and the competitiveness of the property in the rental market. However, the behaviours of the host may be amended to become favourable to the renters on Airbnb, thus increasing the rental price and the competitiveness of the property. Our second objective of this project is to obtain at least three insights about what the best hosts are doing, in order to help hosts on the Airbnb platform to make better decisions.

# 3 Data Pre-processing and Cleaning

The provided dataset, `listings.csv`, was scraped from Airbnb, containing detailed information on a number of existing Airbnb listings in Sydney. It contains a lot of information including the rental price, geolocation data, property type, room type, number of bedrooms and more. The data types are mixed, including text, numerical and boolean data. An accompany data dictionary was also provided, detailing the data types and description of each column.

Given such a large number of potential features, there is a necessity to properly pre-process and clean the data before conducting any exploratory data analysis.

## 3.1 Missing Values Analysis

Figure 1 shows a heatmap of missing values present in the dataset. Visually, it is obvious that there is a large proportion of missing values present in this dataset. This informs us in the later stage of feature selection to deal with those features which has a large number of missing values carefully. It can also be observed that there are three columns with completely no values at all: `neighbourhood_group_cleansed`, `bathrooms` and `calendar_updated`. They will simply be dropped, as it makes no sense to purposely perform imputation in these columns.

## 3.2 `dtypes` Conversion

The `Pandas` library from Python automatically treats numbers as "`float64`" or "`int64`", or "`object`" if there are words or punctuation involved. Based on the accompanying data dictionary, we are able to identify the data types of each column. In this subsection we performed `dtype` conversion to convert some of the columns into their respective supposed data types, in preparation for the exploratory data analysis and training of the machine learning models in the later sections.

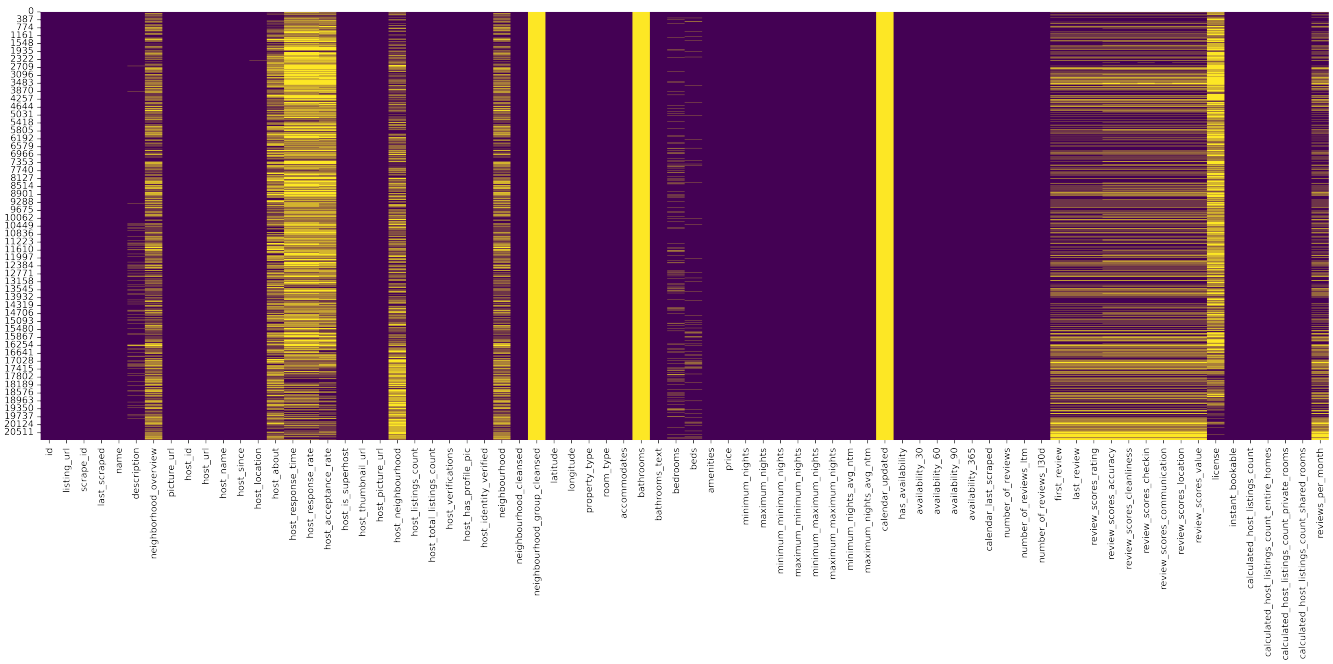



Figure 1: Heatmap of Missing Values.

For example, the column `last_scraped` was interpreted by `Pandas` as an "`object`", when it should have been a datetime datatype. We converted it from "`object`" to "`datetime64`" to avoid confusion in later analysis. For the "`boolean`" datatypes, we have decided to convert it to "categorical" along with the other categorical columns which were not correctly interpreted to allow easier analysis. Some of the categorical variables with large number of levels, such as `neighbourhood_cleansed`, were temporarily ignored and left as "object" due to its potential high cardinality. The details of the columns converted can be found in the accompanying python file and Appendix A.

# 4 Exploratory Data Analysis

In this section we conduct a preliminary investigation of the Airbnb listings through exploratory data analysis, where we extract useful information so we can learn more about this dataset.

## 4.1 Univariate Analysis

We first conduct a univariate analysis on all the variables, including both features and response. The univariate analysis will inform us the summary statistics they have and what kind of distribution each of the variables are following.

#### 4.1.1 Descriptive Statistics

Table 11 in Appendix B summarises the descriptive statistics of numerical data. The descriptive statistics provide some fundamental information on the numeric variables of the dataset. It is also useful in identifying potential data errors by looking at min and max values that may not align with the data dictionary.

Some useful preliminary observations include the fact that our response variable, `price`, has a mean of $251.17, but the median (50%) is at only $140.00, which means that `price` is highly right-skewed. This will be explored further in the next sub-subsection. We also note that `host_listings_count` and `host_total_listings_count` have the exact same descriptive statistics. This could not happen by chance, and shall be further investigated later.

Table 1 summarises the descriptive statistics of the boolean and categorical variables. Unlike the descriptive statistics for numerical variables, Table 1 only summarises the count, unique (number of categories in that variable), top (category with highest frequency) and freq (top category's frequency).

| | count | unique | top | freq |
|---|---|---|---|---|
| host_is_superhost | 20877 | 2 | False | 17841 |
| host_has_profile_pic | 20877 | 2 | True | 20778 |
| host_identity_verified | 20877 | 2 | True | 16295 |
| has_availability | 20880 | 2 | True | 20597 |
| instant_bookable | 20880 | 2 | False | 13516 |
| host_response_time | 7884 | 4 | within an hour | 4585 |
| room_type | 20880 | 4 | Entire home/apt | 13398 |

Table 1: Descriptive Statistics of Boolean & Categorical Variables

#### 4.1.2 Numerical: Scatter Plots

Figure 8 in Appendix C shows a scatter plot of each individual numerical variables. The plots are easy to interpret. The concentration of points in each plot roughly tells the distribution of each variable. For example, all the `availability_`s are fairly distributed throughout, with a slightly higher concentration on the top. On the other hand, `price` is heavily concentrated at the bottom, with extremely marginal points (listings) scattering above $5000. These are perhaps the "rare" extremely priced luxurious apartments that are costly, which was not meant for a budget-friendly accommodation services site such as Airbnb.

#### 4.1.3 Numerical: Box Plots

Similarly, Figure 9 in Appendix D shows a box plot of each individual numerical variables. Under an assumption of normal distribution, a box plot should be symmetrical with mean and median in the center, and few outliers. However, as seen from Figure 9, many of the variables have a large number of "outliers" indicated by the round circles. This is because of the fact that our variables are highly skewed, and cannot be fit into a normal box plot.

#### 4.1.4 Numerical: Histogram Plot & Density Curve

Figure 10 in Appendix E tells similar story to that of scatter plots and box plots. The histogram plot with estimated kernel density curve illustrates clearly that none of the numerical variables follow a normal distribution and that some variables, such as `price`, are highly skewed as well.

#### 4.1.5 Categorical: Violin Plots

Next, we will move on to the analysis of categorical variables. Figure 11 in Appendix F shows violin plots of all the individual categorical variables. A violin plot can be considered a combination of box plot and kernel density plot, which means it has the advantage of visualising the distribution of the variable (Lewinson, 2019). It is worth noting that the categories of these variables are measured against the response variable, `price`.

As seen from Figure 11, just like the numerical variables, the categorical variables are highly skewed as well. As such, it is sufficient to conclude at this stage that our features in its entirety are not suitable in its raw form for the purposes of training our machine learning models, and that they should be dealt with accordingly during Feature Engineering.

#### 4.1.6 Count of Neighbourhoods

Lastly, before we move on to bivariate and multivariate analysis, we investigate a potential variable that we did not previously considered. The variable `neighbourhood_cleansed` indicates which particular neighbourhood in Sydney the listing belongs to. Previously, we did not consider converting this variable due to its potential to cause high cardinality. However, we acknowledge that the neighbourhood (location) of an Airbnb listing should have an impact on the rental price of the listing. As such, we observe the count of neighbourhoods in Figure 12. We observed that there is potential for converting some of the neighbourhoods with lower frequencies into a new category "others" so as to reduce the high cardinality problem.

### 4.2 Bivariate Analysis and Multivariate Analysis

In this subsection, we want to examine the relationship between variables. Specifically, we want to investigate the relationship between the response and the features, as well as just between features.

### 4.2.1 Correlation between Response and Features

First, we explore the correlation between the response variable, `price`, and each of the predictors. Tables 2 and 3 respectively shows the Pearson and Kendall correlation of the top 10 features with price. Unsurprisingly, we find that some of the higher correlated variables include `bedrooms`, `accommodates` and `beds`, which are some of the common prominent factors in affecting rental prices.

We can also observe some differences between the Pearson correlation and Kendall correlation even within the top 10 features. Kendall correlation is a non-parametric model that determines monotonic relationships (Magiya, 2019). As such, unlike Pearson correlation which is limited to measuring only the linear relationships, Kendall correlation can take into account of non-linear relationships.

| | |
|---|---|
| price | 1.0 |
| bedrooms | 0.4755284069742047 |
| accommodates | 0.4342783999222104 |
| beds | 0.4026069469767801 |
| calculated_host_listings_count_entire_homes | 0.10513049140467347 |
| calculated_host_listings_count | 0.08366105311267902 |
| host_listings_count | 0.07999849317712121 |
| host_total_listings_count | 0.07999849317712121 |
| availability_365 | 0.07003043612747227 |
| review_scores_location | 0.05149490452180253 |

Table 2: Pearson Correlation of `Price` and features. Only top 10 features are shown.

| | |
|---|---|
| price | 1.0 |
| accommodates | 0.553427144932982 |
| bedrooms | 0.5423753135041901 |
| beds | 0.4757916575660866 |
| calculated_host_listings_count_entire_homes | 0.4290718277191348 |
| review_scores_location | 0.13124457820159535 |
| availability_365 | 0.12155209130241286 |
| review_scores_rating | 0.0927776407331953 |
| number_of_reviews_ltm | 0.09196397013869713 |
| number_of_reviews_l30d | 0.08142491309780926 |

Table 3: Kendall Correlation of `Price` and features. Only top 10 features are shown.

### 4.2.2 Correlation between Features

Next, we will explore the relationship between features. A Kendall correlation heatmap between the predictors is plotted in Figure 13 in Appendix H. We can observe from Figure 13 that there are many predictors which are highly correlated with one another, thus creating a multicollinearity situation. These high correlations mainly occur within its contextual groups (i.e., minimum and maximum

nights, availability, and review scores).

For instance, the predictors relating to review scores contain an overall rating `review_scores_rating`, and a more detailed breakdown of the ratings - accuracy, cleanliness, check in, communication, location, and value. An increase or decrease of the smaller breakdown rating will affect the overall rating, thus, it creates a strong correlation between these predictors.

It is also observed that `host_listings_count` and `host_total_listings_count` have correlation of exactly 1, suggesting these two fields are exactly the same and one of them could be removed as it is redundant.

We also find very strong correlation between `availability_30`, `availability_60`, `availability_90`, and `availability_365`. As observed in Figure 2, all of the pairplot display linear relationship between the predictors. Contextually, the strong correlation is expected, as the availability measures the number of days the listing is available as determined by the calendar. This is suggesting the number of days a listing is available for 60 days would include the measurement of the number of days that the listing is available for 30 days. Therefore, a strong correlation is evident between these predictors.

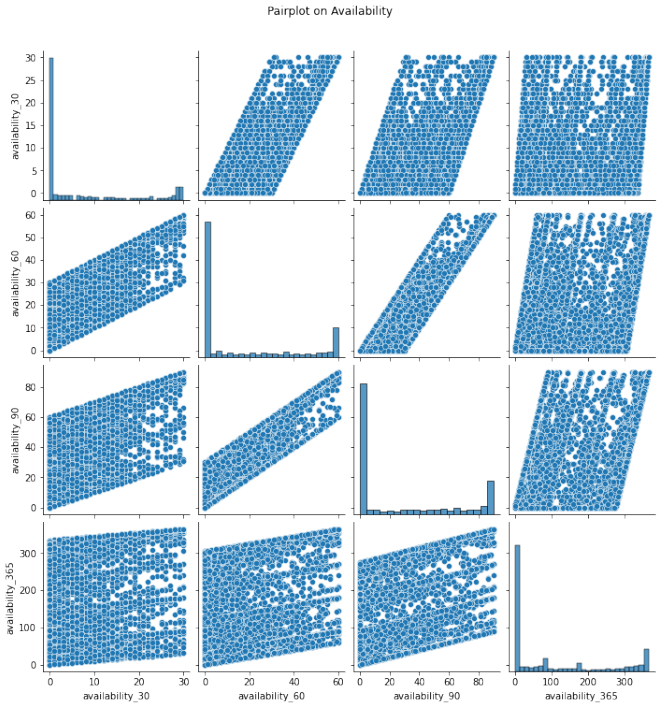



Figure 2: Pairplot on Availability

Our analysis finds that some groups of predictors are highly correlated due to the nature of these groups of variables. It must be taken into consideration during feature engineering and selection in order to minimise the problem of multicollinearity when we train our machine learning models, as multicollinearity can weaken the statistical power of a regression model (Frost, 2017).

### 4.3 Geolocation Analysis

Finally, we move on to consider the effect of locations on price. Figure 20 in Appendix P shows a scatterplot heatmap of Airbnb listings in Sydney, generated using Tableau. Each dot represents a listing in Sydney, with its size and colour representative of the price. The higher the price, the warmer (more red) the colour becomes, and the larger the dot becomes. We can observe from Figure 20 that majority of the listings share a similar shade of colour, which means that most of the listings are priced in a similar (lower-end) range. We can also notice that despite the similar range in price, there is also a noticeable difference in certain areas by comparing the size of the dot. Last but not least, the distribution of rental listings seems to be focused around the coastal areas and centre of Sydney.

It appears that while minimal, locality does have an impact on the price of an Airbnb listing. With that in mind, it is reasonable for us to reconsider the `neighbourhood_cleansed` variable which we did not take into account previously due to its potential high cardinality. This will be explored further in Section 5.2 under Feature Engineering.

### 4.4 Conclusion: EDA

The preliminary exploratory analysis performed on our Airbnb listings dataset provided multiple useful insights. Specifically, our variables, both numerical and categorical, are highly skewed and should be transformed or treated before feeding into our machine learning models. Additionally, the geolocation analysis finds that locality can have an impact on the price, and as such it might be useful to consider the variable `neighbourhood_cleansed` as a feature, but it has to be treated as well to minimise the problem of high cardinality.

## 5 Feature Engineering

Domingos (2012) mentioned in his book, "A Few Useful Things to Know about Machine Learning", '... Some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used.' The choice of features and how to correctly treat these features are crucial steps in building a machine learning model. In this section we conduct feature engineering on the feature variables accordingly to what we have identified in the previous section.

## 5.1 Handling Missing Values

We first would want to deal with the missing values in the dataset. As pointed out in Section 3.1, there is a high proportion of missing values present in the `listings` dataset, and as such it is infeasible to remove all these missing values. Instead, we perform imputation on these missing entries. For the numerical variables, they will be imputed with their median, whereas for the categorical variables, they will be imputed with mode.

## 5.2 Categorical Variables

### 5.2.1 Adding New Categorical Features

In Section 4.3 we identified the potential for locality to be a feature for consideration. Here, we take into consideration of the `neighbourhood_cleansed`. The variable `neighbourhood_cleansed` contains the name of the suburb where a listing is located. In the dataset, there are 38 unique values of the neighbourhood. Such high cardinality predictor is not suitable for one-hot encoding or dummy encoding as a high dimensionality space could lead to potential curse of dimensionality. To reduce the cardinality of this field, suburbs with low occurrence (less than 250 listings) in the dataset are aggregated into 'Other category'. Figure 3 shows the number of listings within each of the suburb after aggregation. The cardinality of this predictor is reduced from 38 to 21.

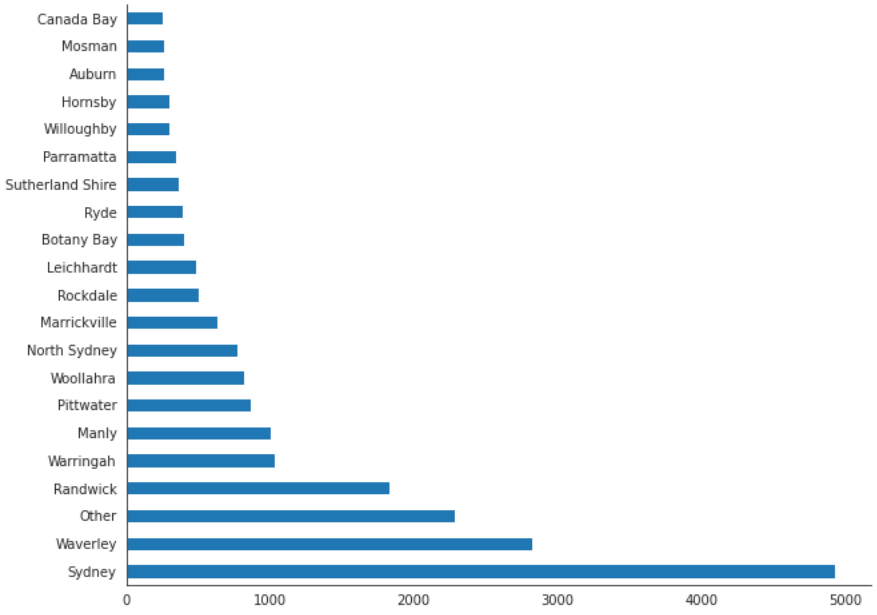



Figure 3: Bar Graph of Aggregated Neighbourhood

Apart from `neighbourhood_cleansed`, we also looked at the potential of the variable `property_type`. It describes the rental information in more detail, such as whether the listing is a private room in a

rental unit or the entire rental unit. There are 85 different property type in this dataset. A similar aggregation approach is used, that if a property type has less than 100 listings, then it is aggregated into 'Other' category. As shown in Figure 4, the cardinality after aggregation is reduced from 85 to 16.

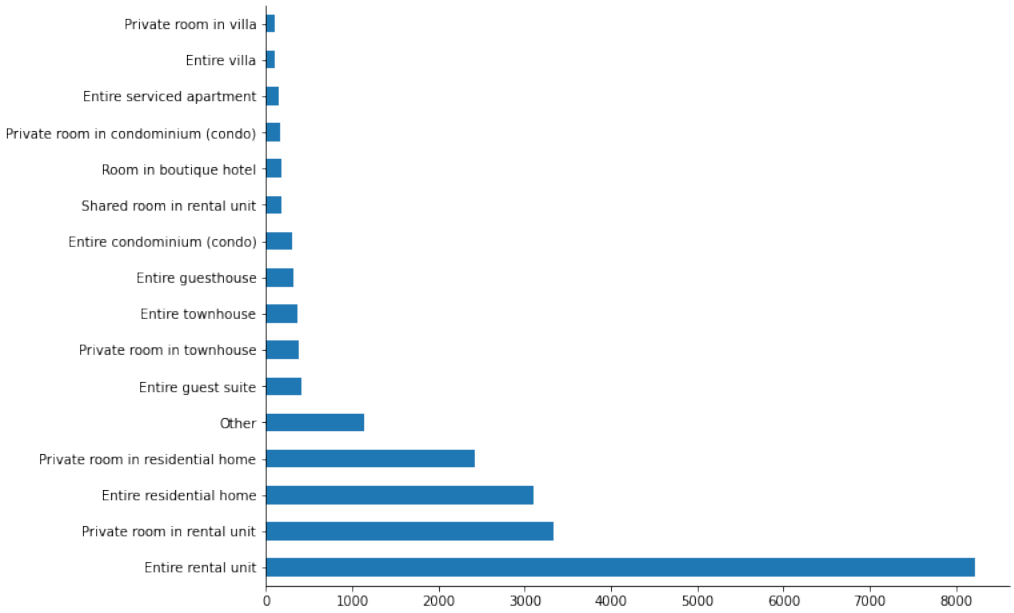



Figure 4: Bar Graph of Aggregated Property Type

### 5.2.2 Encoding

Categorical variables in the dataset have either a cardinality of 2, or a cardinality of 2 or more. These variables are encoded to numeric values, in order to be used in the machine learning models.

The boolean predictors are nominal categorical variables with the value of either True or False. The fields `host_is_superhost`, `host_has_profile_pic`, `host_identity_verified`, `has_availability`, and `instant_bookable` are boolean type values. Dummy encoding method constructs a binary indicator for each category like one-hot encoding, then delete one feature to avoid perfect collinearity. It encodes False as 0, and encodes True as 1. This method is used for the Boolean predictors in this machine learning study, as it creates only one dummy variable for each category and does not increase the dimensionality of the dataset.

For high cardinality categorical variables, dummy variable encoder is not suitable as it would lead to sparse data and too many columns. Therefore, CatBoost encoding, which is a target-based encoding method, was utilised, since it only transforms the data into numerical values without producing more columns. CatBoost encoding is similar to target encoding, which replaces a categorical feature with

the average of the response variable within that category and the probability of that category over the entire dataset. One of the disadvantage of target encoding is target leakage (Saxena, 2020), as it utilises the response variable into transforming the training data. However, CatBoost overcomes this disadvantage using an ordering principle (GeeksForGeeks, 2021).

### 5.3 Numerical Variables

#### 5.3.1 Box Cox Transformation

From the results of our exploratory data analysis, we have concluded that most of our variables, both categorical and numerical, are skewed. There is a need to address the skewness before fitting the features into our machine learning models.

We have decided to only address features where their skewness is serious and requires transformation. Table [**insert table later**] some of the variables with a very high skew index, starting from the most skewed variable. We have decided to perform a Box Cox transformation on features whose skew index is above 0.5.

The Box Cox transformation is one of the means to "normalise" a univariate data. While it is usually performed on the dependent variable, here we performed the Box Cox transformation on the features instead. The univariate objective of a Box Cox transformation is generally to create a transformed variable that is more "normally" distributed. In the context of regression, transformation of dependent or independent variables can often reduce the complexity of the model required to fit the data (Lalonde, 2012). (See Appendix I for the formula of Box Cox transformations)

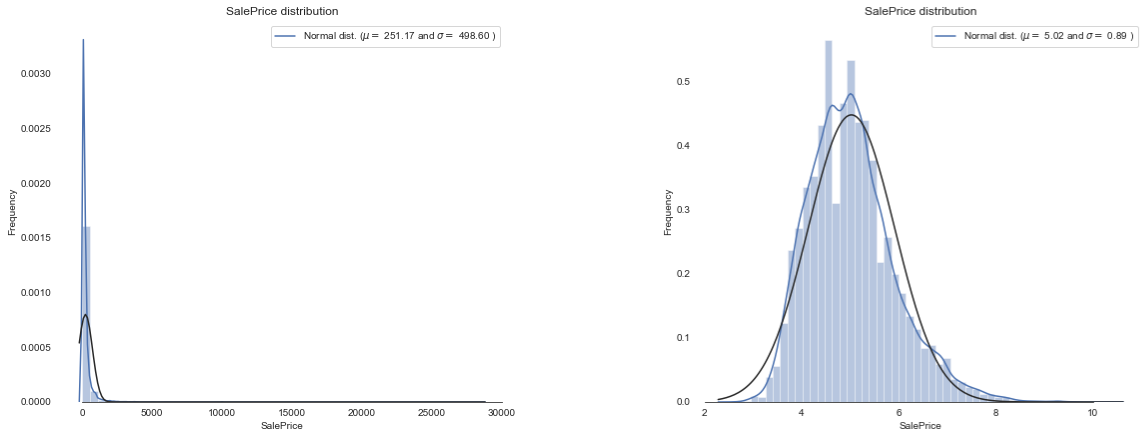
As seen from Figure , the transformed features exhibit a relatively more "normal" distribution, as compared to what they were before the transformation.

#### 5.3.2 Log1p Transformation

For the response variable, `price`, as we know it is extremely right-skewed, we have conducted a `log1p` transformation. Figure 5 shows two plots of the distribution of `price`, before and after the `log1p` transformation. Evidently from Figure 5b, the transformed variable follows a normal distribution more closely than that of Figure 5a. We are now ready to fit the data into our machine learning models.

## 6 Modelling

In order to build a predictive model for the purposes of predicting Airbnb rental prices of an Airbnb listing, five main classes of models were deployed: Linear Models, Tree-Based Models, Neural Network




(a) Original Distribution

(b) `log1p`-Transformed Distribution

Figure 5: Distribution of `price`

Model, Generalised Additive Model and Model Stack. Given the ease of interpretability and it being the foundation of any regression problem, the Linear Models are also our benchmark models.

The processed `listings` dataset is randomly split into training and testing sets for the model building and assessment of accuracy. The split ratio is 80% for training and 20% for testing.

## 6.1 Model 1: Linear Models (Benchmark Model)

Three models are considered for linear regression model - linear regression without penalty, ridge, and LASSO.

### 6.1.1 Linear Regression

Figure 15 shows the estimated coefficients from a fitted linear regression model. From the graph, it is evident that the number of bedrooms and the maximum capacity of the listing (from the `accommodates` column) have a stronger positive relationship with the rental price than other predictors. This contextually can be expected, as a higher number of bedrooms suggesting a larger property, of which a higher rental price can be expected. In comparison, the number of shared room listings a host has, has a significant, linear and negative impact to the rental price than other predictors.

As this model is simple and has high interpretability, it is used as the benchmark model to evaluate against other machine learning models.

### 6.1.2 Ridge Linear Model

Ridge linear model is based on the linear regression, but with a penalty term on the complexity of the model to avoid overfitting.

The training data is normalised before fitted into the model. The data is fitted with 10-fold cross validation. Figure 16 lists the 20 largest coefficients in the ridge model. The ridge regression has no variable selection feature, as it penalises complexity of the model by reducing the magnitude of the coefficients without ever reaching 0. The order of the 20 largest coefficients in absolute value is the same as the coefficients produced from the linear model.

### 6.1.3 LASSO Model

Similar to Ridge, LASSO linear model is based on the linear regression model, but with a penalty term on the complexity of the model. The difference between ridge and LASSO is that LASSO have an in-built variable selection feature. Depending on how large $\alpha$ is, it would force the coefficients of predictors to be 0, in order to reduce the complexity of the model.

The data is standardised before fitting into the model. The training data is standardised using its mean and standard deviation. As the purpose of the test data is an unseen dataset by the model to calculate the prediction error, it is standardised using the mean and standard deviation of the training data.

The training data is fitted with 10-fold cross validation. Figure 17 shows the 20 largest estimated coefficients in absolute value. The order of the magnitude of the coefficients are slightly different from ridge and linear regression without the penalty term. For example, in ridge and linear regression, predictor bedroom had the most positive significance over rental price, and the predictor of the number of shared room listings of the host has the most negative impact over price. However, in the LASSO model, predictor accommodates have a larger value of coefficient than bedroom, and the predictor of the number of private rooms has a more negative significance over the number of shared rooms.

The LASSO model removes 9 predictors to reduce the complexity of the model. These predictors are: `host_acceptance_rate`, `minimum_nights`, `review_scores_communication`, `minimum_nights_avg_ntm`, `calculated_host_listings_count_entire_homes`, `availability_60`, `availability_90`, `maximum_nights`, `maximum_nights_avg_ntm`.

### 6.1.4 Prediction Accuracy of Linear Models

For each of the fitted models, predictions are made using the testing set predictors. The result is compared to the rental price in the test set and the prediction accuracy is calculated, using various

metrics as shown in Table 4.

| Model | Test RMSE | Test $R^2$ | Test MAE | Test RMSE on `Price` |
|---|---|---|---|---|
| Linear | 0.4957 | 0.6899 | 0.3678 | 386.5119 |
| Ridge | 0.4993 | 0.6854 | 0.3711 | 392.5573 |
| LASSO | 0.4959 | 0.6897 | 0.3678 | 387.0839 |

Table 4: Prediction Accuracy of Linear Models

Although linear model is the simplest model in terms of formula and hyper-parameter tuning, it has the best performance out of the linear models.

## 6.2 Model 2: Tree-Based Models

Three tree-based models are adopted in this subsection: LightGBM, XGBoost and Random Forests models.

### 6.2.1 LightGBM, XGBoost & Random-Forest

The three models are often considered as classification methods. When those used for regression, the predictions we get are discrete because those models will assign some entries with similar feature valves the same prediction value. LightGBM and XGBoost are both based on the method of 'Gradient Boosting Decision Tree', an ensemble method that perform regression or classification by combining the outputs from individual trees. The gradient of the loss function with respect to the output of the previous model will be added in the current model's calculation, and the current objective is to lower the loss compared to the previous model alone (Dhlingra, 2020). From the documentation, the difference between LightGBM and XGBoost is that the latter employs histogram subtraction to accelerate the traversal. Besides, random forest model also uses bagging method to combine all trees' outputs. Due to such features of tree-based models, they perform well in large dataset and they are insensitive to monotone transformations (Hastie et al., 2001). This project's data fits some description of dataset which trees perform well in. Hyperparameter tuning is also applied to detect the best values for hyperparameters as shown in Figure 6, such as the learning-rate in XGBoost. After the process of tuning and fitting the models, they are fitted using the test features to compare their prediction accuracy.

| Model | Test RMSE | Test $R^2$ | Test MAE | Test RMSE on `Price` |
|---|---|---|---|---|
| LightGBM | 0.4447 | 0.7504 | 0.3227 | 345.3431 |
| XGBoost | 0.4410 | 0.7546 | 0.3181 | 334.4902 |
| Random Forests | 0.4618 | 0.7308 | 0.3338 | 349.2299 |

Table 5: Prediction Accuracy of Tree-Based Models

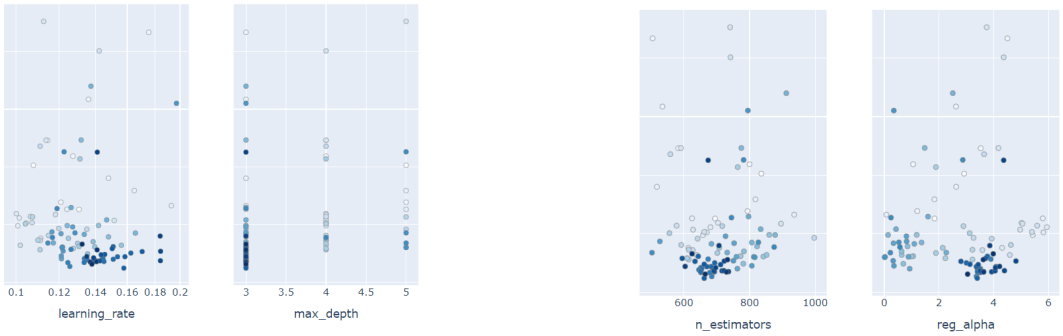



Figure 6: Some Plots for Hyperparameter Optimization Outcomes

Table 5 summarises the results of the prediction of tree-based models on the test set. XGBoost performed the best, with a test rmse of 0.4410 on the `log1p`-transformed response, `price`.

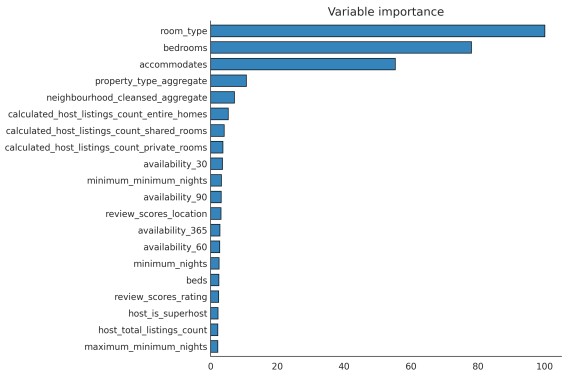
### 6.2.2 Feature Importance Analysis

One of the many advantages of tree-based models include their ability to provide an anlysis of feature importance, which contributes to its ability to be interpreted. Figure 7 shows 3 graphs of features ordered by their importance, based on the three models we have just built.

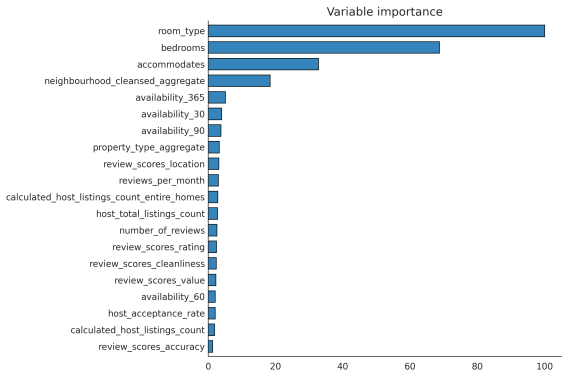
Figure 7a for XGBoost and Figure 7b for Random Forests agrees that some of the highly important features include `room_type`, `bedrooms` and `accommodates`. Although LightGBM displays somewhat different results, it also agrees with the other two models that `accommodates` and `bedrooms` are some of the more important features. This might be unsurprising, given that the number of bedrooms and the number of people an Airbnb listing can accommodate are some of the primary concerns when consumers, such as travellers in groups, are searching for a place on the Airbnb website. The ability of the tree-based models to provide such analysis of relative importance of features prove useful during Data Mining in Section 8.

## 6.3 Model 3: Spline and Generalised Additive Model

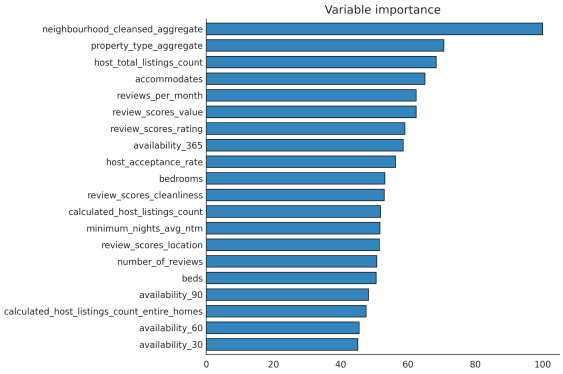
Spline regression for one or two features has less interpretablity and poor performance in this project with 42 selected features. Generalized additive models, which are more automatic flexible statistical methods, are used here to identify and characterize nonlinear regression effects (Hastie et al., 2001). According to the visualisation plots of the relationship between the response and each of the inputs, we set the spline term, linear term and a tensor term. Here, we use optuna library tool for the optimization of parameters' coefficients. After trials and errors, we pick the best combination among different subsets of all features. And finally, we just picked the combination of 'accommodates,bedrooms,beds' as interaction effect and 'host-acceptance-rate' as spline term.




(a) XGBoost




(b) Random Forests




(c) LightGBM

Figure 7: Feature Importance from the 3 Models

| Model | Test RMSE | Test $R^2$ | Test MAE | Test RMSE on `Price` |
|---|---|---|---|---|
| GAM | 0.4773 | 0.7126 | 0.3520 | 361.1443 |

Table 6: Prediction Accuracy of Generalised Additive Model

Table 6 shows the performance of our generalised additive model, measured on different metrics. Limit to inadequate hyper-parameters tuning, this model is worse than all tree-based models.

## 6.4 Model 4: Neural Network Model

In this subsection we consider the applicability of a neural network model for regression. Specifically, we will be building a deep feedforward network model through the deep learning using `Pytorch`. There are many advantages of building a feedforward neural network model, including the fact that the handling and processing of non-linear data can be done easily (Edgell, 2021), hence uncovering relationships that "simpler" models might have missed. However, notwithstanding its powerful application of representation learning, neural network models often come with low interpretability and high computational cost.

In building our neural network model, we have chosen to include 2 hidden layers, each with 128 units. The activation functions have been standardised to using `ReLU` throughout. The model will be trained using a loss function of Mean Squared Error and a learning rate of 0.001, for an epochs of 100. Note that the above combination is the final outcome of repeated hyperparameter tuning of the hidden layers, number of units, activation functions, loss functions and learning rate.

| Model | Test RMSE | Test $R^2$ | Test MAE | Test RMSE on `Price` |
|---|---|---|---|---|
| Neural Network | 0.5043 | 0.6791 | 0.3836 | 380.1261 |

Table 7: Prediction Accuracy of Neural Network Model

Table 7 shows the performance of our neural network model, measured on different metrics. While the performance is slightly better than our benchmark model, Linear Regression, the difference is however very marginal.

### 6.5 Model 5: Model Stack

Finally, we have built a Model Stack based on some of the models we have built previously. Model stacking is an application of ensemble learning where we combine predictions from multiple learning algorithms as the final model. The ability of ensemble learning methods to generalise can sometimes exceed the performance of individual models. The stack consists of 4 models: Linear Regression, XGBoost, Neural Network and Generalised Additive Model.

| Model | Test RMSE | Test $R^2$ | Test MAE | Test RMSE on `Price` |
|---|---|---|---|---|
| Stacking Model | 0.4449 | 0.7502 | 0.3223 | 339.2054 |

Table 8: Prediction Accuracy of Model Stack

Table 8 provides the prediction results of the Model Stack on the test set, measured on different metrics. While the performance of Model Stack was decent, it was not better than the best-performing model, XGBoost, which was one of the models included in the stacking of models.

## 7 Evaluation & Analysis

Table 9 conveniently summarised the predictive performance of all the models on the test set, measured on different metrics. Overall, tree-based model using XGBoost has the best performance with a test RMSE of 0.4410, or $334.49 in terms of actual dollar values. Surprisingly, the Neural Network model did not perform as well as we thought it would be. This could be perhaps due to undesirable feature engineering, or simply overfitting the model than we should have. While the Model Stack

has a very close performance to that of XGBoost, it should be assumed that such behaviour resulted from the fact that bulk of the weights are attributed to the XGBoost with minimal weights on the remaining of the models in the model stack.

| Model | Test RMSE | Test $R^2$ | Test MAE | Test RMSE on `Price` |
|---|---|---|---|---|
| linear | 0.4957 | 0.6899 | 0.3678 | 386.5119 |
| ridge | 0.4993 | 0.6854 | 0.3711 | 392.5573 |
| lasso | 0.4959 | 0.6897 | 0.3678 | 387.0839 |
| LightGBM | 0.4447 | 0.7504 | 0.3227 | 345.3431 |
| XGBoost | 0.4410 | 0.7546 | 0.3181 | 334.4902 |
| Random Forests | 0.4618 | 0.7308 | 0.3338 | 349.2299 |
| Neural Network | 0.5043 | 0.6791 | 0.3836 | 380.1261 |
| GAM | 0.4773 | 0.7126 | 0.3520 | 361.1443 |
| Stacking Model | 0.4429 | 0.7525 | 0.3198 | 335.7857 |

Table 9: Model Performance Summary

In conclusion, XGBoost is our preferred model due to its relative good interpretability. While Model Stack is a close fight, the fact that it still underperforms to XGBoost and the amount of computation resources required simply makes it not attractive as an option for our predictive model. Similarly, the Neural Network model, due to its low interpretability and relatively poor results, should not be considered.

# 8 Data Mining for Business Decisions

As investors in the real estate industry, hosts of the Airbnb listings might be interested in finding out what the best hosts are doing in order to improve their property rental income. The aim of this section is to generate three insights from the process of our predictive modelling, or any other further analysis of the `listings` dataset, so we can provide a detailed explanation and recommendation to improve income revenue from the leasing of properties through Airbnb.

## 8.1 Insight from Predictive Modelling

Firstly, we draw on the results that we have already seen repeatedly throughout our modelling. In particular, the linear models in Section 6.1 and tree-based models in Section 6.2. Figures 15, 16 and 17 outlined the estimated coefficients of each predictors. We can immediately identify that, some of the top features that affect the rental price include `neighbouthood_cleansed_aggregate`, `property_type_aggregate`, `bedrooms` and `accommodates`. This is unsurprising as we have previously mentioned, locality, property type, number of bedrooms and number of people that can be accommodated are some of the common features a consumer would almost certainly take into account when renting through Airbnb.

For the real estate investors hosts, this result however should convince them that these features are some of the top priorities to consider when looking to invest in a property for rental. Investing in a big property in a prominent locality is likely to maximise rental price and hence rental income. However, it should be noted that such a "prime" estate usually comes at the cost of heavy tax, high operating and maintenance costs. In addition, a real estate investment has low liquidity and is subject to capital gains or losses. While there is no doubt that an investment in large property at a prime location will lead to high rental prices, the ability to attract tenants and generate profitable passive income will also depend on other factors.

## 8.2 Insights from Further Analysis

### 8.2.1 LASSO on `price` using Host Behaviours

Next, we consider variables specifically concerning host behaviours, which include: `host_response_time`, `host_response_rate`, `host_acceptance_rate`, `host_total_listings_count`, `host_has_profile_pic`, `host_identity_verified`, `host_is_superhost`.

A LASSO linear regression model was fit using host attributes as features and `price` as response. The training data is fitted to a 10-fold cross validation LASSO model. The hyperparameter (penalty term) is optimised by running an algorithm that tests a series of incremental $\alpha$ starting from $1e^{-13}$ with an increment of 0.00001. A validation set is used to evaluate the prediction accuracy of each model produced on a different $\alpha$ value. Figure 18 shows the change of RMSE on the validation dataset with the increase of $\alpha$. A local minimum point is evident. When $\alpha = 0.0089$, it provides the smallest RMSE. Figure 19 is a graph of the estimated coefficients after fitting the LASSO model with $\alpha = 0.0089$.

Figure 19 provided some useful insights on the relationship between host behaviours and price. The response time of the host has a significant positive impact on the rental price of the listing. This is expected contextually, as whether a host replies a message within an hour, few hours or days, would leave a positive or negative impression to the customers, as well as impact their willingness to pay potentially high price for an accommodation. The coefficients of whether the host has a profile picture and their response rate does not impact the rental price. Thus, the hosts should focus less on these two aspects, but try to improve their response time, in order to increase the rental price of the listings.

### 8.2.2 Superhost

We are also aware from 19 that the variable `host_is_superhost` is also positively associated with `price`. A superhost title is 'earned' through positive customer experience such as receiving positive reviews and being responsive. Intuitively, a listing by a superhost would be more well-received by

customers and these hosts are more likely to earn a higher rental income through the title of a superhost.

Here, we decided to make `host_is_superhost` the response variable and run a classification model using LightGBM. Figure 21 shows the list of features ordered by their importance in predicting a superhost status. Apart from the rental price, some of the common features in determining a superhost is the number of reviews per month, the host acceptance rate, number of reviews. In fact, out of the listed important features, reviews take up quite a number of them.

As such, the reviews from customers can heavily impact whether a host can earn a superhost title, and consequently the income revenue of the host. It might be in the best interest of hosts and property investors to come up with a strategy to encourage customers to leave (positive) reviews, such as sending a thank you note and asking for reviews as a personal touch.

# 9 Conclusion

In conclusion, we found that tree-based models, inparticular XGBoost had the best prediction accuracy in predicting the price of an Airbnb listing in Sydney, based on the provided dataset. Some common features determining the price of a listing include number of bedrooms, number of people that can be accommodated and the locality of the listing. Hosts and property investors should consider these factors when looking to invest in a potential rental property. In addition, they should be aware of the common traits a good host have. Some strategies should be considered to ensure that the host do not response too late to a potential tenant, and that there customers are consistently providing positive reviews on the Airbnb platform. That way, it will maximise the web patronage as well as the rental price so hosts and property investors can maximise their rental income.

# Appendices

## A Details of the Columns Converted

| Column | Before Conversion | After Conversion |
|---|---|---|
| last_scraped | object | datetime64[ns] |
| host_since | object | datetime64[ns] |
| first_review | object | datetime64[ns] |
| last_review | object | datetime64[ns] |
| calendar_last_scraped | object | datetime64[ns] |
| host_response_time | object | category |
| room_type | object | category |
| host_is_superhost | object | category |
| host_has_profile_pic | object | category |
| host_identity_verified | object | category |
| has_availability | object | category |
| instant_bookable | object | category |
| host_response_rate | object | float64 |
| host_acceptance_rate | object | float64 |
| id | int64 | object |
| scrape_id | int64 | object |
| host_id | int64 | object |
| price | object | float64 |

Table 10: Conversion of `dtype` of Variables

# B Descriptive Statistics of Numerical Data

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| host_response_rate | 7884.0 | 0.9046 | 0.2425 | 0.0 | 0.97 | 1.0 | 1.0 | 1.0 |
| host_acceptance_rate | 8846.0 | 0.7756 | 0.3159 | 0.0 | 0.68 | 0.94 | 1.0 | 1.0 |
| host_listings_count | 20877.0 | 10.9494 | 40.9933 | 0.0 | 1.0 | 1.0 | 3.0 | 457.0 |
| host_total_listings_count | 20877.0 | 10.9494 | 40.9933 | 0.0 | 1.0 | 1.0 | 3.0 | 457.0 |
| accommodates | 20880.0 | 3.3833 | 2.1881 | 1.0 | 2.0 | 2.0 | 4.0 | 16.0 |
| bedrooms | 19444.0 | 1.7043 | 1.0361 | 1.0 | 1.0 | 1.0 | 2.0 | 18.0 |
| beds | 19976.0 | 2.0398 | 1.5383 | 1.0 | 1.0 | 1.0 | 3.0 | 39.0 |
| price | 20880.0 | 251.1662 | 498.5989 | 13.0 | 80.0 | 140.0 | 250.0 | 28613.0 |
| minimum_nights | 20880.0 | 62.1895 | 52.7393 | 1.0 | 4.0 | 90.0 | 90.0 | 1125.0 |
| maximum_nights | 20880.0 | 905.051 | 407.0761 | 1.0 | 1125.0 | 1125.0 | 1125.0 | 1500.0 |
| minimum_minimum_nights | 20879.0 | 61.5631 | 52.9267 | 1.0 | 3.0 | 90.0 | 90.0 | 1125.0 |
| maximum_minimum_nights | 20879.0 | 62.5667 | 52.263 | 1.0 | 5.0 | 90.0 | 90.0 | 1125.0 |
| minimum_maximum_nights | 20879.0 | 103815.5177 | 14861916.7367 | 1.0 | 1125.0 | 1125.0 | 1125.0 | 2147483647.0 |
| maximum_maximum_nights | 20879.0 | 1646629.6788 | 59426307.4844 | 1.0 | 1125.0 | 1125.0 | 1125.0 | 2147483647.0 |
| minimum_nights_avg_ntm | 20879.0 | 62.0717 | 52.5958 | 1.0 | 4.5 | 90.0 | 90.0 | 1125.0 |
| maximum_nights_avg_ntm | 20879.0 | 1501654.7747 | 54301668.6437 | 1.0 | 1125.0 | 1125.0 | 1125.0 | 2147483647.0 |
| availability_30 | 20880.0 | 7.0628 | 10.7423 | 0.0 | 0.0 | 0.0 | 12.0 | 30.0 |
| availability_60 | 20880.0 | 16.9735 | 22.7144 | 0.0 | 0.0 | 0.0 | 35.0 | 60.0 |
| availability_90 | 20880.0 | 28.1032 | 35.2623 | 0.0 | 0.0 | 0.0 | 62.0 | 90.0 |
| availability_365 | 20880.0 | 102.0672 | 134.2559 | 0.0 | 0.0 | 7.0 | 180.0 | 365.0 |
| number_of_reviews | 20880.0 | 17.8957 | 42.9944 | 0.0 | 0.0 | 2.0 | 13.0 | 881.0 |
| number_of_reviews_ltm | 20880.0 | 2.8804 | 9.4075 | 0.0 | 0.0 | 0.0 | 1.0 | 565.0 |
| number_of_reviews_l30d | 20880.0 | 0.2988 | 1.06 | 0.0 | 0.0 | 0.0 | 0.0 | 30.0 |
| review_scores_rating | 15071.0 | 4.471 | 1.05 | 0.0 | 4.5 | 4.81 | 5.0 | 5.0 |
| review_scores_accuracy | 14459.0 | 4.7304 | 0.521 | 0.0 | 4.68 | 4.91 | 5.0 | 5.0 |
| review_scores_cleanliness | 14469.0 | 4.589 | 0.6271 | 0.0 | 4.5 | 4.8 | 5.0 | 5.0 |
| review_scores_checkin | 14452.0 | 4.8241 | 0.4404 | 0.0 | 4.83 | 5.0 | 5.0 | 5.0 |
| review_scores_communication | 14469.0 | 4.8212 | 0.46 | 0.0 | 4.84 | 5.0 | 5.0 | 5.0 |
| review_scores_location | 14453.0 | 4.8165 | 0.3843 | 0.0 | 4.8 | 4.96 | 5.0 | 5.0 |
| review_scores_value | 14448.0 | 4.63 | 0.5387 | 0.0 | 4.5 | 4.77 | 5.0 | 5.0 |
| calculated_host_listings_count | 20880.0 | 8.0945 | 25.1916 | 1.0 | 1.0 | 1.0 | 3.0 | 197.0 |
| calculated_host_listings_count_entire_homes | 20880.0 | 6.719 | 24.1348 | 0.0 | 0.0 | 1.0 | 1.0 | 197.0 |
| calculated_host_listings_count_private_rooms | 20880.0 | 1.2747 | 7.4286 | 0.0 | 0.0 | 0.0 | 1.0 | 100.0 |
| calculated_host_listings_count_shared_rooms | 20880.0 | 0.0602 | 0.6455 | 0.0 | 0.0 | 0.0 | 0.0 | 17.0 |
| reviews_per_month | 15071.0 | 0.6415 | 1.0923 | 0.01 | 0.06 | 0.2 | 0.83 | 44.49 |

Table 11: Descriptive Statistics of Numerical Data

# C Scatter Plot of Numerical Variables

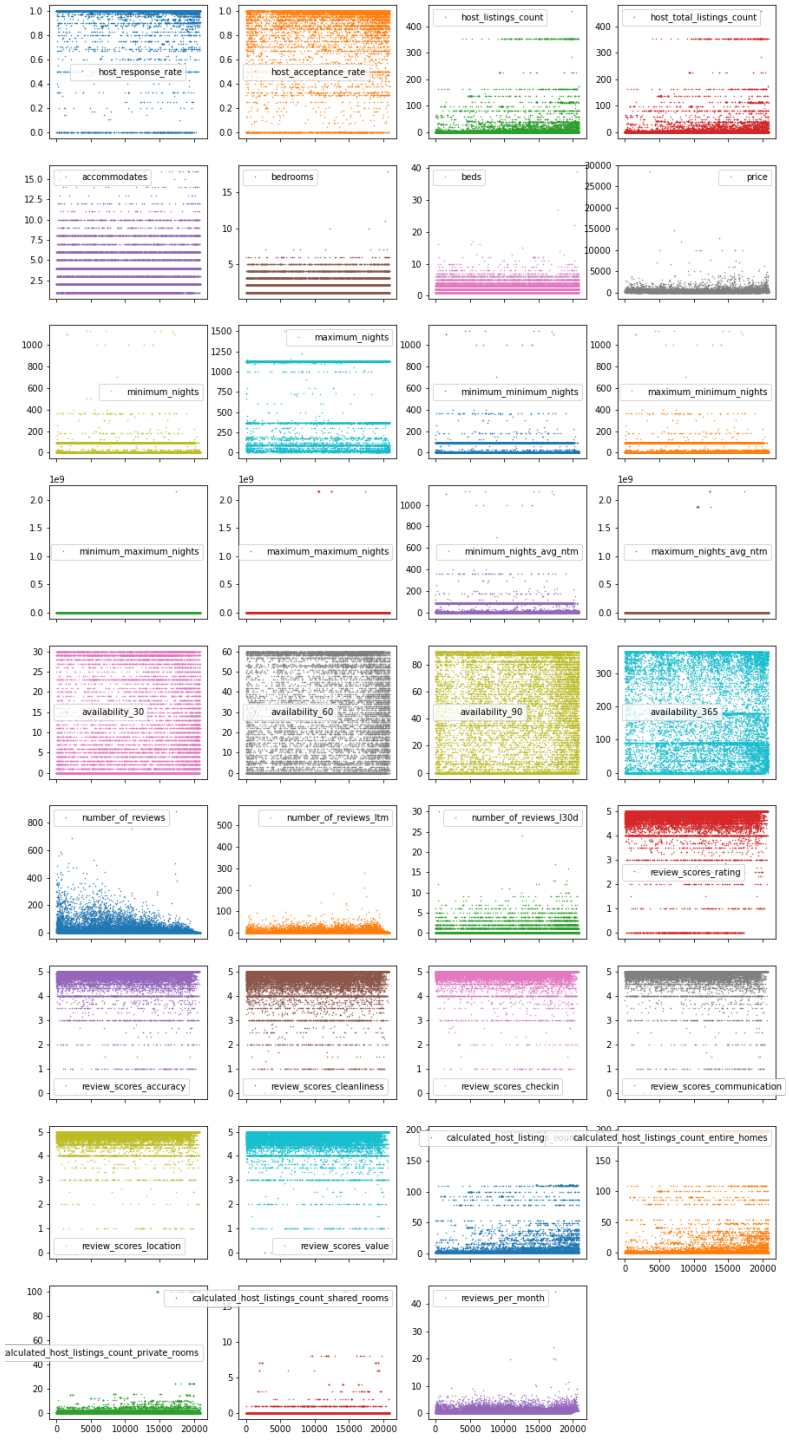



Figure 8: Scatter Plot of Numerical Variables

# D Box Plot of Numerical Variables


Figure 9: Box Plot of Numerical Variables

# E Histogram & Density Curve of Numerical Variables

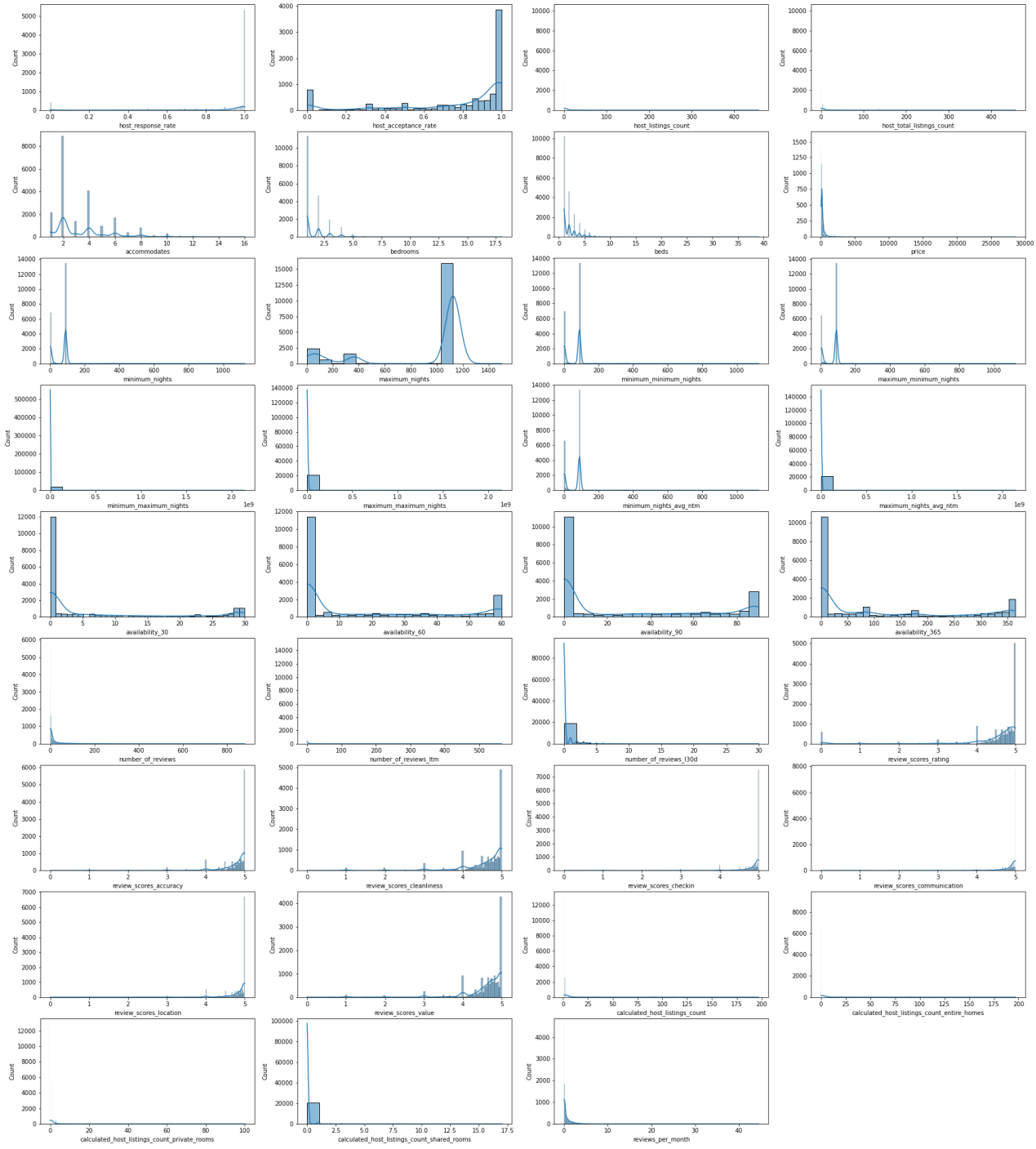



Figure 10: Histogram of Numerical Variables. The line indicates the Estimated Density Curve of the variable.

# F Violin Plot of Categorical Variables

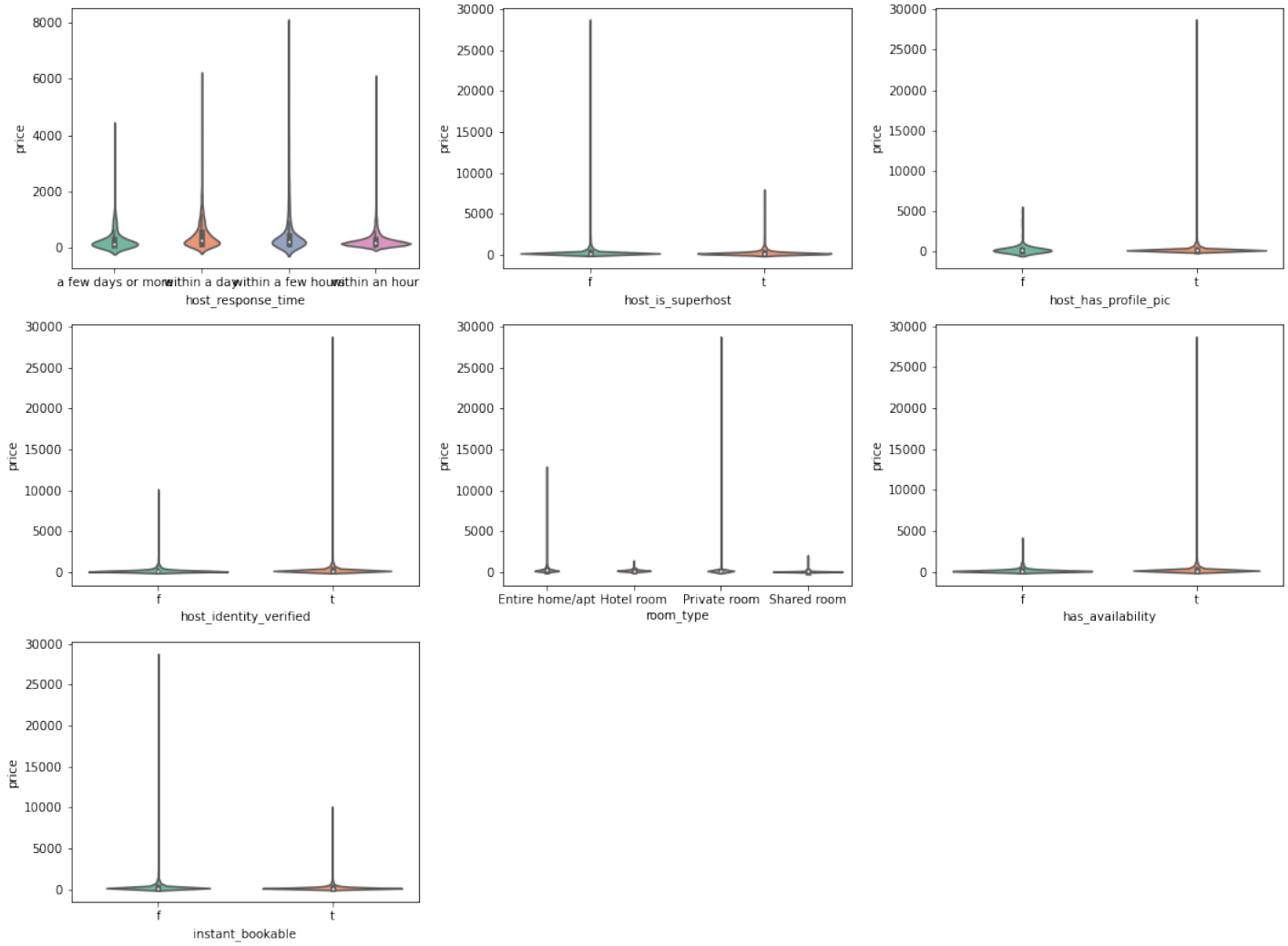



Figure 11: Violin Plot of Categorical Variables

# G Count of Neighbourhood

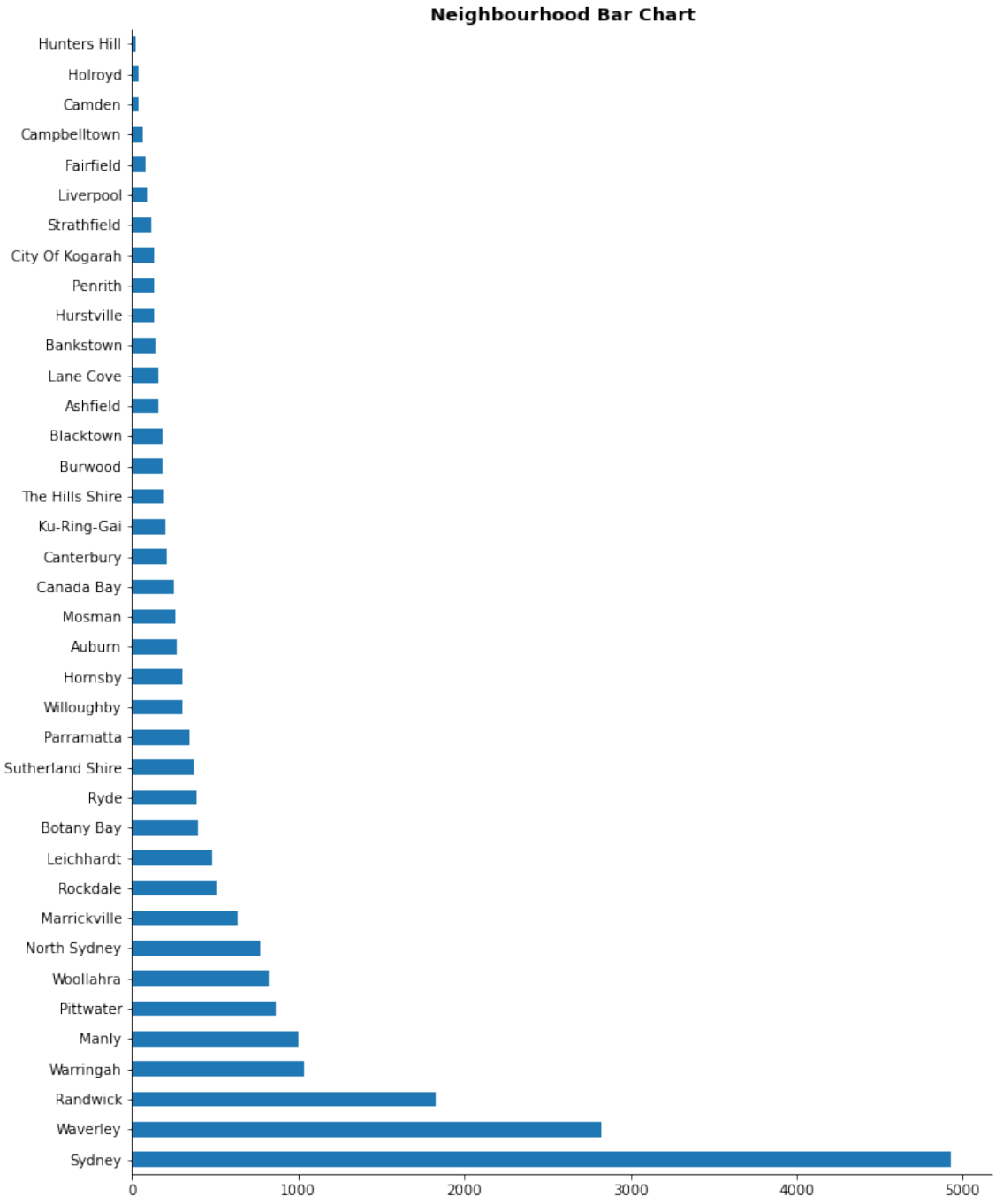



Figure 12: Bar Graph of Count of Neighbourhoods

# H Kendall Correlation Matrix Heatmap

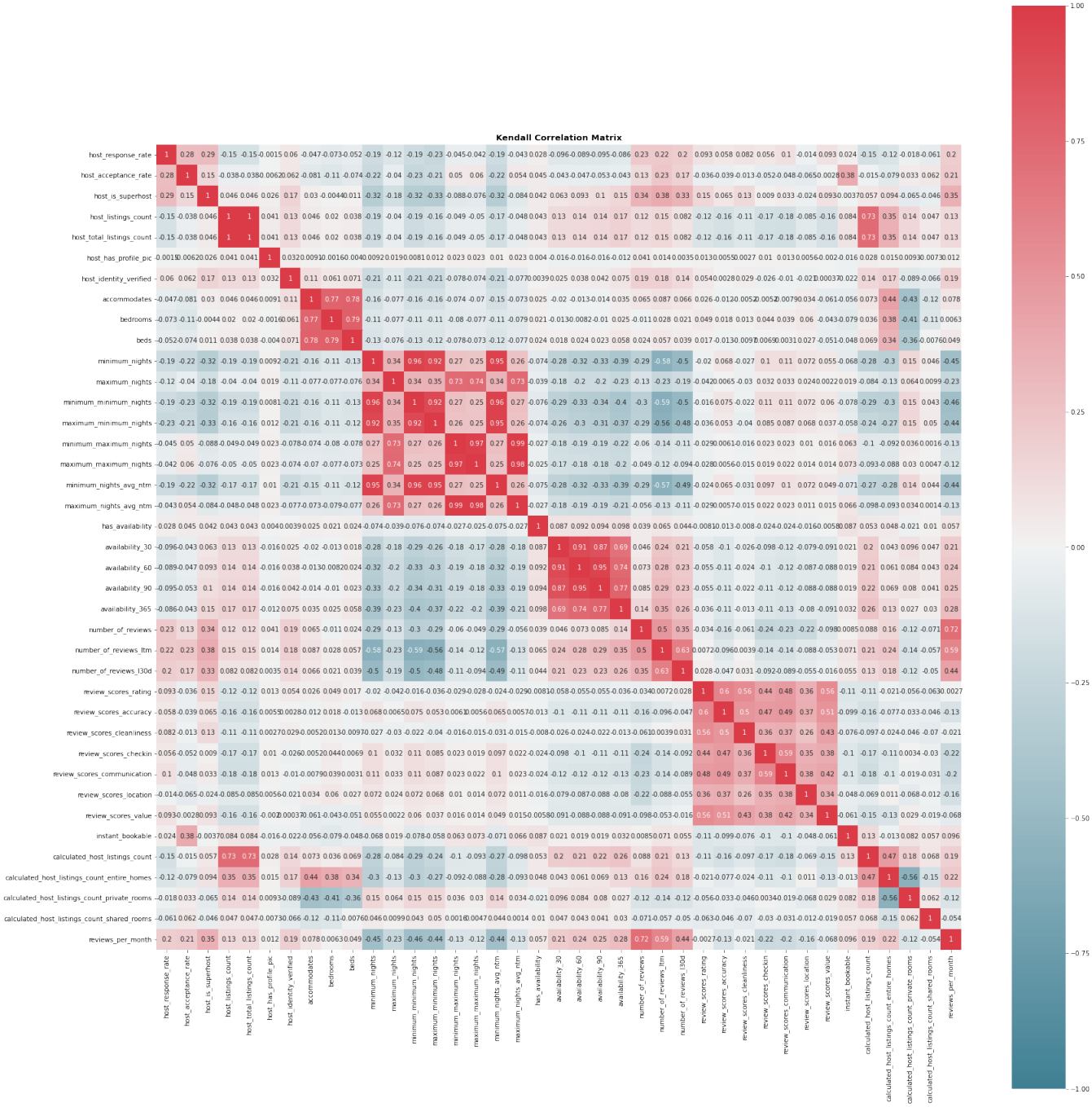



Figure 13: Heatmap of Kendall Correlation Matrix between Numerical Variables

# I Box Cox Transformation

The transformation rule for Box Cox is as follows:

$$y(\lambda) = \begin{cases} \frac{y^{\lambda}-1}{\lambda}, & \text{if } \lambda \neq 0; \\ \log y, & \text{if} \lambda = 0 \end{cases} \tag{1}$$

Where $\lambda$ varies from -5 to 5. All the values of $\lambda$ are considered, and the optimal value for the input data is selected. In the case of our project, the Box Cox transformation is performed on the features, $X$ instead of the response $y$.

# J Transformed Features

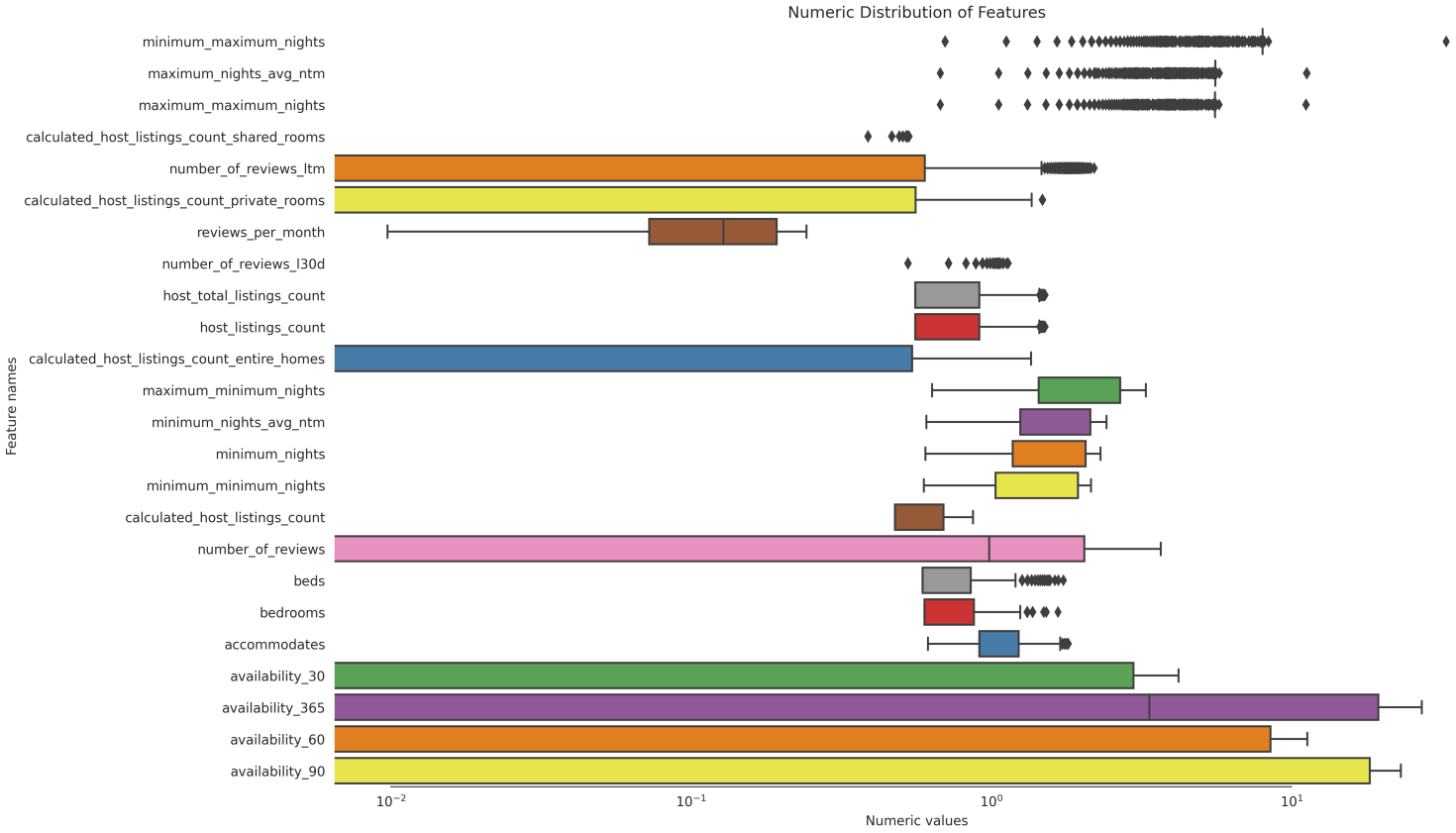



Figure 14: Linear Regression Coefficients

# K Linear Regression Coefficients

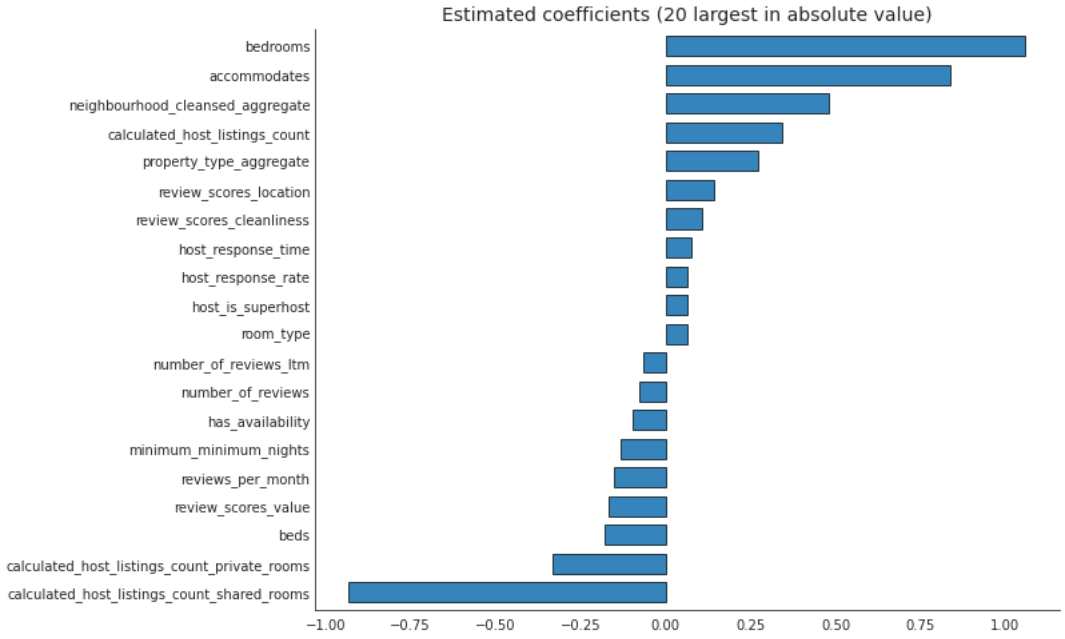



Figure 15: Linear Regression Coefficients

# L Ridge Regression Coefficients

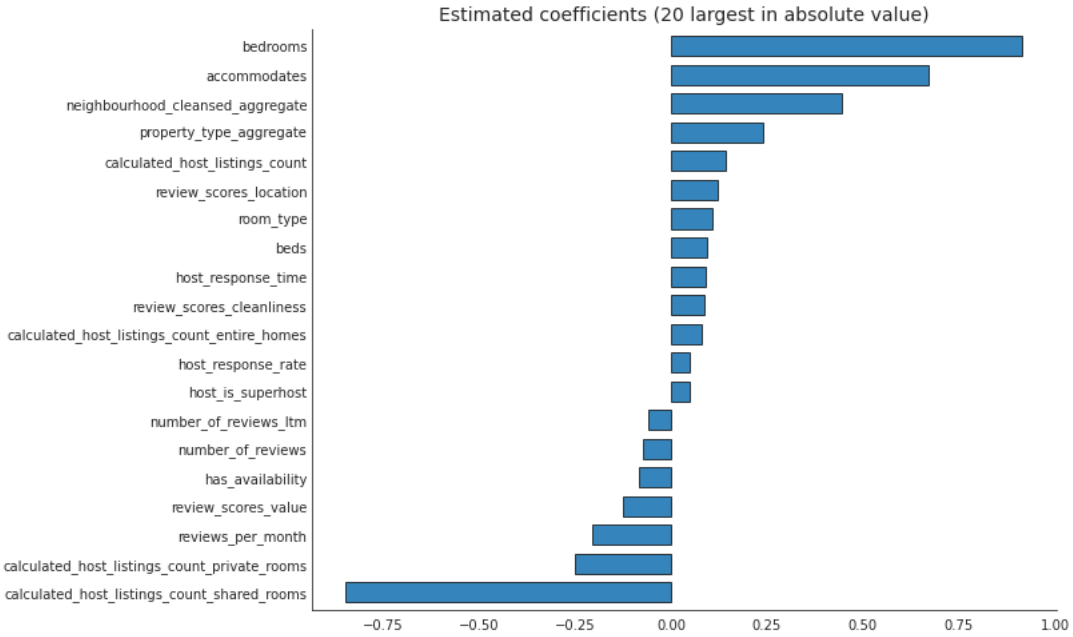



Figure 16: Ridge Regression Coefficients

# M LASSO Regression Coefficients

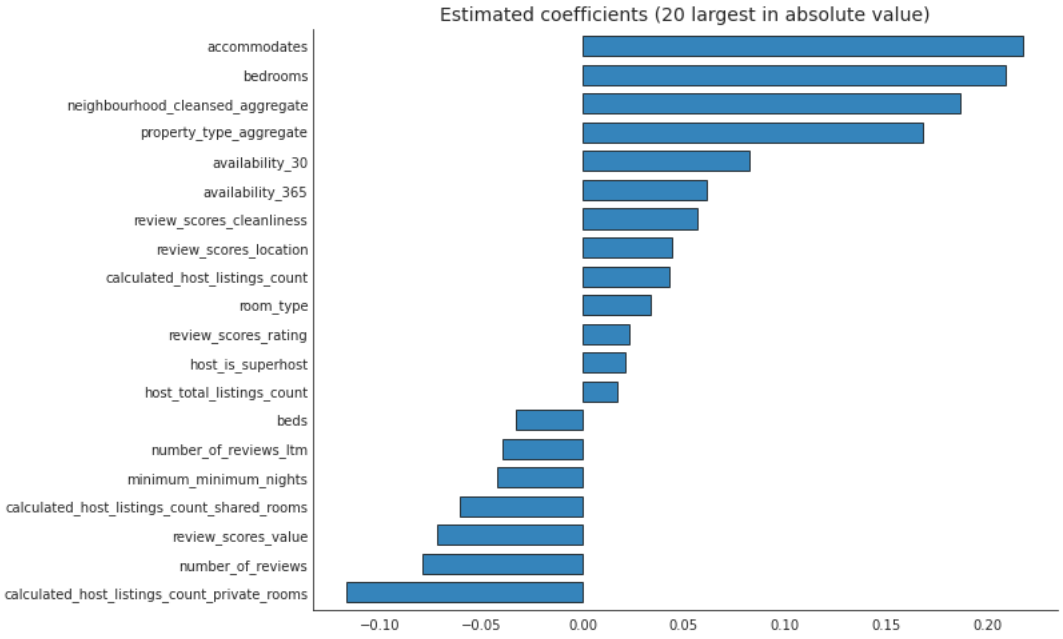



Figure 17: LASSO Regression Coefficients

# N RMSE of Validation Data with Different $\alpha$

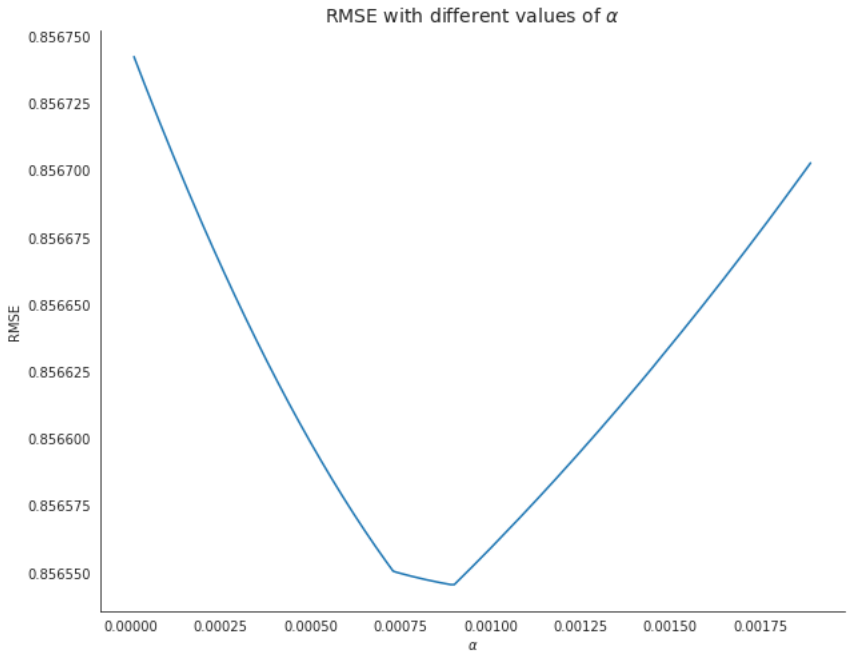



Figure 18: RMSE of Validation Data with Different $\alpha$

# O Estimated Coefficients from LASSO for Host Behaviours

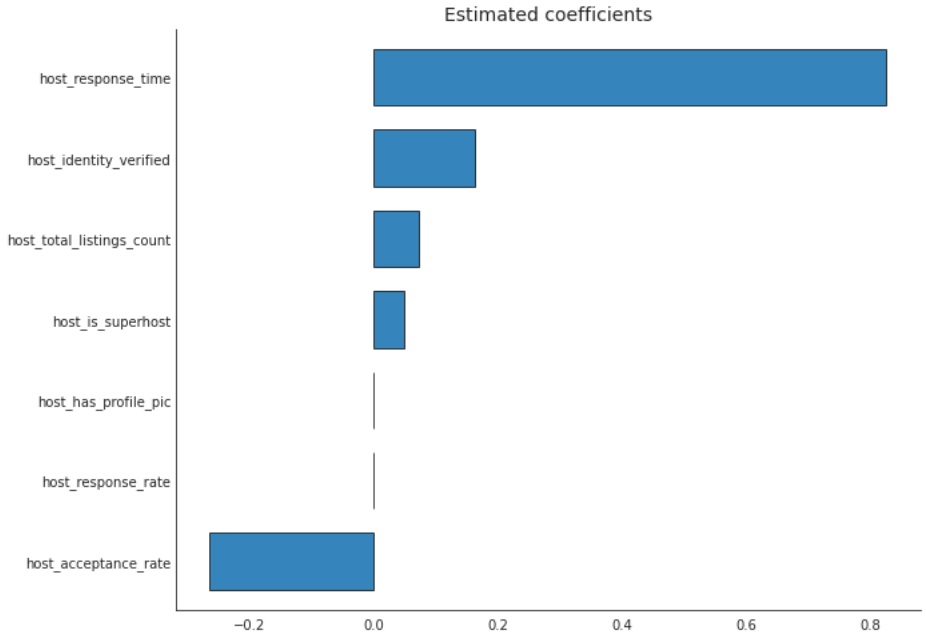



Figure 19: Estimated Coefficients of LASSO

# P Broad Geographical Analysis

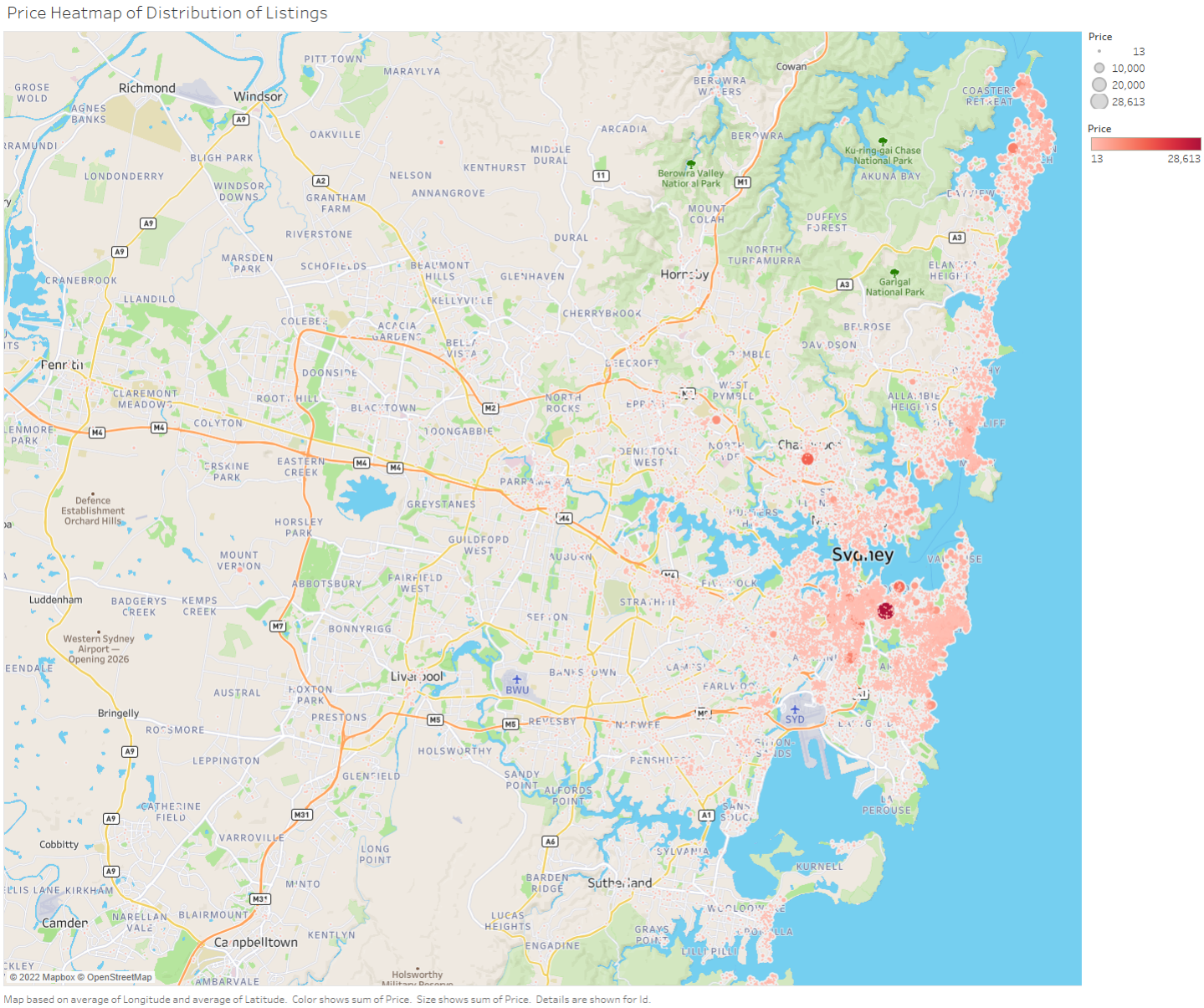



Figure 20: Price Heatmap of Listings in Sydney. The graph is generated using Tableau.

# Q Feature Importance for Superhost Classification

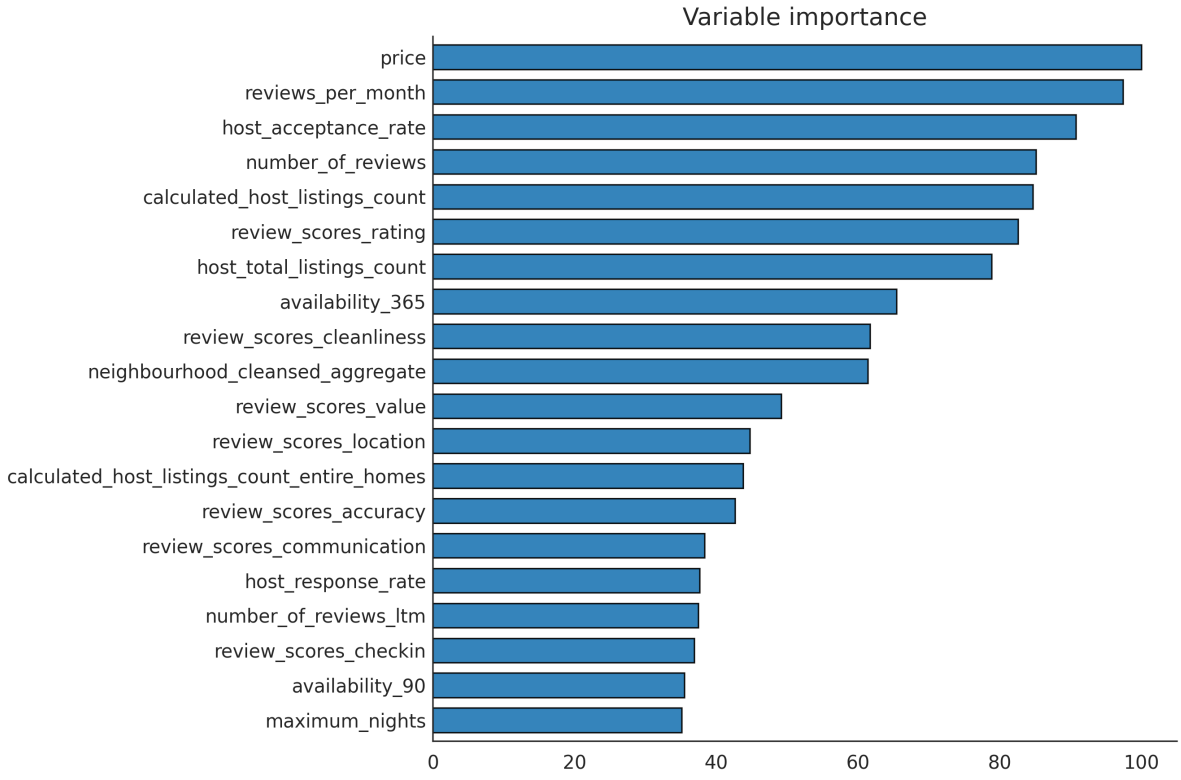



Figure 21: Feature Importance from LightGBM on Classification of Superhost

## References


AllTheRooms (2021). Airbnb statistics: Airbnb stats for your market [2021]. https://www.alltherooms.com/analytics/airbnb-statistics/.

Dhlingra, C. (2020). A visual guide to gradient boosted trees (xgboost). https://towardsdatascience.com/a-visual-guide-to-gradient-boosted-trees-8d9ed578b33.

Domingos, P. (2012). A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87.

Edgell, A. (2021). Feedforward neural networks. https://www.datasciencecentral.com/feedforward-neural-networks/:∼:text=Advantages%20of%20Feedforward%20Neural%20Networkstext=The%-20handling%20and%20processing%20of,is%20alleviated%20in%20neural%20networks.

Frost, J. (2017). Multicollinearity in regression analysis: Problems, detection, and solutions. https://statisticsbyjim.com/regression/multicollinearity-in-regression-analysis/.

GeeksForGeeks (2021). Categorical encoding with catboost encoder. https://www.geeksforgeeks.org/categorical-encoding-with-catboost-encoder/.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

IBM (2021). Crisp-dm help overview. https://www.ibm.com/docs/en/spss-modeler/SaaS?topic=dm-crisp-help-overview.

Lalonde, S. (2012). Transforming variables for normality and linearity – when, how, why and why not's. *SAS Conference Proceedings NESUG*.

Lewinson, E. (2019). Violin plots explained. https://towardsdatascience.com/violin-plots-explained-fb1d115e023d.

Magiya, J. (2019). Kendall rank correlation explained. https://towardsdatascience.com/kendall-rank-correlation-explained-dee01d99c535:∼:text=Kendall%20rank%20correlation%20(non%2Dparametr-ic)%20is%20an%20alternative%20to,and%20has%20many%20tied%20ranks.

Saxena, S. (2020). Here's all you need to know about encoding categorical data (with python code). https://www.analyticsvidhya.com/blog/2020/08/types-of-categorical-data-encoding/.