QBUS6810 Statistical Learning and Data Mining
Group Assignment

Group 160: 480140973, 510488543, 520600843, 480038647, 500135329

Semester 2, 2022

Contents

1. Introduction

Airbnb is a global platform that runs an online marketplace for renting and leasing short-term lodging. The goal of this report is to predict the price per night of listings for properties in Sydney, Australia. The first part is data processing. The method of data cleaning and processing was in detail described. The second part is exploratory data analysis. All the variables were separated into different categories, they are property-related variables, location-related variables, and host characteristics. Feature selection was introduced in the third part. We made assumptions for linear regression and outliers, set dummy variables and interaction terms. Next, 5 models were introduced in the fourth part Methodology, they are Ordinary Least Square model (OLS), Ridge/Lasso, Single Regression Tree, Gradient Boosting, and Model Average. The final part is validation and comparisons. To improve the accuracy of the predictions, we used our established models as the base model for model stacking and introduced the Random Forest (RF) model to improve the performance of the model combination. The result of the model stacking resulted in a 2-unit decrease in RMSE compared with the optimal base model.

2. Data Processing

2.1 Data Description

The dataset includes 4000 rows and 36 columns. Except “id”, there are 14 categorical variables and 21 numerical variables.

Table 1: Missing proportion of different variables

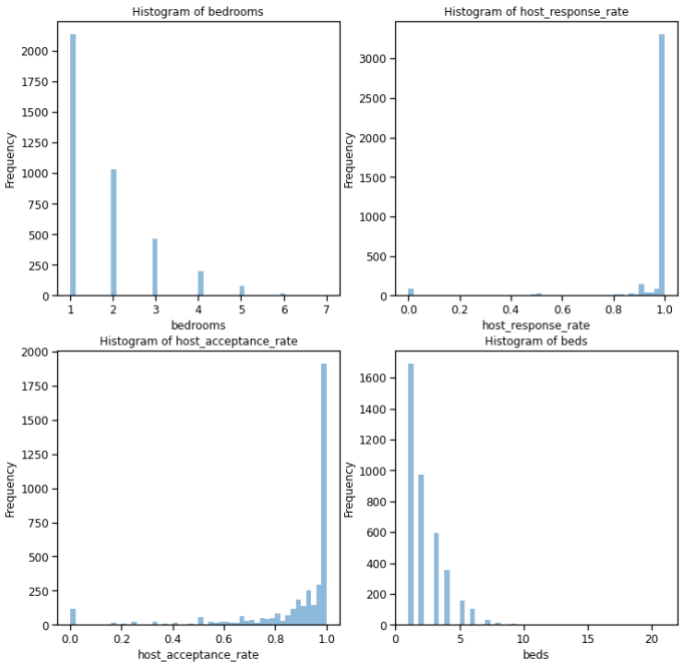
Variables	Missing percentage	Variables	Missing percentage
host_neighbourhood	39.0	host_response_rate	3.2
neighborhood_overview	10.2	host_acceptance_rate	2.2
neighbourhood	10.1	beds	0.5
bedrooms	6.8	description	0.1
host_response_time	3.2	Rest 27 variables	0

The data quality is good as there is no duplicate and missing data only takes a small proportion. Table 1 shows that there are only 9 variables that have missing values. Our response variable “price” is complete and is stored in AUD. However, the accuracy of the data type is low. We noticed that some numerical values which ought to be percentages were mistakenly stored as strings, including “host_response_rate” and “host_acceptance_rate”. The DateTime variable “host_since” was also wrongly stored. In terms of the usage of the dataset, we can gain an understanding of the selling points of each listing from descriptive variables such as “neighborhood_overview” as well as its other features from variables like “beds”, “bedrooms” and “amenities”. We can also get the information of hosts, including how long the host responds and that rate at which a host accepts booking requests. Data for reviews tell how popular the property is.

2.2 Data Processing

We first dropped variables that are unrelated to “price” or the information contained can be explained by other variables , including “id”, “description”, “host_location”, “host_neighbourhood”, “neighbourhood”, “latitude”, “longitude”. We then corrected the wrong data types, by converting “host_since” to DateTime, and 'host_response_rate' and “host_acceptance_rate” to float.

Figure 1: Distribution of incomplete numerical variables



To decide the imputation method for missing values, we inspected distributions of 4 incomplete numerical variables. Since they are all missing completely at random and for each variable, the number of the mode is significant greater than that of other values, we believe that we can replace nulls with the mode. As missing values are most likely to be the mode, mode imputation in this case will not skew the histograms. Here, we treated "bedrooms" differently. Considering that sometimes a missing value can mean that the corresponding characteristic does not apply to that particular listing, it is reasonable to believe that missing values for "bedrooms" represent no bedroom, rather than lack of information. Thus, we filled nulls with 0.

Since values of "bedrooms" falls in only 7 categories, we tried both approaches, treating "bedrooms" as numerical and as categorical for modelling, and tested the model performance. For incomplete categorical variables, we filled nulls with "Unknown" to keep information of other columns. We then generated a new variable by extracting the year from "host_since". We applied the same data processing to the test set.

3. Exploratory Data Analysis

The analysis conducted within the EDA section is aimed at identifying interesting patterns and relationships in the datasets, especially those features that seem to influence the target variable "price".

At first glimpse, the data provided can be roughly summarized in 4 categories. The first group is composed of those variables that are somehow strictly related to the host himself, for instance, "host_since" or "host_is_superhost". Then we have data providing information regarding the location of the property. Next, we find those variables that summarize features of the dwelling and the last reviews given by previous guests. This first preliminary evidence can be used as an outline that can help to make sense of the data and grasp the meaning of the underlying patterns. Indeed some answers investigated throughout the EDA rose directly from the evidence previously discussed.

Some questions that were looked into are:
Which of the variables within every category seems to influence the response variable more? Are the predictors in each of these categories somehow related to each other and maybe redundant? Is there any pattern between variables across these categories?

3.1 Property's Features

(description, property_type, room_type, bedrooms, beds, accommodates, amenities, minimum_nights, maximum_nights)

Figure 2: Bar plots of the most 12 common amenities occurring in both cheap and expensive properties

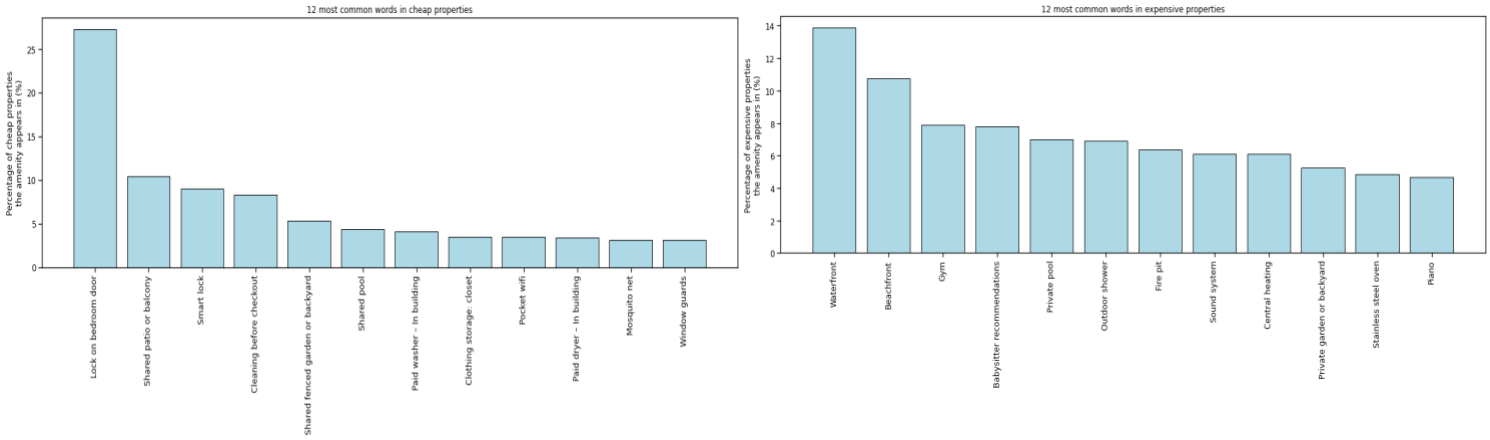


Figure 2.1: Bar plots of the 27 most common words (tokenised from "description") occurring in both cheap and expensive properties

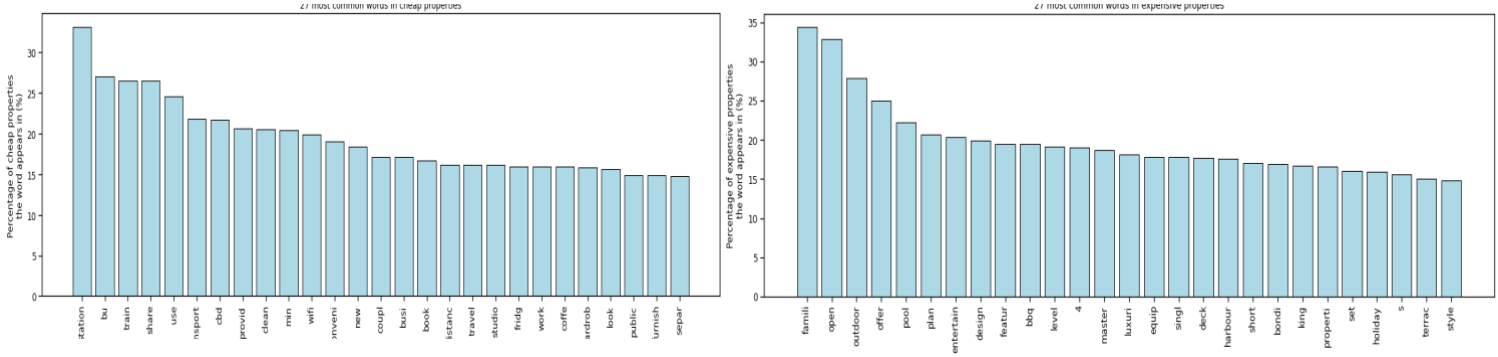


Figure 3:
Bar plot of the ratio of beds per bedrooms against the average price of such a property.

Note: rate of 0 corresponds to those categories where the number of bedrooms does not apply and therefore takes on a value of 0. In this case the average price totalized is relatively low which is reasonable given the property's types for which the "bedrooms" category does not apply. Hence, ratio of 0 takes on 1 in the dummy variable "high_occupancy_per_bedroom".

Figure 4:
Bar plot of the average price totalized by every different property_type

As predictable “bedrooms”, when plotted against “beds”, “accommodates” and “price”, shows to have a rather clear linear relationship. Strong linear correlation among beds bedrooms and accommodates might negatively influence performances of linear regression models so just one of these three variables should be kept when using a linear model.

Moreover, a pattern that was investigated is whether there exists some sort of relation between the number of beds per bedroom and the average price of the property. In Figure 3 it can be noticed how the average price seems to drop as the number of beds per bedroom approaches 3. Hence a dummy variable was created, named as “high_occupancy_per_bedroom”.

As for the variable “property_type”, since it takes on many different values corresponding to the big variety of properties listed by Airbnb, it would not be a feasible solution to compute a dummy variable for every possible property type. To find the best solution for dummy creation,
We first grouped the data by property type and calculated the average price for each property type. The results can be observed in Figure 4. Then we created three two dummy variables, “expensive_property_type” and “cheap_property_type” and “average_property_type”. Each datapoint was assign to be 0 or 1 for each of the three dummy depending on the average price totalized by its category:
<= the first quartile → cheap_property_type
>= the third quartile → expensive_property_type
(> the first quartile) & (< the third quartile) → average_property_type

As for “room_type”, it is a replica of the same information provided by the variable “property_type”, just less in details since there are fewer categories, so we did not looked into it.

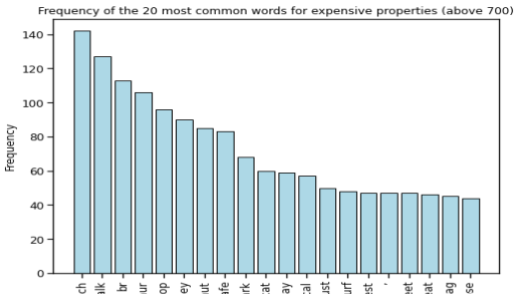
An interesting pattern to analyse for the variable “amenities” is whether there are some types of amenities that occur either just in rather expensive properties or in cheap ones. Although at first there seemed to be quite too much overlap in the types of amenities to notice anything interesting, once the overlapping types were deleted it was possible to identify 12 of the most common words for both cheap and expensive dwelling, where the threshold was set using once more the first and third quartile of “price”. Two dummy variables were then created: “cheap_properties_amenities” and “expensive_properties_amenities” that took on either 1 or 0 depending if an occurrence of either an expensive kind of amenity or a cheap one was found. In Figure 2 it can be seen which amenities were included in each dummy variable. The same approach has been used in respect to variable distribution after tokenizing the “description”. Results in figure 2.1. No significant token was identified, so we excluded “description” from our predictors.

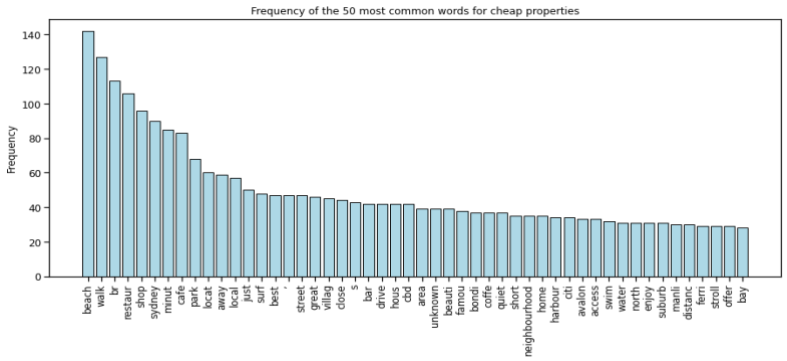
No interesting pattern was observed by plotting the minimum and maximum number of nights against price.

3.2 Location Related Variables

(neighborhood_overview, host_location, host_neighbourhood, neighbourhood, neighbourhood_cleansed, latitude, longitude)

Figure 5: Bar plot of the most common words used in describing the neighbourhood in which pricy (third quartile) and cheap (first quartile) properties are listed. It can be noticed that the most common words occur in both graphs.





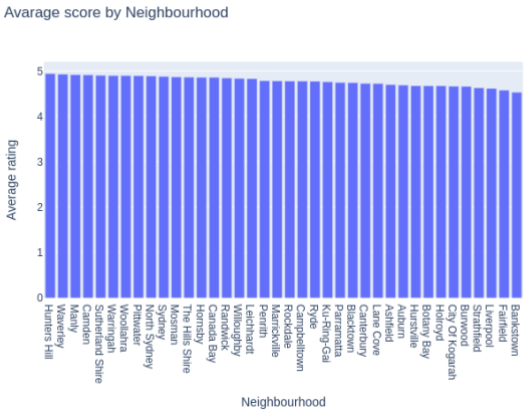


Figure 6:
Here neighbourhoods have been plotted against the average price of properties in such locations. The bar plot is in descending order so the most pricy locations appear in the left hand of the bar-graph. Moreover the most frequent neighbourhoods are highlighted on the x axes. Also note that label “Sydney” corresponds to the neighbourhood “CBD”.

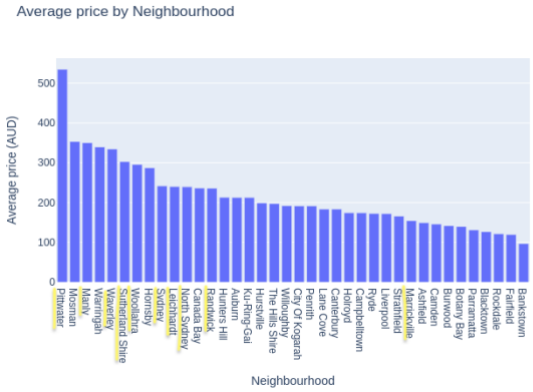


Figure 7:
Bar plot of the average score totalized by each neighbourhood

The variable “neighborhood_overview” is represented by a textual description of the surrounding. To be able to extract some useful information from it first of it had to be “tokenized”. Doing so it was possible to analyse which words are associated with the most pricy properties and which with the cheapest ones as it can be seen in Figure 5. At first the first and third quartile were used to categorize dwellings as either cheap or expensive, then a higher threshold for expensive properties was selected in the quest of some more interesting differences in the occurrence of most frequent words for these two categories. Although the most common words for both cheap and expensive properties seem to be roughly the same. Given this observation this variable is probably not very useful to perform regression over prices. It is indeed reasonable that an host would emphasize the best quality of the neighbourhood no matter the price of the property he/she is listing.

It is easily guessed that the location might be a crucial predictor of the price. In the dataset provided however there are multiple variables related to the location. As for the reasons explained in the data description section, “neighbourhood_cleansed” is the variable used to assess which neighbourhood seems to be the wealthiest one. Indeed neighbourhood_cleansed first of does not have missing variables, secondly it is consistent in the format since the neighbourhood is geocoded using

longitude and latitude. It is also obvious at this point that “longitude”, “latitude” as well as “neighborhood” do not provide any further relevant information. A dummy for each neighbourhood identified was created.

Moreover, observing Figure 6 the top five most expensive neighbourhoods can be selected, given that the rationale to pick only the first 5 is that doing so the neighbourhoods falling in the third quartile are being selected. Subsequently the dummy “expensive_neighbourhood_tokens” was created. An interesting pattern that seems worth to further investigate is whether the priciest neighbourhoods are also the best reviewed ones. Indeed the hypothesis is confirmed as those neighbourhoods that rank the highest for what concern the price are also among the top rated ones (Figure 7). As a result of this observation an interaction term between these two variables was added, namely “interaction_rating_neighbourhood”.

3.3 Host Characteristics

(description, host_since, host_response_time, host_response_rate, host_acceptance_rate, host_is_superhost, host_listings_count, host_verifications, host_identity_verified)

The result of plotting the average price for the properties where variable “host_is_super_host” took on value true and false showed no interesting pattern, hence it could be derived that is perhaps not a valuable predictor for “price”. The same conclusion was derived for “host_since” even after investigating possible patterns between number of properties listed per year vs the average price. Variables “host_acceptance_rate” and “host_response_rate” are both weakly correlated to variable price as can be observed from the correlation heatmap. Overall no interesting pattern was identified. On the other hand it was observed that the average price of a property would linearly decrease as the average response time increased.

No interesting patterns were discovered for “host_listings_count”, “host_verifications” and “host_identity_verified”.

3.4 Reviews

(number_of_reviews, review_scores_rating, review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_scores_location, review_scores_value, instant_bookable, reviews_per_month)

From the correlation heatmap (Figure 8) it can be seen that none of these variables present a strong correlation with the target variable “price”. “number_of_reviews” presents surprisingly a weak negative correlation with “price” fact that can otherwise be explained with the rational that the more expensive a property is the least affluence it will reasonably have. The heatmap also evidences that all the different variables deriving from user’s reviews are strongly correlated among each other. Same observations are deducted both when plotting the heatmap obtained with Pearson’s Correlation Coefficient and Spearman's rank Correlation coefficient.

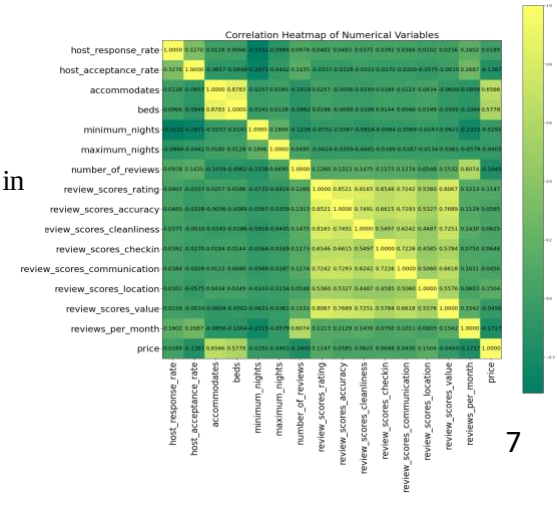


Figure 8:
Heatmap of the Pearson’s Correlation coefficient computed between every numerical variable. Lighter are in the map (yellow) indicates a strong linear correlation.

4. Feature Selection

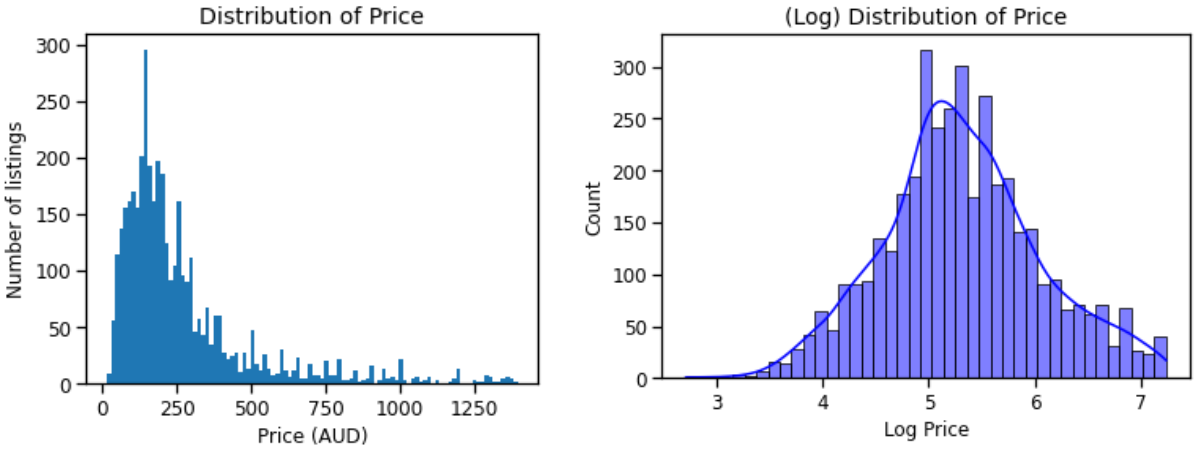


Figure 9:
The graph on the left illustrates the distribution of variable "price" with the number of listings for each price. The distribution is clearly right skewed in this first graph. On the right end side the same bar plot is reported but after a log transformation

4.1 Assumptions for Linear Regression & Outliers

To begin with, the distribution of the response variable "price" has been analysed. Indeed this investigation was performed to check whether the linearity assumption needed for linear regression models was met or not.

As clearly visible in figure 9, the price's distribution is right skewed. Hence a log transformation was performed. In Figure 8, it can be seen how the transformation successfully normalized the distribution of price. Indeed afterwards the distribution of the response variable presented a small skewness value of 0.167 and a kurtosis of -0.012. Multicollinearity assumption was also checked as it is a prerequisite for linear regression as well. In Figure 8 it can be noticed how there exist a strong linear correlation between several predictors. This of course represent a problem that depending on the regression model implemented can be dealt with in different way. Some observation can although be immediately noticed. For instance, as it could have been easily anticipated, "accommodates" and "beds" are strongly correlated; moreover they both also show to have a strong linear correlation with "price", and once again the reason for this linear relationship can be easily explained by domain knowledge. Of course accommodation that can host more people are also going to have a higher price.

High linear correlation can also be found between "reviews_per_month" and "numer_of_reviews", "review_scores_rating" and "review_scores_value". Although none of these variables seem to have a strong correlation with the target variable. Given the high correlation and on our knowledge of the dataset these groups of predictors are probably redundant and including all four of them will most likely not help explain more variance of the dataset but only increases the chance of our models to overfit. Moreover all the variables related to some sort of review seem to be strongly correlated among each other. In addition, some outliers were evidenced when plotting variables, although they were kept as could provide valuable information regarding the variability of the dataset and given the study domain they certainly do not arise from erroneous records. Moreover, removing them could somehow distort the results.

4.2 Dummy Variables and Interaction Terms Creation

The following dummy variables and interaction terms were created as a consequence of the analysis performed and explained in the EDA section.
"high_occupancy_per_bedroom"

“expensive_property_type” and “cheap_property_type”
“cheap_properties_amenities” and “expensive_properties_amenities”
“expensive_neighbourhood_tokens”
“interaction_rating_neighborhood”

The following dummy variables where created for the variable host_response_time:
“a few days or more”
“within a day”
“within a few hours”
“within an hour”

Variable “bedrooms” was encoded with 7 dummy variable as follow:
“1 bedroom”
“2 bedroom”
“3 bedroom”
"4 bedroom”
“5 bedroom”
“6 bedroom”
“greater or equal 7 bedroom”

Accommodate was also included as a predictor since it shows to have a clear strong linear correlation with the target variable.

Variable ‘host_response_rate’ was kept as a predictor for the reason explained in the previous section.

Variable ‘host_acceptance_rate’ was kept has it was noticeable that highly pricey properties would generally have a lower acceptance rate (weak negative correlation with price).

Following the same rationale ‘reviews_per_month' was kept in the list of the final predictors for price. It is indeed obvious how highly pricy properties would generally have a lower affluence and hence less reviews per month.

Variables ‘review_score_rating’ and ‘review_score_location’ were included as they seemed to be the most significant predictors among all the reviews.

5. Methodology

In this part, we split the original dataset into training data and validation data. Firstly, we estimated different models on the training data. Secondly, we made predictions on the validation set. Then, the model with best validation set performance was selected. We re-estimated the selected model by combining the training and validation sets. And finally, we assessed the performance of the selected model by making predictions on the test data and uploaded the result on Kaggle. Based on the nature of different algorithms, we chose linear regression, single regression tree and stacked model to be explained in detail as they are of different theories. In general, the linear regression supports linear solutions, the decision tree captures non linearity patterns, and the stacked model helps find the optimal combination of a collection of regressors. Besides, since Ridge/Lasso regression is an extension of linear regression, and gradient boosting and random forest both use an ensemble of decision trees, we believe it is more appropriate to explain the foundation models which are linear regression and single decision tree in details.

5.1 Model 1: OLS

Four OLS models were built based on the dataset. The first is a base model. We set all the 23 predictors in training data as independent variables and the price in training data as dependent variables. We can see that the coefficient for greater_or_equal_7_bedroom is 98.8940. This means that if we were comparing an accommodation with 3 bedrooms and 7 or more bedrooms, we would expect the accommodation with 3 bedrooms to be priced $92.4318 more (coefficients depend on the random split into training and test data, so you might get different values). The coefficient for 3 bedrooms is statistically significant, so that we can reliably conclude that the accommodations with 3 bedrooms have higher prices on average controlling other predictors constant.

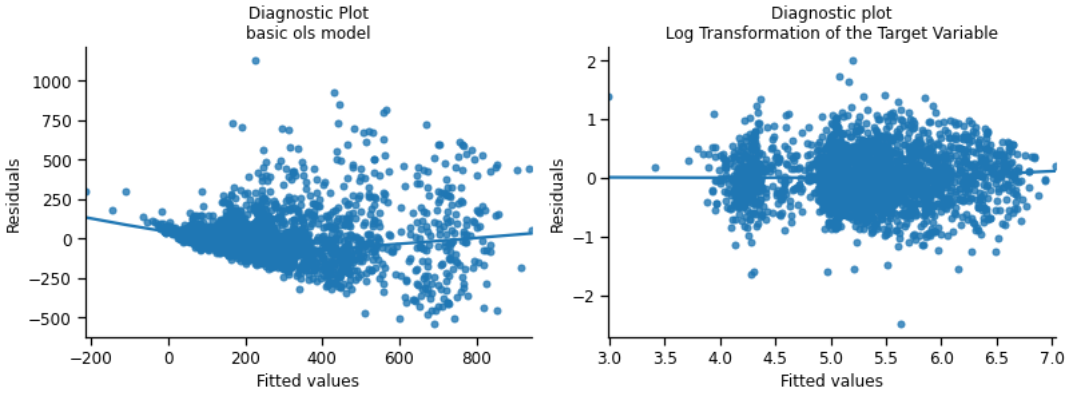


Figure 10: Residual Plots for basic OLS model (the left)
and log transformation model (the right)

The second model is the interaction model. We set accommodates and cheap_properties_amenities as an interaction variable. However, it didn't make any obvious improvement through the R-squared and the residual. The third model is the log transformation of the response variables. The residual is shown in Figure 10. Compared with the figure on the left, we can find the model is significantly better than the basic model by looking at the y-axis. The final OLS model is the log transformation of the response variable with an interaction term, but it also didn't make any obvious improvement. Overall, we concluded log transform model is the best OLS model in our analysis with a lowest RMSE of 150.474 and a highest R-squared of 0.620. In log transform OLS model, room type is an important feature because the beta coefficient associated with it is relatively large and always positive. Accommodations with more bedrooms will lead to a higher price. According to Guesty (2015), Airbnb hosts who rent one room will earn less money than Airbnb hosts who rent an entire apartment – the more rooms you provide the more opportunity to match demand and supply more potential guests. However, the price will fall if an accommodation has 7 or more bedrooms. To maximum Airbnb profits, Airbnb could try to focus on providing accommodation with 6 or less bedrooms.

Next, we used the forward selection on the original training model to get the best predictors. And the result shows the best model includes all the 23 predictors. Thus, we used all predictors in the following analysis.

5.2 Model 2: Ridge/Lasso

According to the no free lunch theorem, neither ridge regression nor the lasso universally outperforms the other. Ridge regression tends to perform better when all predictors have coefficients of similar size and the lasso tends to perform better when a few of the true coefficients are large, while the rest are (close to) zero. The top 15 beta coefficients are shown in the figure below. Our Ridge model has a

test RMSE of 155.771 and a test r-squared coefficient of 0.69. Our lasso model has a RMSE of 156.948 and a r-squared coefficient of 0.59. With the validation data, ridge performs better than lasso.

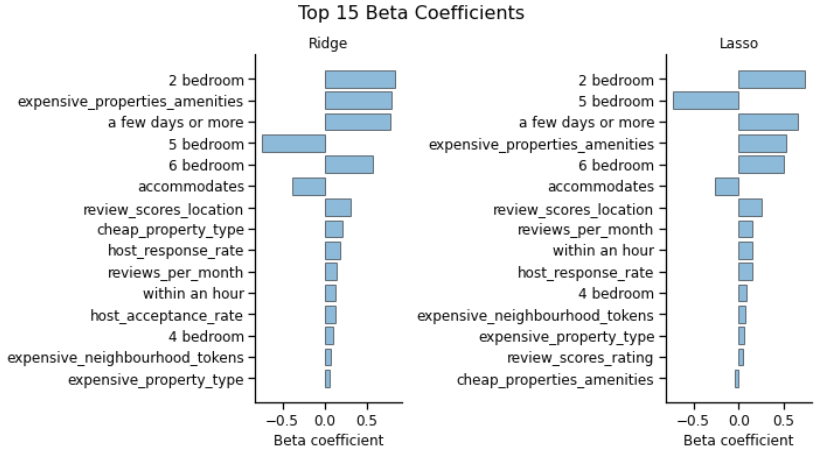


Figure 11: Top 15 beta coefficients for Ridge (the left) and Lasso (the right)

Table 2: Training RMSEs for different linear models

Model	RMSE
OLS – Base model	154.874
OLS –Interaction model	154.893
OLS – Log transform model	150.474
OLS – Log + Interaction model	150.525
Ridge	155.771
Lasso	156.948

5.3 Model 3: Single Regression Tree

5.3.1 Models Assumption and Justification

Compared to the linear model, the decision tree model loses the assumption that there must be a linear relationship between variables and that the data must be normalized, and it assumes that all training set is considered as a root, prefer categorical variable or separate the numeric variables before build model within-node prediction RMSE boundaries are more meaningful (Abhigyan,2021).When subdivided into regions, the regions produced by the tree look like rectangles and can be small or large and can fit into a very simple model. The model is simply a prediction so that one had the same prediction for every individual in the region. Within a given area, this forecast is constant. All training observations made in that area are typically taken. It is created through successive splits of the predictor space into two non-overlapping subsets, using the most popular classes in the region. As a result, it can be considering the prediction room as a decision tree. The decision tree model could begined at the top of the tree, where all the X-values are located, and subdivided the X-values into two regions using just one predictor. For example, to split the X-values into two regions. Partitioning the space, all the X-values, into two zones using just one predictor and its values. The lower value of that predictor is often moved to the area on the left. After that, the larger value is transferred to the bigger region. Therefore, a decision tree could be imaged as it has two branches that indicate a split further down the tree. (González et al., 2015)

5.3.2 Parameter Tunning

Firstly, using the sklearn package for regression tree and GridsearchCV, because mainly focused on the numerical variable within this dataset and Gridsearch could try every tunning parameter and got a

more accurate solution. We filled bedrooms with zero instead of missing values. Since the decision tree model is not sensitive to rescale, we had not chosen to standardize the data of the decision tree model. The model was selected by cross-validation of 5 folds to choose which tree would perform optimally on our validation data, and from ccp_alpha (2.257) by training alpha through the data to filter out the best one. ccp_alpha which is important for the tree model because diff size alpha will lead to different tree sizes and the penalty phase could regulate the index of tree complexity. Then to fit the data, training, filtering out the best max depth at 5 and min samples leaf at 15, getting the tree with the best result, and visualising and deriving RMSE. In last, to gain a Kaggle score, first, constructed a Dataframe by predicting the prepared data, and deriving and predicting the values. In last, the Kaggle score at public got 136.25.

5.3.3 Interpret Business Context and Visualized

Figure 12 shows the feature importance from top to bottom and focuses on two predicts, the number of bedrooms and the review score's location. Those two features might be more relevant with help Airbnb predict Sydney houses per night, from this graph we can see that there were more samples and bedroom stay was a dominant feature in this model. This shows that customer is willing to pay extra for good customer service and room experience. To maximise Airbnb's profits, Airbnb can find solutions to improve the performance of its review score locations to attract more customers.

Furthermore, the number of bathrooms was put in this first region, the tree started to grow from this region which corresponds to all possible X values. Then, we subdivided this region into two more, the left one and the right one. Predictor values were spilt based on each predictor and the number of bedrooms, X1 is used as the first predictor. And then, we spilt the training data, and we compared the training values for each predictor value of predictor X. The number of bedrooms was analysed whether less than or equal to 2.5, if it's true, the individual turned into the left area. If it's false, then turned to the right region and vice versa. As a result, the regression tree is mainly through the decline in MSE (criteria) to justify which is the best one.

5.3.4 Beneficial and Weakness of Tree Model

The advantage of the Tree model is easy to interpret and can followed the decision path directly. But there was an issue with the single regression tree: Once the tree has grown large enough, it will become difficult to interpret and not useful. So, choosing a suitable depth and size of the tree is critically essential to the analysis of the data. Secondly, it had very high volatility. If replaced new data, the structure of the tree may be completely different. If one of the conditions changed on different branches, all the other conditions must follow to change, which will cause the tree to grow quite differently.

5.4 Model 4: Gradient Boosting

The gradient Boosting model is one of the types of boosting in which the approach is to build a tree, used this tree to predict the current error in house prices, and then had a new residual. After that repetitive prior action, this residual got smaller at each step, continuing to build many trees until these become complex. But each stage will reduce the error a little. Starting with a very simple tree, predicting the average current prediction, what are the wide valleys, looking at the differences and calculating some residuals and calculating the gradient with loss. What this model wants is a combination of weak learning to achieve better results. Additionally, Figure 13 shows the feature importance of top five variables, with "beds" identified as the most important one.

Figure 12: Pruned regression tree

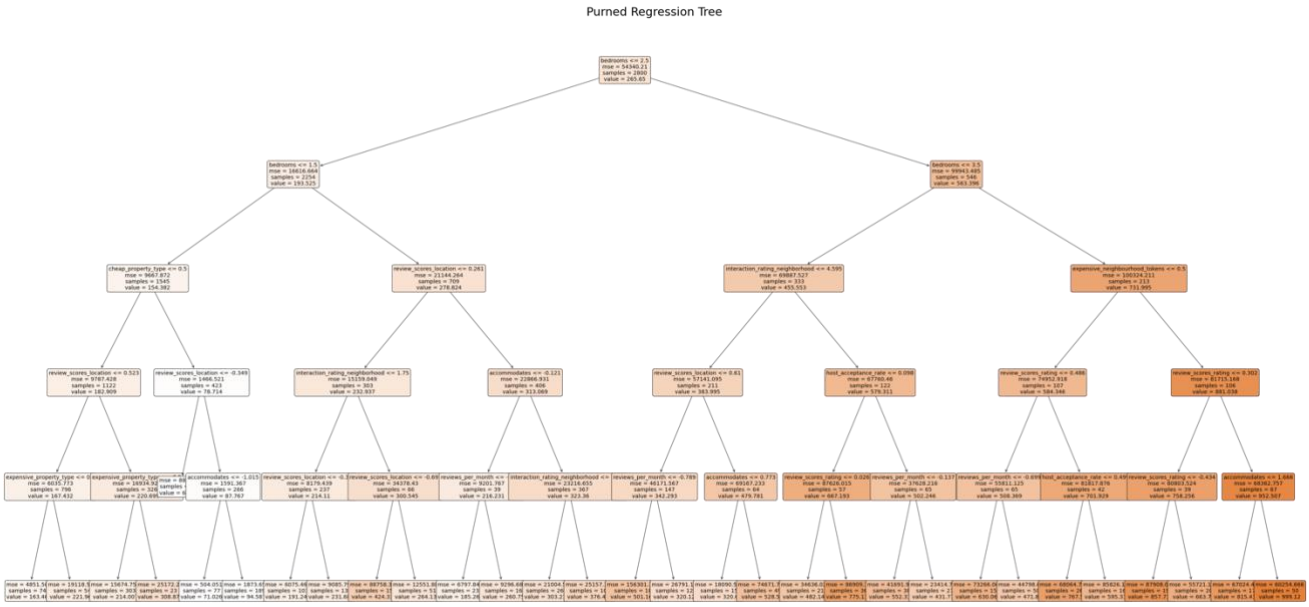
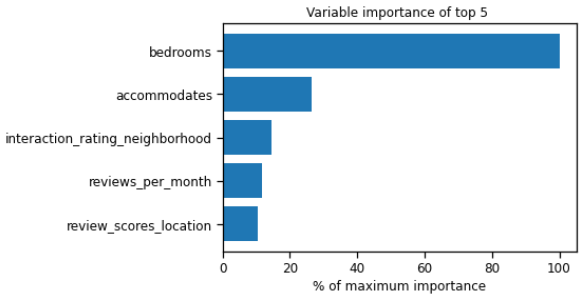


Figure 13: Variable importance of top 5



5.5 Model 5: Model Stack

In order to improve the accuracy of the prediction, the model will be stacked in an attempt to obtain better input results. The stacking model takes the output guesses of our basic model as input components of points in a new space and then generalizes in that space (Breiman, 1996). As analyzed in the previous chapter, the dataset contains numerical data and categorical data and the proportion of data is relatively average. Given that the OLS model and the tree model have different emphases on data processing, using only one type of method may lead to some parameters that cannot be well interpreted, and the stacked model solves this problem very well. Here introducing Stacking Regressor function from sklearn library for modelling。

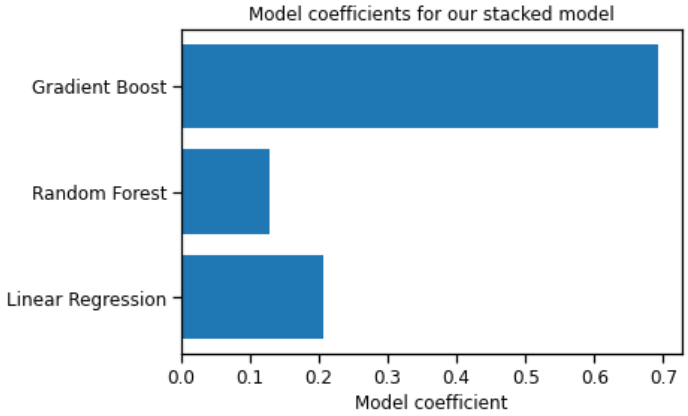
To obtain more efficient model combinations, Random Forest Model will be introduced as one of the alternative models as its generally well-performing. Compared with other algorithms, Random Forest can be used to compute high-dimensional data, has strong anti-interference ability, and still has a good performance for imbalanced data sets. Thus, parameter preprocessing is not required. In addition, it is noted that although the price is affected by all factors, it is more sensitive to only some of them, which is likely to increase the variance of the model. Also, random forests largely eliminate this adverse effect by introducing random attribute selection.

We tried to run the second-layer algorithm to further process all the models we built and found that: when the Gradient Boosting, Random Forest, and OLS models involve, the stacking instruction does not assign weights to other models, so these three data will be used as our base models.

As the error of the tree model for skewed data is not significant, the tree model is trained on the original dataset. To ensure the consistency of response values, we decided to use a linear regression model without log transformation for model stacking. LinearRegression function makes some adjustments when training on the dataset with the bias applied. According to our further validation, the prediction differences between the linear regression model (OLS) and the regression model using log transformation are not large.

The weights of each basic model of the Stacking Model are as follows:

Figure 14: Model coefficients of stacked model



Among them, Gradient boosting is the most effective, accounting for 69.1% of the model output. Random forest accounted for 12.7 % of the contribution. It is worth noting that the OLS model (without log transformation) accounts for 20.6% (the data is taken to three significant figures, and the part exceeding 100 % is due to the deviation of the computing error). According to the preliminary analysis, we have two main reasons for that:

(i) In addition to the OLS model, the other two basic models are tree models, leading to output with poor linearity

(ii) The stacking model uses linear regression as its second-layer algorithm, so the OLS model was given a higher weight to help the stacking model capture the main linear features.

After the stacking model is trained, it is predicted in the validation set (split from the train set). The RMSE of the result is 142.369, and the score of the model on Kaggle is (RMSE) 125.784, which outperforms other models. We also use this model as our final model (unfortunately, due to misoperation, this model was not selected as the final model on Kaggle)

6 Model Validation and Comparisons

the RMSE of the five models we built is shown below (the result is kept to three decimal places) :

Train RMSE Validation RMSE

	Train RMSE	Validation RMSE
Model 1: Log transformation OLS	150.474	135.022
Model 2: Ridge Regression	155.771	138.434
Model 3: Simple Tree	152.871	132.883
Model 4: Gradient Boosting	1 44.383	1 33.849
Model 5: Stack Model	1 42.639	1 25.784

Briefly, we can see the models in the train set perform better that in the validation set. We believe that this result may be caused by the following reasons:

(i) The data in the validation set is much easier to identify and analyze than the data in the training set. This situation may be caused by multiple reasons, for example, the property data of the

training set and the validation set are not collected in the same time period; the validation set and the training set are directly split from a data set that is pre-sorted by the recommendation algorithm, etc. We plan to introduce Binary classification to test this conjecture

（ii） The model is underfitting and the validity is too low. We will try to remove the description variable or increase the dimension of the model to see if there is any improvement.

In terms of model type, the linear regression model does not perform as well as the tree model in predicting property prices. Considering the uniqueness of the original data, we believe that this result is due to the existence of too many categorical data such as property_type and neighbourhood_cleansed in the original data, which reduces the validity of the linear regression model.

For Linear Regression Model (OLS) model, based on our EDA analysis, the log-transformed data for the price (response) exhibits a normal distribution, so the linear regression model (OLS) performs better when adopting a log-transformed response. Additionally, we note that the validity of the Ridge regression model is lower than that of the linear regression model (OLS). Ridge Regression is based on the linear regression model, by introducing weight parameters alpha to reduce the influence of feature parameters on the model. The alpha of
Our Ridge model is 0.487, and the risk of overfitting the model has been reduced. However, since our model itself may be underfitted, the model does not perform well overall.

For the tree models, Gradient Boosting performs significantly better. However, in the validation set, Gradient Boosting is as effective as the simple tree model. We learned that the effectiveness of the Gradient Boosting mode will be more likely to reduce if iterated too many times. In a further study, we will modify the tuning parameters of the model to obtain better performance.

The stacking model performs better than other models in both the training set and the validation set. This is because the stacking model integrates the advantages of the linear regression model and the tree model. However, the performance of the random forest in this model is weaker than that of the linear regression model. We believe this is due to our ignoring that house prices do not exhibit a direct linear relationship with other variables. This caused the function to have to increase the weights of the OLS model to improve its linearity. This weakens the stacking model performance.

7 Conclusions

Overall, our models are based on a commercial perspective exploring the various factors that affect rentals on Airbnb. We classified variables and discussed them. The major variables influencing factors such as location factors, the number of houses, and the landlord's response time are focused. We also incorporate some minor factors into consideration to improve the model's effectiveness. We attach great importance to the quality of the data such as eliminating the noise and imputing the missing values of the original data. Generally, our model reflects the relationship between the property rentals of Airbnb and various factors to a great extent, which can be considered a positive attempt.

It cannot be ignored that there is still much room for improvement in our work. We removed the description and did not try to use NLP to deal with the relevant content of this factor, which may lead to some potential influencing factors loss. Though the house location information was rebuilt as location score, the analysis of the Lasso model showed that is not significant, which may imply the processing of this variable is insufficient. Given that location plays a dominant role in housing prices, we will conduct further engineering on this factor in the future to increase the weight of this factor.

References

Abhigyan. (2021, November 8). *Understanding decision tree!!* Medium. Retrieved November 6, 2022, from https://medium.com/analytics-vidhya/understanding-decision-tree-3591922690a6

Breiman, L. ,1996, . *Stacked regressions.* Machine learning, 24(1), 49-64. https://link.springer.com/article/10.1007/bf00117832

G. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction to Statistical Learning: with Applications in R. Springer, 2013. [Online]. Available: https://faculty.marshall.usc.edu/gareth-james/ISL/

Guesty. (2015). What do the most successful Airbnb's have in common? https://www.guesty.com/blog/what-do-the-most-successful-airbnbs-have-in-common/

González, C., Mira-McWilliams, J., & Juárez, I. (2015). Important variable assessment and electricity price forecasting based on regression tree models: Classification and regression trees, Bagging and Random Forests. *IET Generation, Transmission & Distribution*, *9*(11), 1120-1128

8.Appendix

8.1 Statement of Contribution

SID	Responsibility	
	Codes	Report
480140973	1. Data processing 2. EDA 3. Feature engineering 4. Train test split 5. One gradient boosting model 6. Kaggle files creation and submission	1. Cover page 2. Data processing 3. Codes formatting 4. Report formatting
520600843	1. Data processing 2. EDA 3. Feature Engineering 4. Random Forest 5. Kaggle submissions	1. EDA 2. Feature Engineering
480038647	1. Ordinary least square model 2. Ridge and Lasso	1. Methodology for 2 models 2. Introduction and conclusion
510488543	1. Single Regressing Tree model 2. Gradient Boosting Model 3. Kaggle files submission	1. Methodology for 2 models 2. Organize report structure
500135329	1. Model stacking 2. Validation and comparison	1. Methodology for model stacking 2. Validation and comparisons 3. Report formatting

8.2 Meeting Minutes 1

Date: Sat Oct 22, 5 pm-6.10 pm
Present: 480140973, 520600843, 480038647, 510488543, 500135329

Agenda:
1. Look through the instruction together
2. Discuss coding task allocation
3. Discuss page number allocation

Actions:

480140973&520600843 - data processing & EDA & preliminary feature selection (before 25/10/22)
480038647 - interpretable linear models (OLS, Lasso) (before 29/10/22)
510488543 - single regression tree & boosting (before 29/10/22)
500135329 - model average, validation and comparisons (before 29/10/22)

In the meeting, we created a google doc and workplace for file sharing and synchronizing editing. We first focus on understanding the assignment content and requirements. Then, we chose models that we

wanted to apply to our assignment. We agreed to have two members get EDA done first and the rest groups should review the lecture content during this time.

8.3 Meeting Minutes 2

Date: Thu Oct 27, 5 pm-6 pm
Present: 480140973, 480038647, 510488543, 500135329
Unattended: 520600843

Agenda:

1. Shared findings of the data cleaning and EDA part which has been finished
2. Discuss page allocation of the report
3. Discuss the task allocation

Actions:

480140973&520600843 - data processing & EDA report(before 5/11/22)
480038647 - interpretable linear models (OLS, Lasso) (before 3/11/22)
510488543 - single regression tree & boosting (before 3/11/22)
500135329 - model average, validation and comparisons (before 3/11/22)

People who was responsible for the first part shared their code in the data cleaning and EDA part. All of the members actively participated in the discussion. Other 3 members will start the model-building part and finalize the code before 31/10/22. We will have three members working on the methodology section, and the paper number was allocated by the model complexity.

8.4 Meeting Minutes 3

Date: Mon 31 Oct 8 pm-9.30 pm

Present: 480140973, 520600843, 480038647, 510488543, 500135329

Agenda:

1. Checking each modeling process
2. Find the problem in modeling and further improve
3. Set a deadline for the final code completion

The person that built linear models explained her thoughts and ideas on how to build the OLS model and the one that built tree models explained how to build the decision tree and boosting model. We discussed the model code improvements and helped to fix the problems which finding in the process of modeling, and set a deadline of 8 pm tomorrow night to complete the model with all parts with validation and predictions and pre-prediction log adjustments before prediction was made and the rest people also need to complete his average stock model.