1. Introduction

With the ever-increasing computational power of CPUs and GPUs and the adoption of machine learning in the business field, people are trying to make descriptive and predictive analyses for many activities. For instance, investment in real estate, financial derivatives and the stock market, understanding the factors and drives behind the consequences is vital to make a better decision, therefore, increase profitability. As a team of data scientists and business analysts, we decided to do some data mining and build a few predictive models to support and answer our clients' dilemmas with the Airbnb house rentals market in Sydney. This report will start with problem formulation based on a discussion of decision theory based on pricing policy on Airbnb. After that, it will guide the audience through the procedures covered in data understanding, exploratory data analysis and feature engineering. More importantly, three machine learning models will be introduced and explained in detail. The lowest Root Mean Squared Logarithmic Error (RMSLE) is 0.41092 on Kaggle Public Score (test set), which is generated by Model Stacking. Lastly, this report will provide some suggestions for host and real estate investors with the help of data mining insights and findings.

2. Problem formulation and objectives

From the perspective of decision theory regarding Airbnb house rentals pricing, it is a prediction policy problem. In this machine learning project, the decision-maker or agent for newly listed houses is the predictive model, which can set a higher price or lower price according to house attributes such as location, accommodates, host status, etc. Hence, the state of nature for this procedure is whether the house should be priced at a lower level or higher level compared to the market average situation. By adopting predictive models, hosts, property managers and real estate investors could adjust their listed items' price on Airbnb and lure more tourists' and tenants' attention, which can increase their investments' returns and popularity eventually. Meanwhile, this model could also detect some irregular market behaviors, such as the unreasonable pricing from some landlords, who are eager to rent their houses at a competitive and lower price. On the contrary, some hosts may want to maximize their profitability by setting a high price due to the information asymmetry between supply and demand. As a result, the predictive model tries to price houses fairly and help hosts and investors to make a balance between their price and quantity, which can increase their total revenue and decrease some opportunity costs. On the other hand, data mining can also be helpful for new hosts and experienced hosts by demonstrating some insights discovered on the 'best hosts'. It is important to reveal what kind of factor is influencing hosts' income and popularity, and help opportunity-seeking investors to make wise choices with regard to house status, house facilities, pricing and operating strategies.

3. Data Understanding

Before starting the data preprocessing and preliminary analysis, a good understanding of the original dataset is essential. There are 20408 records and 69 features in the raw data, in order to understand these features comprehensively, the original data set was categorized into *host, location, room, night, avail, review* feature groups. In this case, *host* contains information related the host himself or herself, such as some features measuring their reputations, service quality, identity information, etc. *Location* contains very meaningful information about the geographic locations, where 'neighbourhood_cleansed' covers the name of surrounding area, the combination of latitude and longitude illustrates the exact location of that house. *Avail* contains how popular the house is since if it is not available for a long time, this house might be very popular in the market. In addition, *review* is the counterpart of *host*, because these features evaluate the house from a customer perspective.

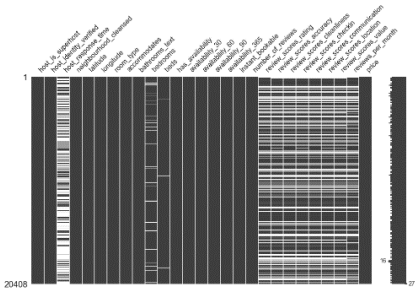
Next, to dive deep into each feature by category and filter the possible useful predictors. That is, we adopted a filter method for this project's feature selection, which is selecting the useful and meaningful features before training models. For many host-based features, such as 'description' and 'host_about' may potentially be useful with the help of natural language processing, however, our team decided to ignore them due to the potential subjectivity included in the host's information and focus more on the properties of the house and consumer's opinions. Furthermore, columns that contained too much missing values will be removed from our target feature lists for two reasons. Firstly, it may not be enough observations for later investigation and model training. Secondly, it is possible to fill in some numerical columns with their median, mean or mode, however, this could be a risky move because the filled values are not guaranteed to be the same as the original value, which might cause some errors in decision-making. Hence, by checking the data information and number of missing values for each feature, we can initially filter out the above features and keep 26 features and 1 response value 'price'.

Groups	Features
Host	'host_is_superhost', 'host_identity_verified', 'host_response_time'
Location	'neighbourhood_cleansed', 'latitude', 'longitude'
Room	'room_type', 'accommodates', 'bathrooms_text', 'bedrooms', 'beds'
Avail	'has_availability', 'availability_30', 'availability_60', 'availability_90', 'availability_365', 'instant_bookable'
Review	'number_of_reviews', 'review_scores_rating', 'review_scores_accuracy', 'review_scores_cleanliness', 'review_scores_checkin', 'review_scores_communication', 'review_scores_location', 'review_scores_value', 'reviews_per_month'
Response	'price'

IMPORTANT: in this report, all visualizations are provided in thumbnails, which can be used as a reference to find the corresponding figures in Appendix.

4. Exploratory Data Analysis and Feature Engineering

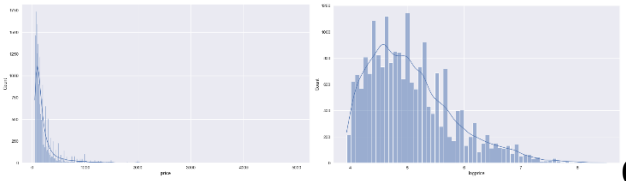
In the next stage, feature engineering and missing values computation will be processed simultaneously based on the information generated by missingno package and insights generated from data visualizations.



(Visualization 1.1)

- Price (data scaling)

At first, let's look at the response variable 'price', which is the target value in this project. As a numerical value, it was stored as an object type because of the currency unit. So, we need to clean the whole column into a numerical data type. From the histogram of original price values, we can see the distribution is right-skewed. This phenomenon should be reasonable since most Airbnb houses are in a price range within $500 per night, while in the minority, a few uncommon houses will cost a few thousand Australian dollars per night. Therefore, a log transformation should be applied to stable the data and mitigate the skewness. In the log-transformed distribution, we can see that the logprice's distribution is now more centered. And we will use 'logprice' as the response value in comparison and model building.

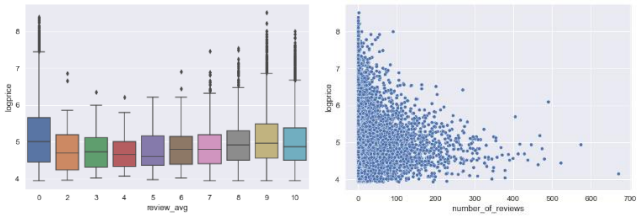


(Visualization 1.2)

After that, we dropped 'host_response_time' and 'reviews_per_month' columns because they contained impractical information and large number of missing values.

- Reviews (merge information)

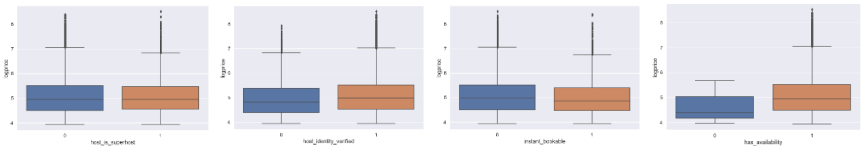
From the customer perspective, the average review score is a required feature concerning the Airbnb house quality. The average review score is computed by taking the average score of information accuracy, cleanliness, check-in experience, host communication, house location and the value of it. And we filled in 0 for missing values which represents not given rather than very low score. Starting from the middle review score of 5, we can read a tendency that the higher the review score goes, the higher the median of price would be and vice versa except for score 0. As for number of reviews, we will keep it as the original form. The scatter plot shows that higher price houses received lower number of reviews due to the consumer's affordability and demand.



(Visualization 1.3)

- Binary predictors (process data format)

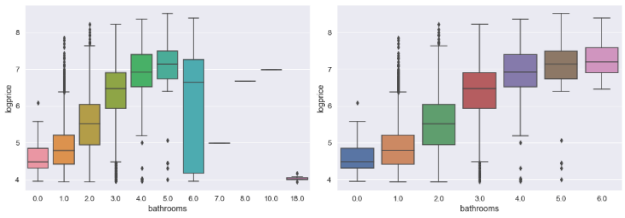
In these selected features, there are four initial binary predictors which are 'host_is_superhost', 'host_identity_verified', 'instant_bookable' and 'has_availability'. We need to convert ‘t’ and ‘f’ into 0 and 1 for the later model inputs. Both host verification and house availability information showed obvious differences on the average prices. Since customers are more likely to spend money on apartment owners who are ‘trustful’, and apartments with no availability would have much lower price compared with those who have availability.



(Visualization 1.4)

- Bathrooms (process data format & sparse categories)

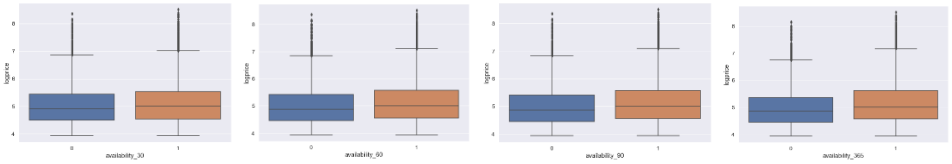
Because the dataset was directly scratched from website, it also contained some short sentences demonstrating house facilities. The ‘bathrooms_text’ is an example of that. As we did data cleaning in price column, the similar approach was used to get the numerical data representing number of bathrooms. However, this column has more issued to solve, such as 15 missing values and 8 extreme values which claims the house has 18 bathrooms (it should be an input error). After a detailed investigation, we decide to replace 1 for these 23 observations’ number of bathrooms because most of the houses are suitable for few people with small number of bedrooms or beds. On the other hand, merging the observations that has more than 6 bathrooms into 6 bathrooms category because we want to create lower number of categories for most features, which can be easier for audience to interpret the findings, and useful for model buildings. From the boxplot we can easily tell that the more bathrooms the living space has, the higher price it would be. Usually, the larger a living space is, a larger number of tenants will move in, and the more bathrooms the house would have. Therefore, we assume that the number of bathrooms should have a positive correlation with price.



(Visualization 1.5)

- Availability (process data format & redefine the meaning)

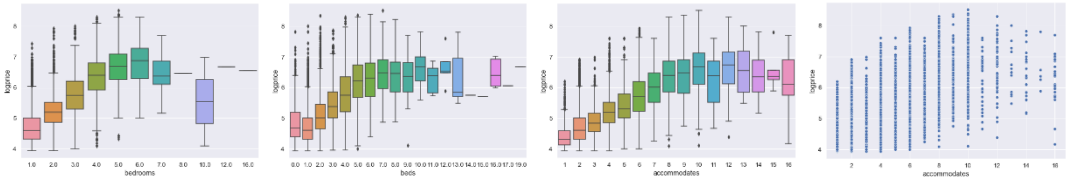
For 'availability_30', 'availability_60', 'availability_90' and 'availability_365', these columns have provided information about how many days the house is available at first place. We converted them into binary predictors as well, and they can provide some information jointly. For example, if a house gives 0 for all four columns, this will indicate that the house is frequently needed by Airbnb users in a year if everything is all right with the house. On the other hand, a house that shows availability in all four columns might be caused by the higher rentals potentially.



(Visualization 1.6)

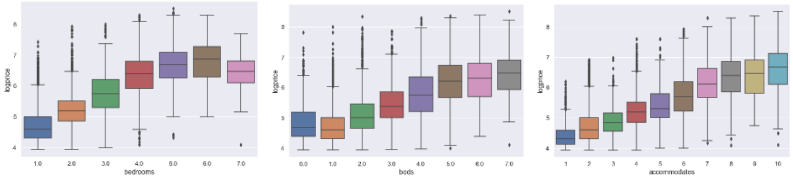
- Accommodates, Bedrooms and Beds (missing values & sparse categories)

According to previous analysis, we firstly need to compute the missing values in 'bedrooms', 'beds' columns. Rather than filling in the mean values, we must fill in the median for these missing values because both items are discrete values rather than continuous values (number with decimals). Afterwards, we merged observed values as the procedure used in handling bathrooms based on the findings from the following graphs. For both bedrooms and beds, we will redefine the meaning of 7 as 7 or more than 7 beds or bedrooms. In addition, for ‘accommodates’, we merged more than 10 accommodates categories into 7 accommodates due to the similar distributions of their prices.



(Visualization 1.7)

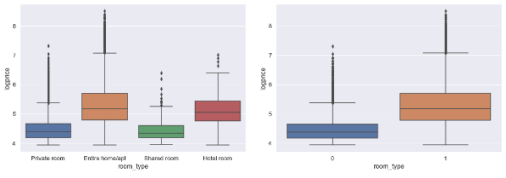
When talking about accommodates, number of bedrooms and number of beds, we would assume that they all have a positive correlation with price, such that the more visitors the living space can accommodate, means either the more bedrooms it has or the more beds it has. The boxplots below can illustrate our assumptions.



(Visualization 1.7)

- Room types (process data format & merge information)

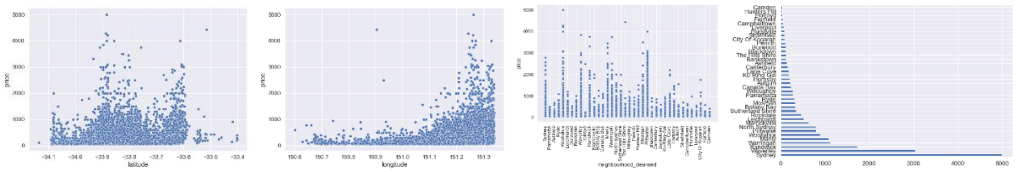
There are four different room types, private room, entire home or apartment, shared room and hotel room. From the following price boxplot among different room types, we can read that the price of an entire home or apartment is the highest, then the price of a hotel room is the second highest, probably because hotels can offer services and daily room cleaning. Both private room and shared room would have a lower price compared with the other room types. As a result, the room types are transformed from nominal data to numerical data and split into ‘small house’ and ‘big house’ categories using numbers 0 and 1.



(Visualization 1.8)

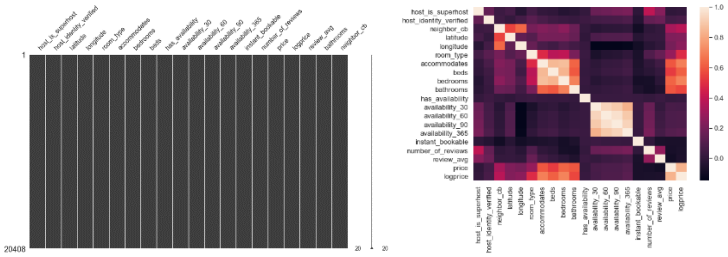
- Geographic location (target encoding)

In this research, continuous variables are very rare, however, the latitude and longitude are two very few and meaningful continuous variables as they present the precise geographic coordinates of the house. On the other hand, the 'neighbourhood_cleansed' works as a supplementary information related to latitude and longitude, which tells the surrounding area of the house. Since this column contains nominal data with relatively higher number of categories, we decide to use target encoding to replace categories with their corresponding average response values. Although it may result in a data leakage problem, it is very helpful for further modelling because house rentals in different blocks and business district differs a lot.



(Visualization 1.9)

In previous parts, the original dataset was processed thoroughly with missing value imputation, data scaling, data reformatting, merging sparse categories, target encoding, etc. As a result, 20 columns remained in the created data frame with 18 features and 2 response values. It is important to notice the potential existence of multicollinearity. As we can see from the heatmap visualization of the correlation matrix, three geographic features are partially correlated, as well as six features related to the house facilities and four features measuring house popularity.



(Visualization 1.10)

5. Methodology

- Train-Valid split

In machine learning, the most basic goal is to generalize beyond the provided observations given by the training set, because it is very hard to see the exact same observation in the test set or our real applications (Domingos, 2012). In some cases, the model has poor generalization performance because it overfitted the training data. Therefore, we need to use techniques like cross-validation and regularization to combat the overfitting problem. In this research, we will split the original training data into training set (70%) and validation set (30%). Using training set to estimate the models' configurations like hyperparameters and coefficients based on validation set performances. Therefore, we also need to apply performance evaluation metrics like Root Mean Squared Logarithmic Error (**RMSLE**), Root Mean Squared Error (**RMSE**) and Coefficient of determination – R squared ($\mathbf{R^2}$). In addition, we will test the models' true generalization performances by using Kaggle Public Leaderboard Scores. In this report, we will mainly discuss on Linear Regression, Light Gradient Boosting Machine and Model Stacking separately.

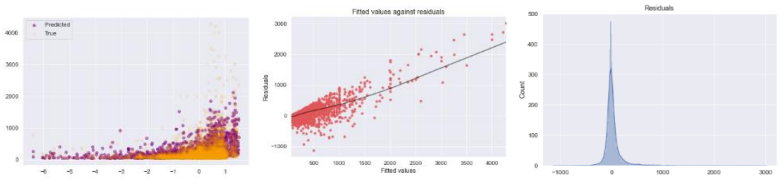
Models	RMSLE (Valid)	Kaggle Public Score (Test)	RMSE (Valid)	R^2 (Valid)
Linear Regression (Benchmark)	**0.456**	**0.45915**	193.605	0.568
Lasso	0.548	0.55924	257.796	0.233
KNN (n=28)	0.443	0.44864	193.702	0.567
Random Forest	0.418	0.42048	181.344	0.621
Generalized Additive Model	0.446	0.44714	188.299	0.591
Gradient Boosting	0.418	0.41569	174.229	0.650
Cat Boost	0.412	0.41195	170.955	0.663
Light GBM	**0.411**	**0.41473**	171.836	0.659
Model Stacking	**0.411**	**0.41092**	171.668	0.660
Model Average	0.421	0.42424	187.821	0.593

- Linear Regression (Benchmark)

Linear regression is a widely used linear model across different regression tasks. James et al. (2021) stated that linear regression is a simple and useful approach for supervised learning such as predicting a numerical response. A linear regression model can be described as:

$$\mathbf{Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon}$$

where Y is the response value, X_1 to X_p represent all features we passed in the model, β_1 to β_p are the corresponding coefficient for each feature, β_0 is the intercept estimated in the model, and ϵ refers to the error term or the noise. For this Airbnb house rental prediction task – a supervised learning task with numerical response value, linear regression is very suitable as a benchmark model. Because it is fast to implement and easy to interpret. These coefficients and intercept can be estimated by using ordinary least squares method or scikit-learn package in a very short time. And coefficients can be seen as a weight for every feature, which showed how a feature is correlated with the response value – 'logprice'.



(Visualization 2.1)

After the model is fitted, we plot the predicted values with the true values from validation set, and plot fitted values against their residuals. From these two scatter plots, we can infer that this linear regression model has violated some model assumptions such as homoscedasticity (constant error variance) and linearity. Using histogram to visualize the residuals, we can see that it looks approximately normally distributed, however, the residuals do not follow a Gaussian distribution with some extreme large or small values. Thus, linear regression tends to underestimate observations with very large response values, and it may overestimate small response values because the model tried to find a best coefficient combination to minimize the sum of squared error. As a result, the model accuracy is not very ideal. The RMSLE in validation set is 0.456 while the Kaggle Public Score is 0.45915 in RMSLE. Despite of the accuracy, linear regression's first advantage is its interpretability. The model's output showed that it assigned relatively larger positive coefficients on features like accommodates, bedrooms, bathrooms and its room type. On the other hand, longitude has a positive coefficient while latitude has a negative coefficient, which is reasonable because the larger the longitude is, the closer the position is. In Sydney, that means closer to the coastal area or beach.

- Light GBM

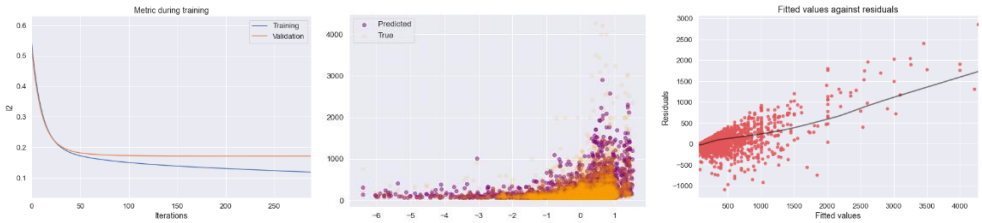
To increase the prediction accuracy, we need to implement other models with higher capacity. Because the house rentals data is a tabular data, Boosting should be considered as a feasible choice to improve prediction results. According to James et al. (2021), weak learners like decision trees are used sequentially in the procedure of boosting, which is growing trees step by step with the help of last step's information – current residuals.

$$\boldsymbol{f}(x) = \sum_{m=1}^{M} T(\boldsymbol{x};\, \boldsymbol{\theta}_m)$$

James et al. (2021) also mentioned that 'learn slowly tend to perform well' in machine learning, therefore, we can also apply a shrinkage factor $\boldsymbol{\lambda}$ to reduce each new tree's contribution and giving more opportunities to different types of simple decision tree (weak learner). So, many weak learners could be added up to a powerful learner, and it performs well on prediction and handles the overfitting issue efficiently. After knowing the concept of boosting, here is the reason why we choose Light Gradient Boosting Machine (Light GBM). In house rental market, the price is updated every week, every day, and even for different hours due to larger demand in holidays. So, under the challenges of big data in real applications, the framework we use should be 'light', which means lower computational cost and higher efficiency.

Light GBM achieves an improvement in a smarter way, that is, it uses a histogram-based method to reduce the training time and memory usage and it implements leaf-wise growth instead of level-wise growth which requires high computational cost (Ke et al., 2017; LightGBM, 2021). However, because of light GBM's leaf-wise growth method, if we do not set constraints on the depths of trees, it would easily lead to overfitting. In addition to the depth of trees, the learning rate could shrink the contribution of previous trees to spare room for future trees to improve the performance. In other words, a smaller learning rate helps to improve prediction accuracy. Another hyperparameter is the number of trees, as boosting is slow to overfit, we could increase the number of trees to lift the capacity and reduce training error. Furthermore, we could sample a fraction of observations to grow a new tree at each iteration to regularize the effect and speed up computations. Meanwhile, as mentioned in introducing the idea of boosting, Lambda l1 and l2 are used to penalize tree complexity to avoid overfitting in Light GBM.

In order to get a comparatively optimal hyperparameter combination, we use the Optuna (a Python model optimization package) to select parameters with the help of 5-fold cross-validation in half-hour. Based on the best parameters generated by Optuna, there will be no more than 38 leaves in one tree, and at least 20 data in each leaf, which could prevent overfitting. Lambda_l1 and lambda_l2 are used to limit the contribution of each newly added decision tree. In every 4th iteration, this model will randomly select 93.9% of the data to use for the next 4 iterations. It will select 71.9% of features before training each tree. And both bagging_fraction and feature_fracture help to speed up training and deal with overfitting. As for iteration, we use the early stopping method to set the maximum number of boosting iterations. At the end of each iteration, we compute the validation error and stop iteration as if the validation error stops decreasing. With this embedded early stopping function in Light GBM package, the training time of the model was extremely fast, which is 929ms (nearly 1 second), because the iteration stops at the 288th round. In this research, we also used Gradient Boosting from scikit-learn and Cat Boost (another algorithm that uses GDBT) to make a comparison. Gradient Boosting takes 56s to train the model with some hand-tuning parameter and default configuration, while Cat Boost with its default configurations requires 7.53s to train the model. So, the difference in computational cost is quite significant, although the other two models are not optimal yet.



(Visualization 2.2)

By comparing with the benchmark Linear Regression, we can see that Light GBM's distribution of residuals is more flatten, which indicates a higher prediction accuracy. In the validation set prediction, Light GBM achieved a near 10% improvement compared to the benchmark linear regression, whose RMSLE is 0.411 and the Kaggle Public Score is 0.41473.

- Model Stacking

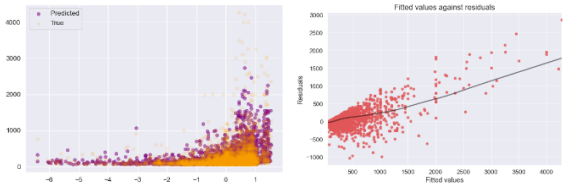
In this project, except training different machine learning algorithm models with hyperparameter optimizations to find the best model, we can include many algorithms at the same time to meet the business requirements. Some ensemble learning techniques should be considered. Model stacking is a useful ensemble learning approach in machine learning, which combines different regression models by using a meta-regressor (Breiman, 1996; Mlxtend, 2021).

$$\boldsymbol{f_{stacking}}(x) = \sum_{m=1}^{M} \boldsymbol{w}_m \boldsymbol{f}_m(\boldsymbol{x})$$

Our ultimate goal is to make our model achieve the best generalization performance so that the model stacking should be applied as a possible final model. Except for the formerly mentioned models like Linear regression, Gradient Boosting and Cat Boost. We also exploited Lasso, k-Nearest Neighbors (kNN) and Random Forest for model stacking (Light GBM is not considered because it has different training process rather than a scikit-learn estimator, which is not a feasible input). Here are some brief explanations for why adding in these models:

Lasso is also known as least absolute shrinkage and selection operator, which is a linear regression model embedded with feature selection and regularization. It has these two characteristics because it penalized the complexity of linear regression model on its coefficients' absolute value. As the penalty term $\boldsymbol{\lambda}$ increases, the shrinkage factor decreases from 1 to 0, if $\boldsymbol{\lambda}$ is close to asymptotic infinity (QBUS6810, 2021), Lasso will produce a constant value estimation (a linear regression with only intercept). And then, kNN is a flexible model which has no coefficient and estimates new observations based on similarity. The primary reason we use kNN is that it is conceptually easy to understand with real applications. In this house rentals prediction task, it makes predictions by searching all training data for several 'nearest neighbors' which have similar host status, room facilities, geographic locations, etc. After that, the response value is the average response value of these most similar observations, which is quite fair if all these features are undeniable related to daily house price. However, this model is suffering from the curse of dimensionality which is the model will have worse prediction accuracy as the number of features increase. On the other hand, it is also an expensive model due to the instance-based learning style, which requires training data storage. As for Random Forest, this model was added because we want more opinions from different types of models in the final model stacking. As another tree-based method that forces each split of decision trees to consider a subset of the original features, resulting in decorrelates among trees (James et al., 2021). Just like we are giving Random Forest a chance to contribute to model stacking, decision trees are compelled to give more opportunities for other less important predictors in Random Forest, to differentiate trees and make a more comprehensive final decision.

Afterwards, passing six models into StackingCVRegressor (a framework in Mxltend) and compute the best combination of weights for models (a linear model stacking) by using 5-fold cross-validation. From the output, we found that Cat Boost, Gradient Boosting and Random Forest get higher weights, while the other three simpler models get very small weights. So, this model stacking regressor pays more attention to more complex and accurate models.



(Visualization 2.3)

From graphs, we can see that model stacking has a similar residual plot in Linear Regression, however, the metric showed an improvement in prediction accuracy, so we can infer that model stacking has better performance in lower house price prediction compared to the benchmark. From metrics, we can see that model stacking works as good as Light GBM with respect to validation set performance with 0.411 RMSLE, however, it beats Light GBM on generalization performance, the Kaggle Public Score is 0.41092, which is even lower than the validation set metric.

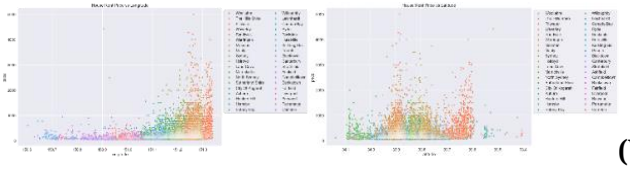
6. Results

Models	RMSLE (Valid)	Kaggle Public Score (Test)
Linear Regression (Benchmark)	**0.456**	**0.45915**
Lasso	0.548	0.55924
KNN (n=28)	0.443	0.44864
Random Forest	0.418	0.42048
Generalized Additive Model	0.446	0.44714
Gradient Boosting	0.418	0.41569
Cat Boost	0.412	0.41195
Light GBM	**0.411**	**0.41473**
Model Stacking	**0.411**	**0.41092**
Model Average	0.421	0.42424

Recall from the summarized table for model performance comparison, the best model in this research is Model Stacking. However, the Light GBM achieved a similar result on the validation set, however, its generalization performance on the test set is slightly lower than the best model. In order to make a clear comparison, we also tested input models in model stacking, which is led by Cat Boost, Gradient Boosting and Random Forest, whose parameters are also optimized by Optuna to prevent overfitting. On the other hand, k Nearest Neighbor seems to have little improvement compared to the benchmark, which might be caused by a larger number of inputs. Overall, this project's model accuracy is on a moderate level compared to other Kaggle competitors, but we do prioritize the goal of generalization during model building. Additionally, we prefer to build simpler and faster models by limiting computational time and model complexity, because the house market updates frequently, which implies that a heavy model may be inappropriate and expensive although it can predict more accurate results. Nevertheless, our project also can be updated by integrating state-of-the-art neural networks or auto-machine learning frameworks (such as TPOT), and try to generate more accurate results for houses, and finally, providing more justified and well-quality instructions for our clients.

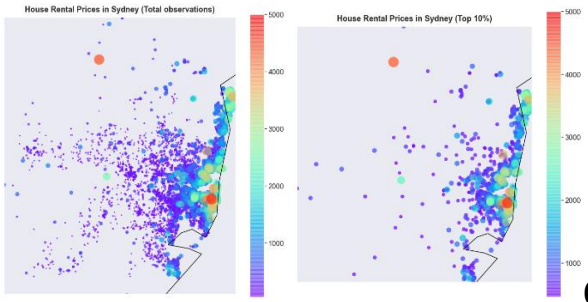
7. Data mining

For the question “what are the best hosts doing”, we need to define what “best” means. In our opinion, the best hosts are house owners who can maximize their revenue. As we define it in this equation ***Revenue = Price * Quantity***, our data mining process will mainly focus on what features will affect the house rental prices and the number of booking. However, there is no feature directly measures the sales of quantity, we will examine related information such how popular the house is. Thus, we selected three aspects, which are geographic locations, room types and popularity in that area.



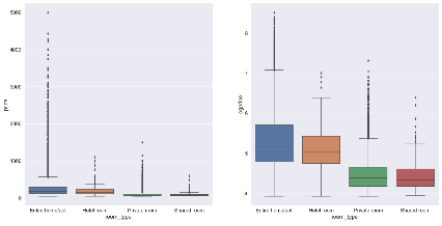
(Visualization 3.1)

Firstly, features like 'neighbourhood_cleansed', 'latitude', 'longitude' are used to find out what geographic locations are highly correlated higher price. Based on these two scatter plots, we can see that house rent prices between 151.2 and 151.3 in longitude, between -33.9 and -33.8 in latitude and between -33.7 and -33.6 tend to have higher price observations than other areas. And then we should know what these geographic numbers mean in the real world, so we put them on a map.



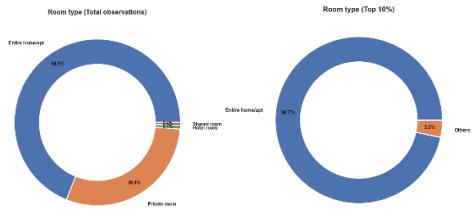
(Visualization 3.1)

To visualize price information on a map, we find out that those rental houses which have higher prices are mainly located at the coastal area which are close to downtown Sydney. We can also find that there are smaller number of higher price houses are located at different places. They may have other features determine their house prices, such as the type of rooms, and the number of rooms, which are closely related to the size of house. Other lower prices houses are separated almost equally in different areas of Sydney. At the end of this part, we select houses that have the top 10% price. We can clearly figure out what we find above. In conclusion, we find that the best hosts will choose their houses that are close to downtown and seaside area to increase ***Price*** factor.



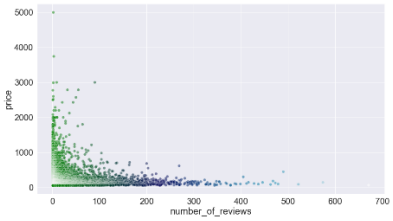
(Visualization 3.2)

In the second part of data mining, we want to find out which type of room has a higher house rental price. Firstly, according to the boxplots showing ‘price’ and ‘logprice’ distributions, most of the prices are between 0 and 1000, however, most of the outliers come from ‘Entire home/apt’. Simultaneously, the ‘logprice’ figure emphasized the difference among four room types. Based on that, we can get the conclusion that the ‘Entire home/apt’ gets a higher average house rental price. On the other side, we can also measure the market share to reveal which room type is more popular in ‘best hosts’ group.



(Visualization 3.2)

By comparing the percentage of all observations' and top 10% observations' room types percentages with respect to ***Price*** in two donut charts, we can find that most of the houses (68.9%) are entire houses, and 29.8% are private rooms. However, for top 10% high price houses, 96.7% of those houses are entire houses. So, the best hosts would be more possible to promote the prices by renting entire homes or apartments. But this could be a problem for new entrants in Airbnb house rental market, because this will require a larger amount of investment.



(Visualization 3.3)

Lastly, we will show the relationship between the number of reviews (popularity) and the house rental prices. From the scatter plot, we can assume that there is a non-linear relationship between these two variables, where higher price houses are hard to get a lot of reviews, while lower price houses with nice facilities or other advantages could have many reviews from the market. Moreover, many houses are struggling to get customers, which might be caused by irrational pricing strategies, unfavorable facilities, inconsistent descriptions, inconvenient neighborhoods, etc. But these are just pure guesses. In addition, we also dug into some special cases, such as observation No.14940 and No. 18295. The first one is the host with highest number of reviews and cheaper house located around Airport and CBD, while the second is the host who owns a luxury beach house but has good feedback from consumers. These two hosts have very detailed information from the description, host about and neighborhood overview, and both are very experienced in the Airbnb market. Consequently, there also exists some extra nominal features can influence the 'best hosts' performances.

In summary, in order to help hosts, make better decisions in the Airbnb house rental market, we strongly suggest investing larger entire houses around the downtown or coastal area in Sydney if they have sufficient funds. However, it also can be considered as a risky move, because it may bring a very huge opportunity cost if the house is not demanded frequently. So, another safer choice for the majority of hosts, they should think about the affordability for themselves and common Airbnb users, then decide the location, room types and what business strategy can increase their properties' popularity.

8. References

Breiman, L. (1996). Stacked Regressions. *Machine Learning 24, 49-64*. https://doi.org/10.1023/A:1018046112532

Domingos, P. (2012). *A few useful things to know about machine learning*. Commun. ACM 55, 10 (October 2012). DOI: https://doi.org/10.1145/2347736.2347755

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning: with applications in R*. Second Edition. New York: Springer. https://www.statlearning.com/

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. (2017). LightGBM: A highly efficient gradient boosting decision tree. *In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 3149–3157.

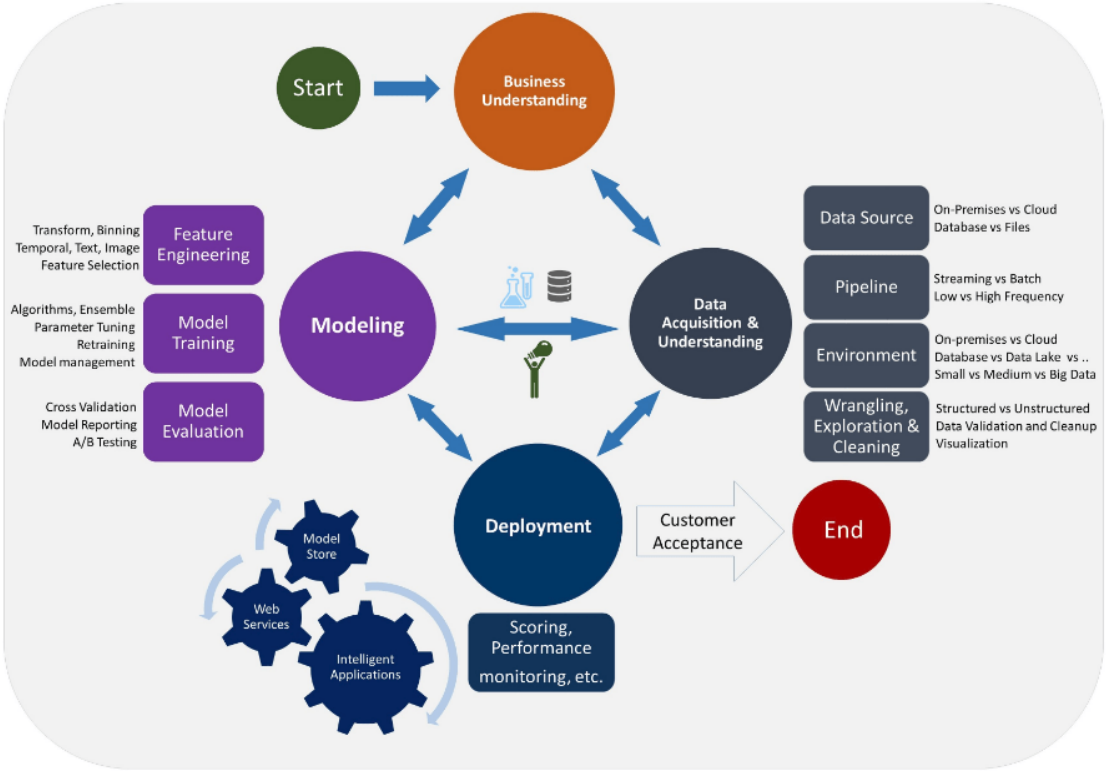
LightGBM. (2021). LightGBM's documentation. Retrieved November 7, 2021, from https://lightgbm.readthedocs.io/en/latest/

Mlxtend. (2021). Mlxtend's documentation. Retrieved November 10, 2021, from http://rasbt.github.io/mlxtend/

QBUS6810. (2021). *Statistical Learning and Data Mining - Lecture: Feature Selection and Regularisation*. University of Sydney Learning Material. Accessed at https://canvas.sydney.edu.au/courses/36330/pages/lecture-feature-selection-and-regularisation?module_item_id=1339070

9. Appendix

- ***Project management report***



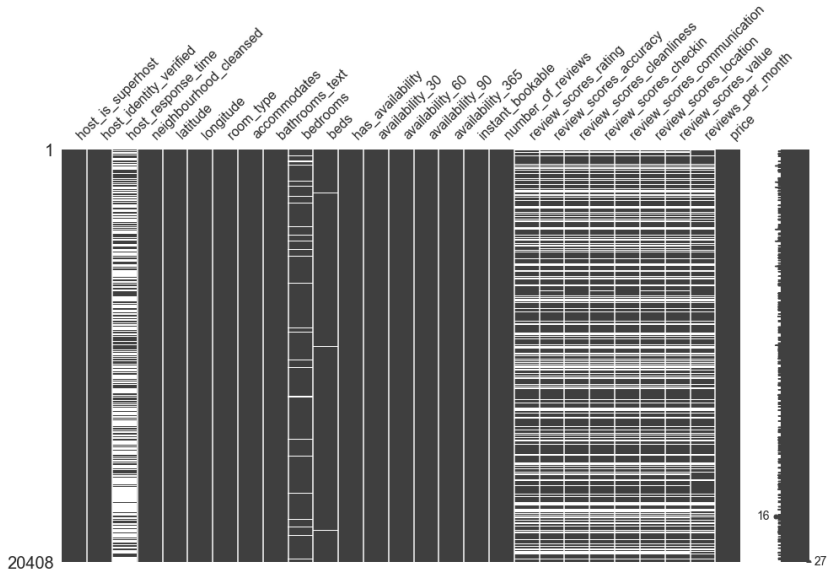
(Microsoft TDSP, Microsoft 2020, from https://neptune.ai/blog/data-science-project-management-in-2021-the-new-guide-for-ml-teams)

In our machine learning project of house rental price regression task, we implemented Microsoft TDSP to guide through the whole process. It contains stages like Business Understanding, Data Acquisition and Understanding, Modeling, Deployment and Customer Acceptance. Firstly, in the business understanding stage, we unpacked the question into two parts that are predicting house prices and providing insights and suggestions for our clients. So, we need data to get a better understanding for the task, and this is why these two steps are connected. After getting many features, we made a choice on what features to use, and what features to keep for later investigation. Before building models, we need some feature engineering techniques to prevent the project from 'garbage in garbage out' results. And we also need to make sure the model performance and accuracy by evaluation based on some metrics. However, before the final deployment, there should be some iterative processes among business understanding, data understanding and modelling, in order to use more meaningful features. Finally, in the deployment phase, we are using the platform Kaggle to test our outputs with many competitors to justify the overall project performances. And then, providing results and recommendations to our clients who are hosts, property managers and real estate investors, and receiving feedback for future updates. In this project management framework, the most useful strategies are thinking and updating iteratively between procedures. Because it is extremely hard to find the only correct one answer in machine learning, therefore we have to try many possibilities and use some method to filter out the proper answers. Outside of the data science project framework, we would say that effective and frequent communication is the key to reaching a comprehensive and concrete understanding in all stages. Since we kept weekly zoom meetings every Sunday night at 20:00 (GMT+8:00), all members could communicate and share findings and task progress routinely and catch up on the missed theories or model details in this project.

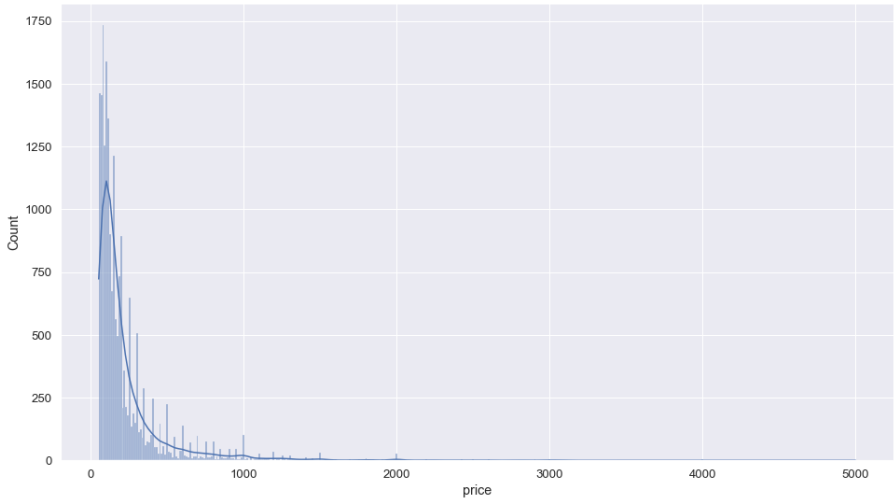
- **Visualizations**

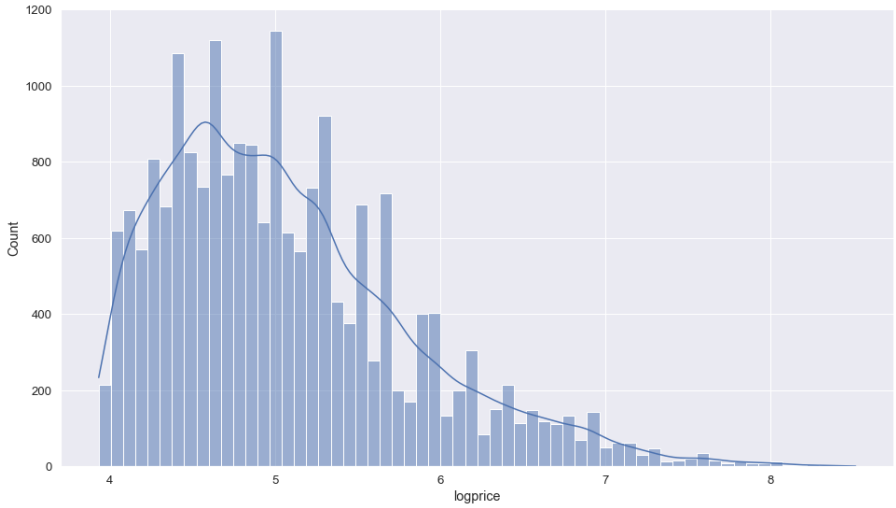
Feature engineering and Exploratory Data Analysis

1.1 Selected Features

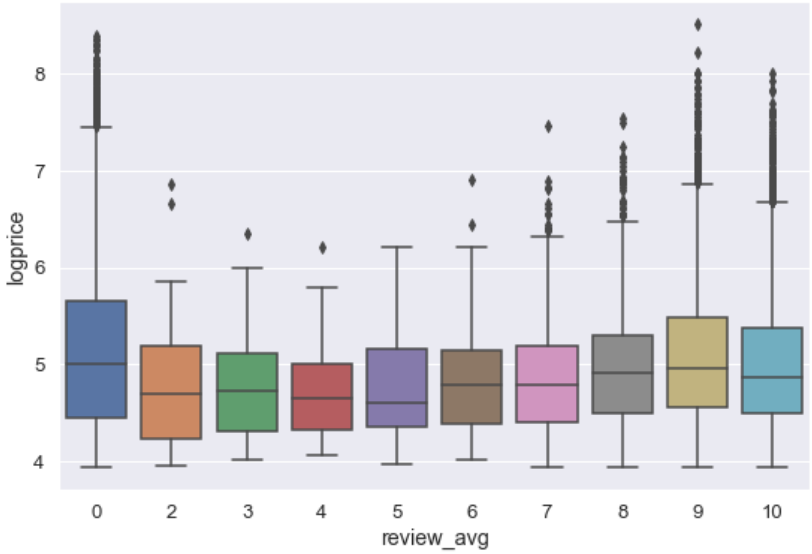


1.2 Response: Price and Logprice

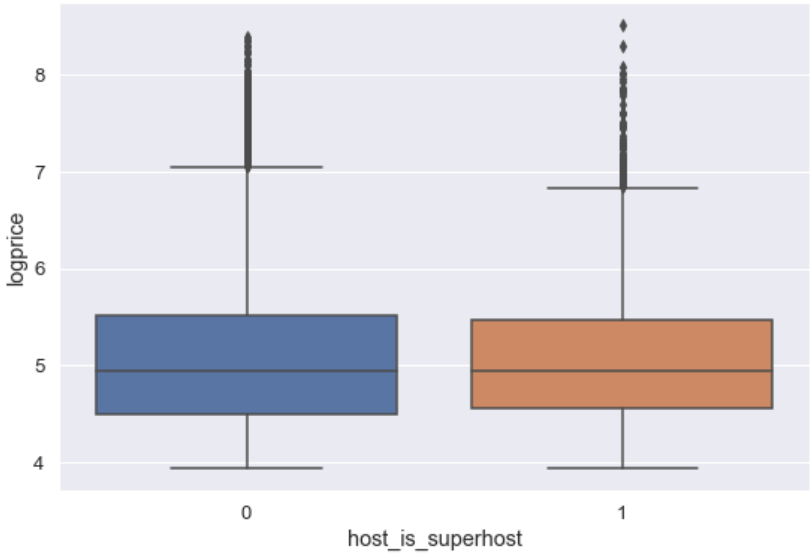


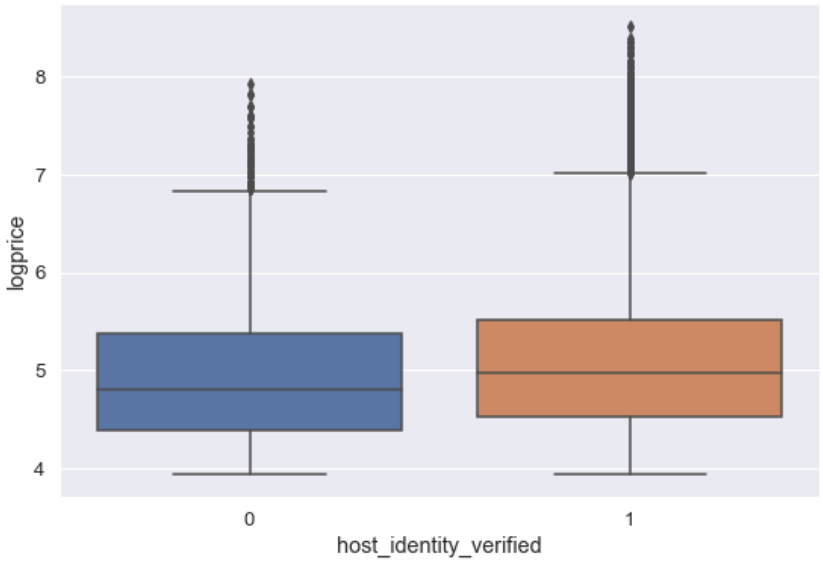


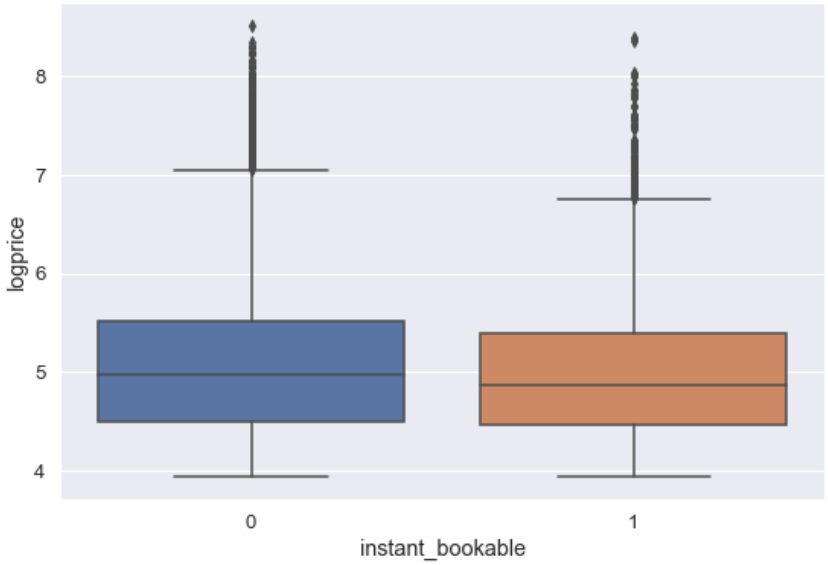
1.3 Reviews

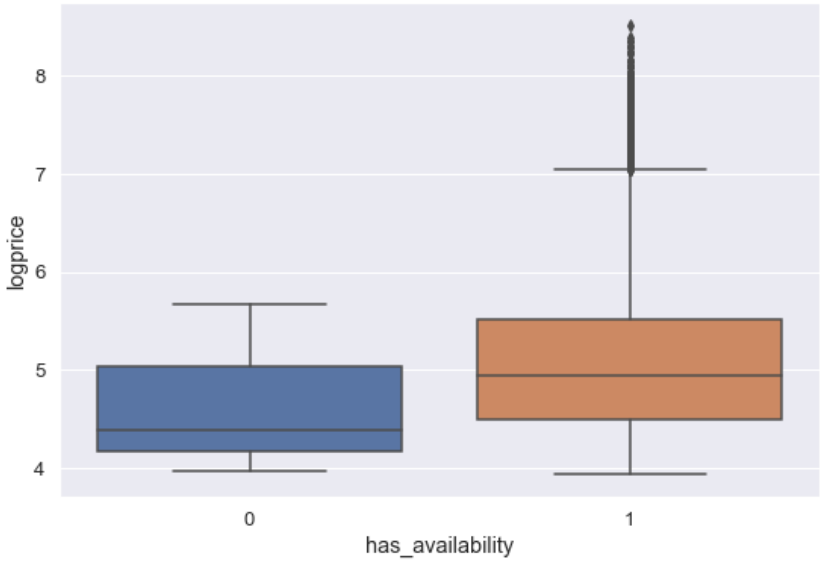


1.4 Original Binary predictors

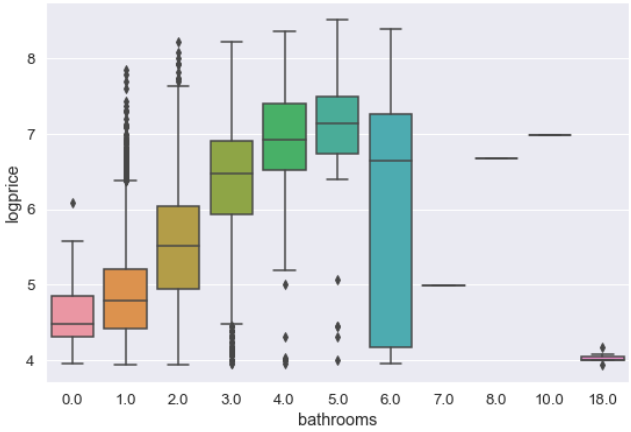


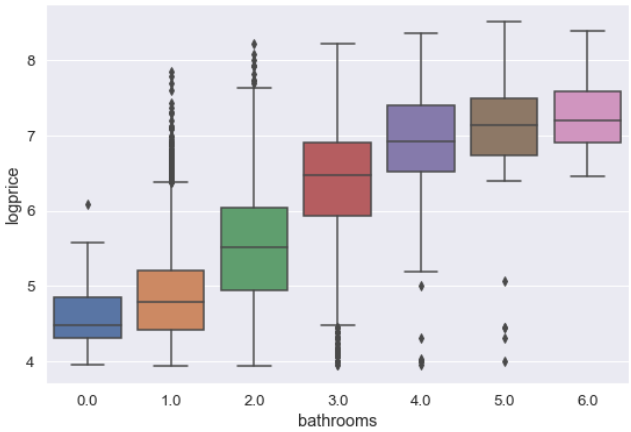

logprice
8
7
6
5
4
0
1
host_identity_verified


logprice
8
7
6
5
4
0
1
instant_bookable

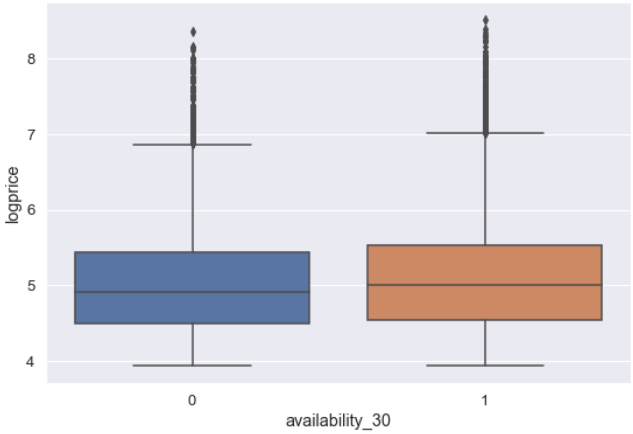

logprice
8
7
6
5
4
0
1
has_availability

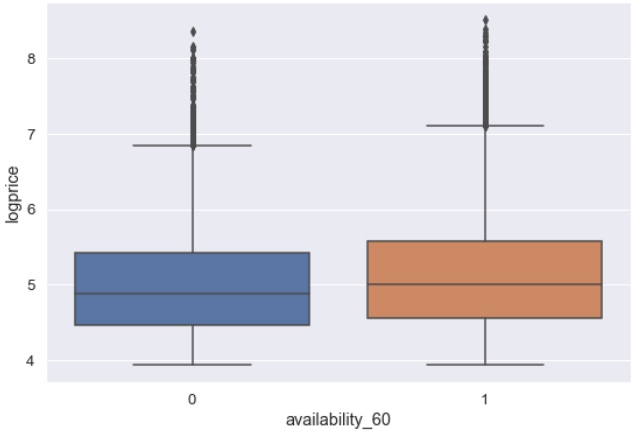
1.5 Bathrooms

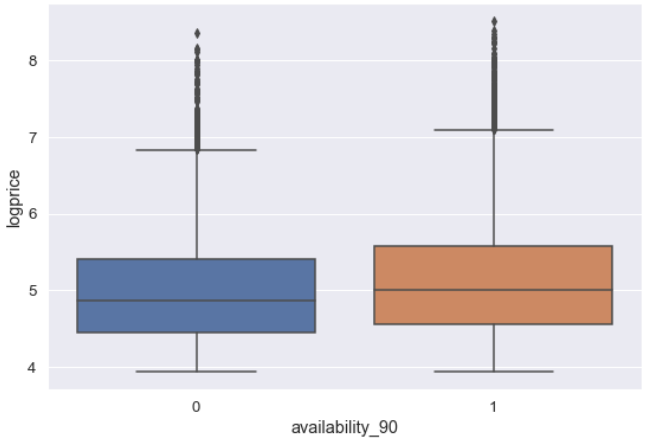


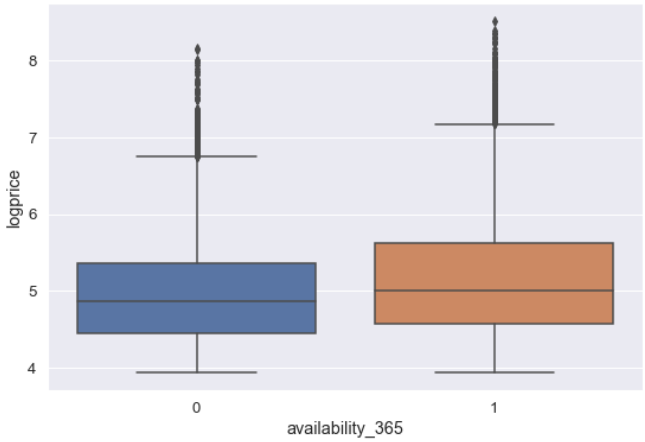


1.6 Availability

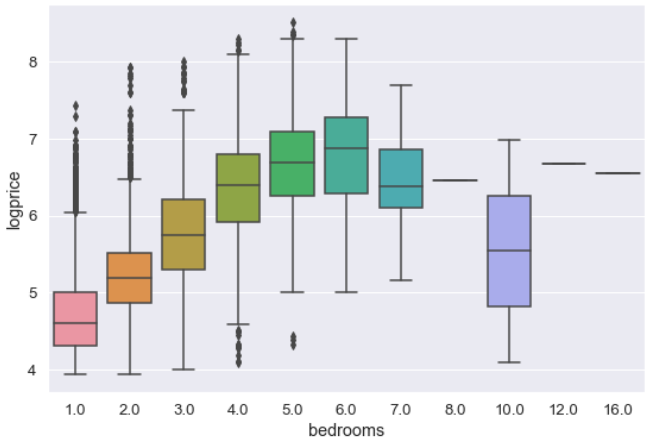


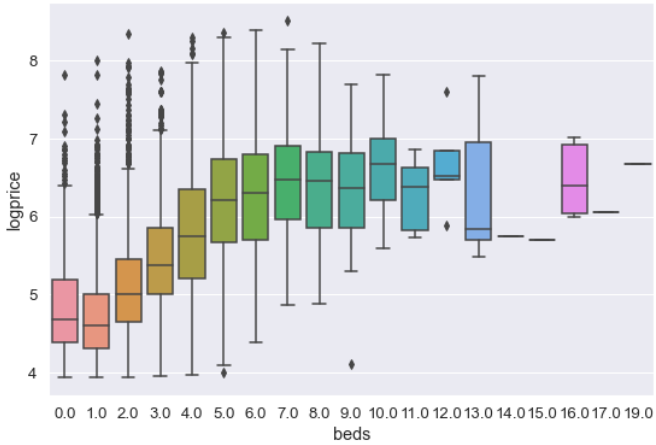


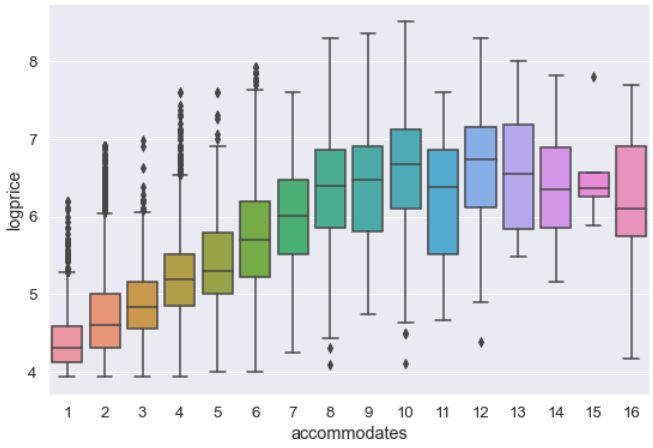


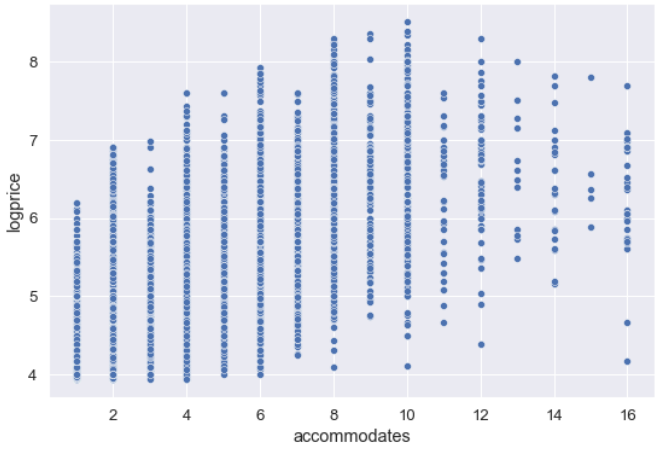


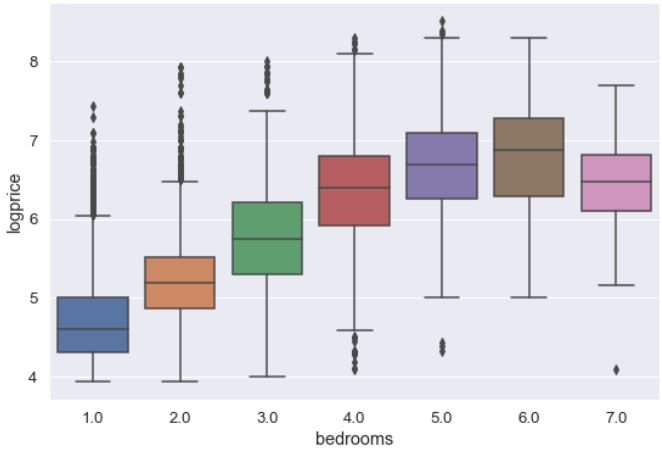
1.7 Accommodates, Bedrooms and Beds

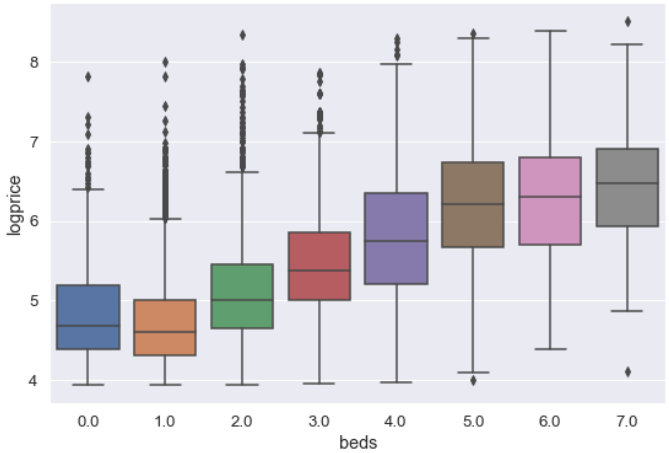


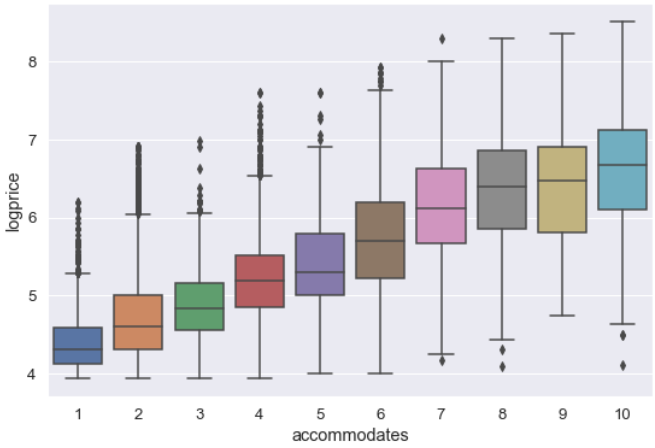



logprice
accommodates

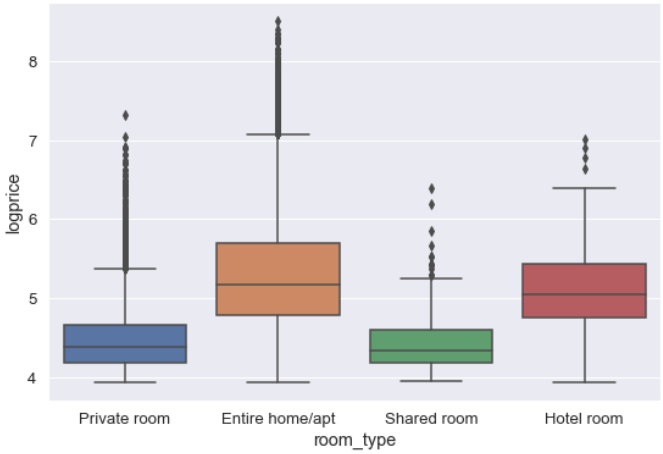

logprice
accommodates

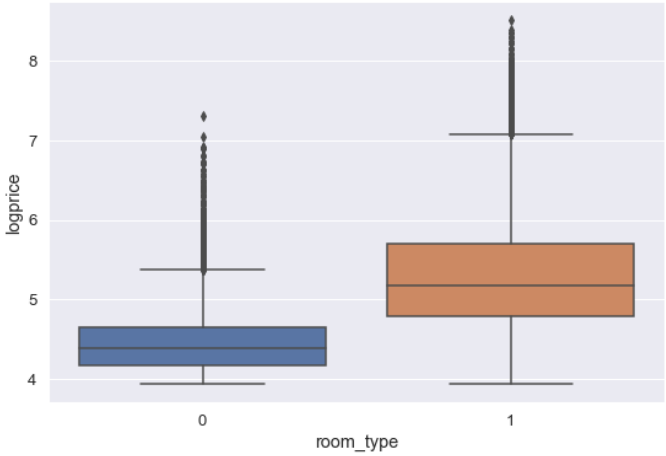

logprice
bedrooms


logprice
beds

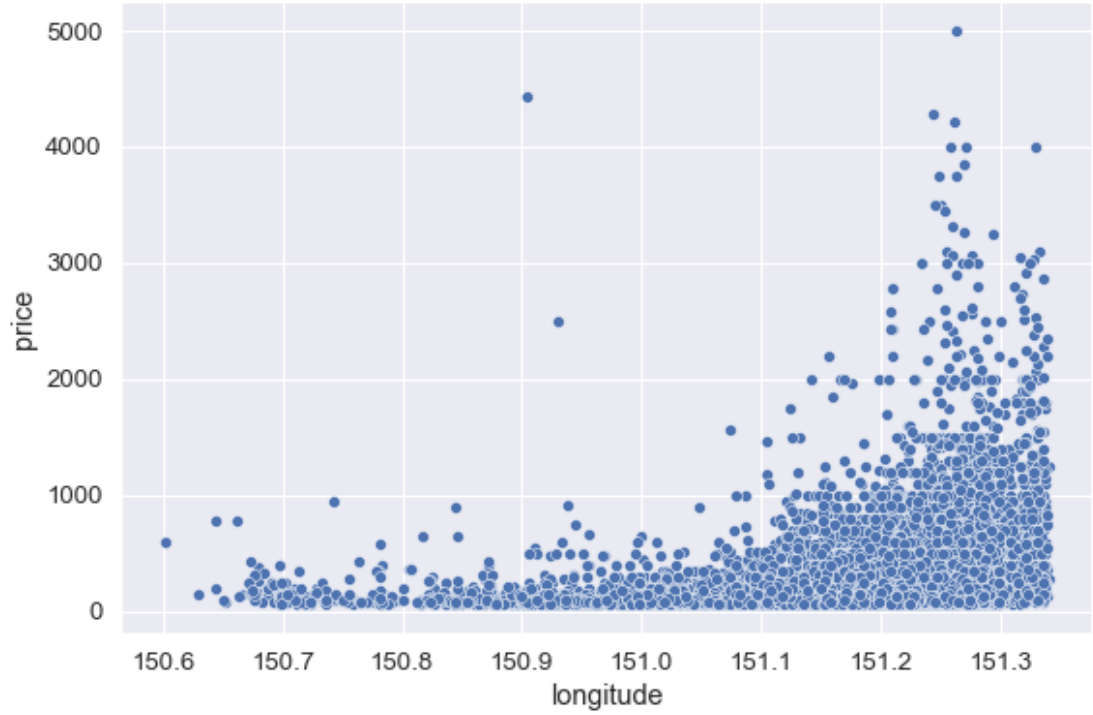


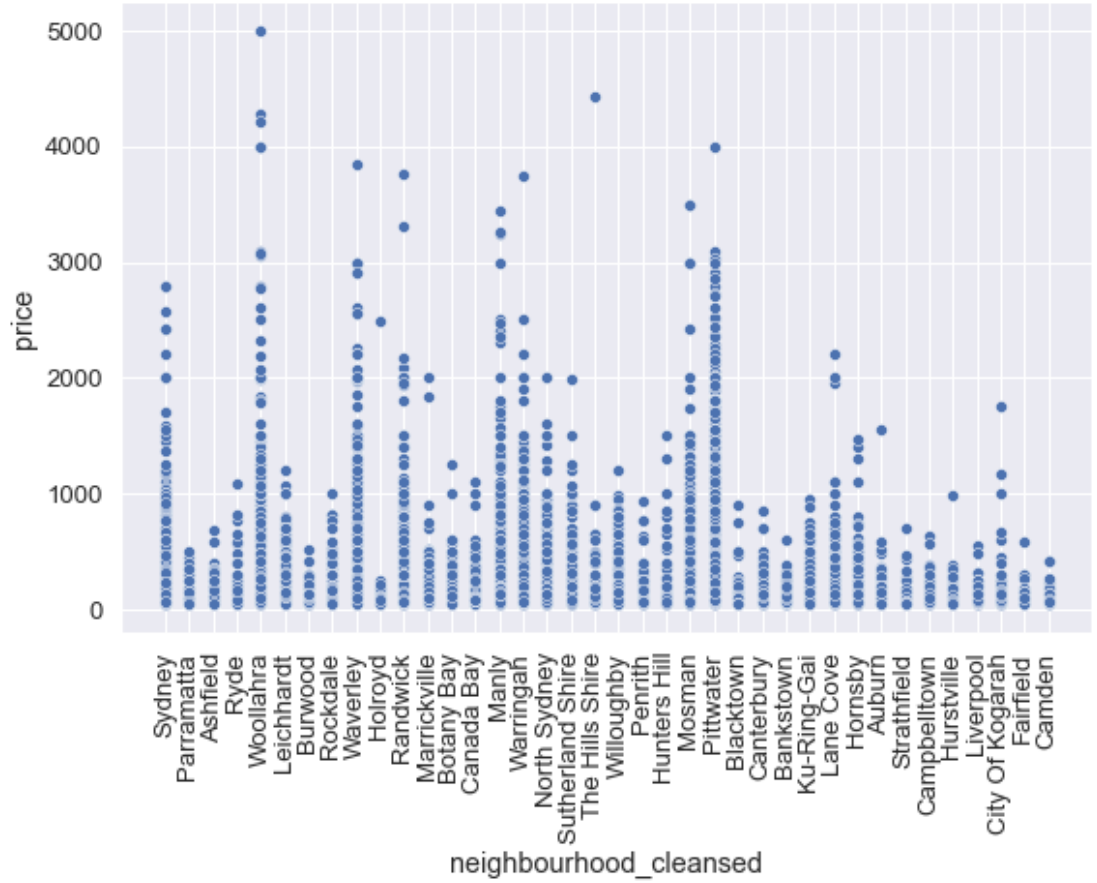
1.8 Room types

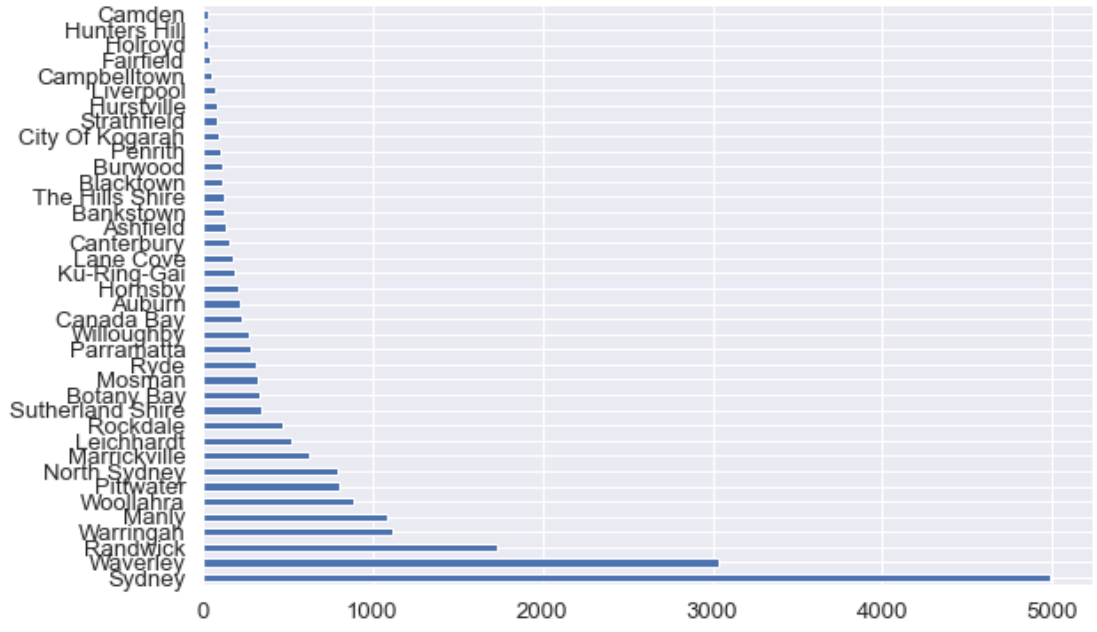




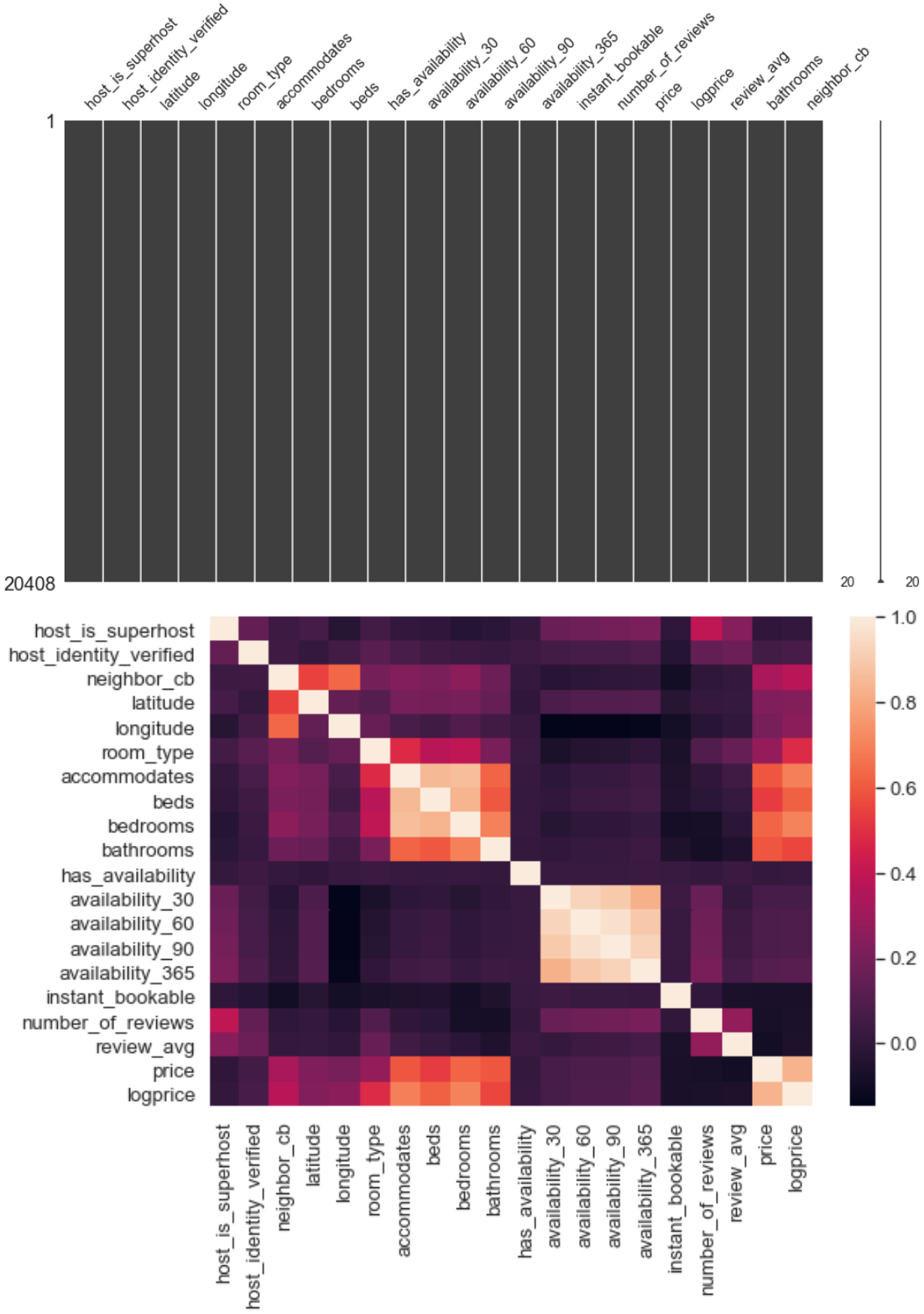
1.9 Geographic location




price
5000
4000
3000
2000
1000
0
Sydney
Parramatta
Ashfield
Ryde
Woollahra
Leichhardt
Burwood
Rockdale
Waverley
Holroyd
Randwick
Marrickville
Botany Bay
Canada Bay
Manly
Warringah
North Sydney
Sutherland Shire
The Hills Shire
Willoughby
Penrith
Hunters Hill
Mosman
Pittwater
Blacktown
Canterbury
Bankstown
Ku-Ring-Gai
Lane Cove
Hornsby
Auburn
Strathfield
Campbelltown
Hurstville
Liverpool
City Of Kogarah
Fairfield
Camden
neighbourhood_cleansed

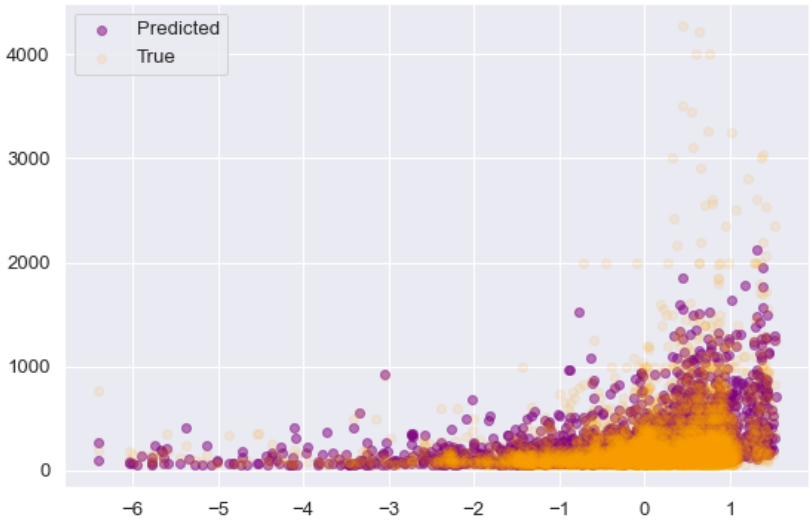

Camden
Hunters Hill
Holroyd
Fairfield
Campbelltown
Liverpool
Hurstville
Strathfield
City Of Kogarah
Penrith
Burwood
Blacktown
The Hills Shire
Bankstown
Ashfield
Canterbury
Lane Cove
Ku-Ring-Gai
Hornsby
Auburn
Canada Bay
Willoughby
Parramatta
Ryde
Mosman
Botany Bay
Sutherland Shire
Rockdale
Leichhardt
Marrickville
North Sydney
Pittwater
Woollahra
Manly
Warringah
Randwick
Waverley
Sydney
0
1000
2000
3000
4000
5000

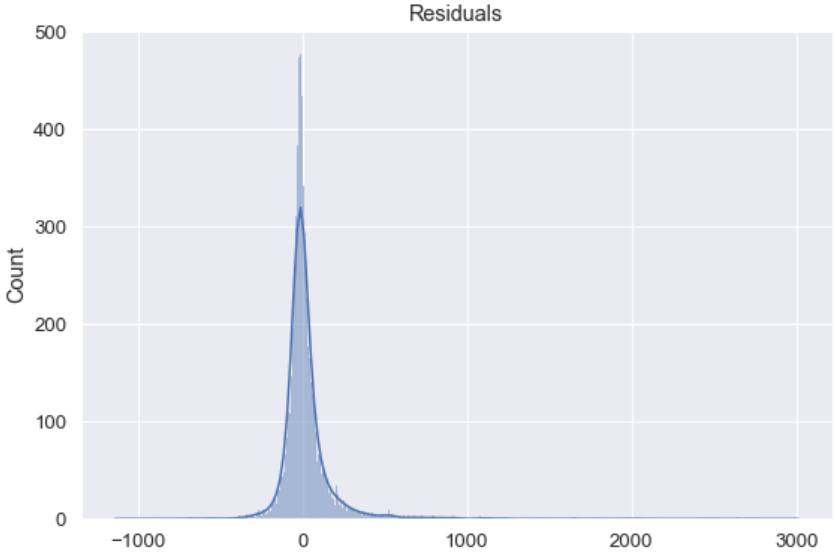
1.10 Summary before model building



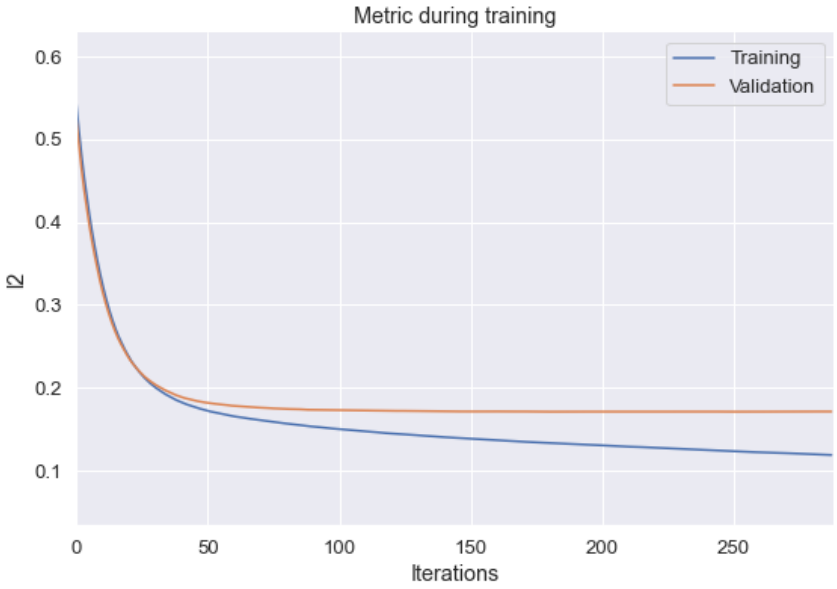
Methodology

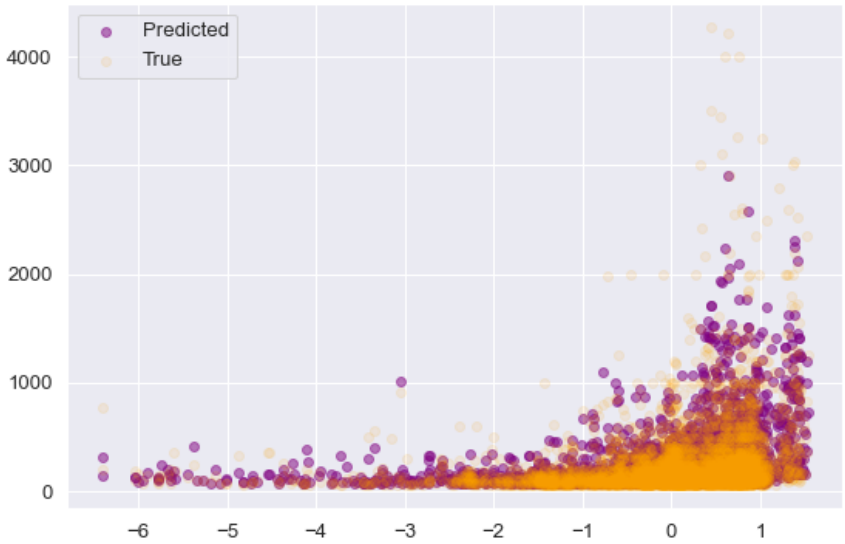
2.1 Linear Regression

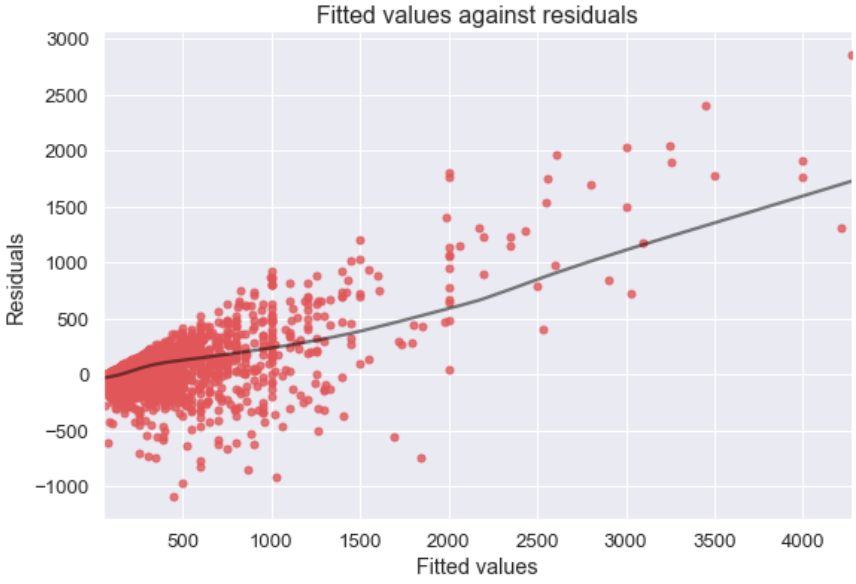




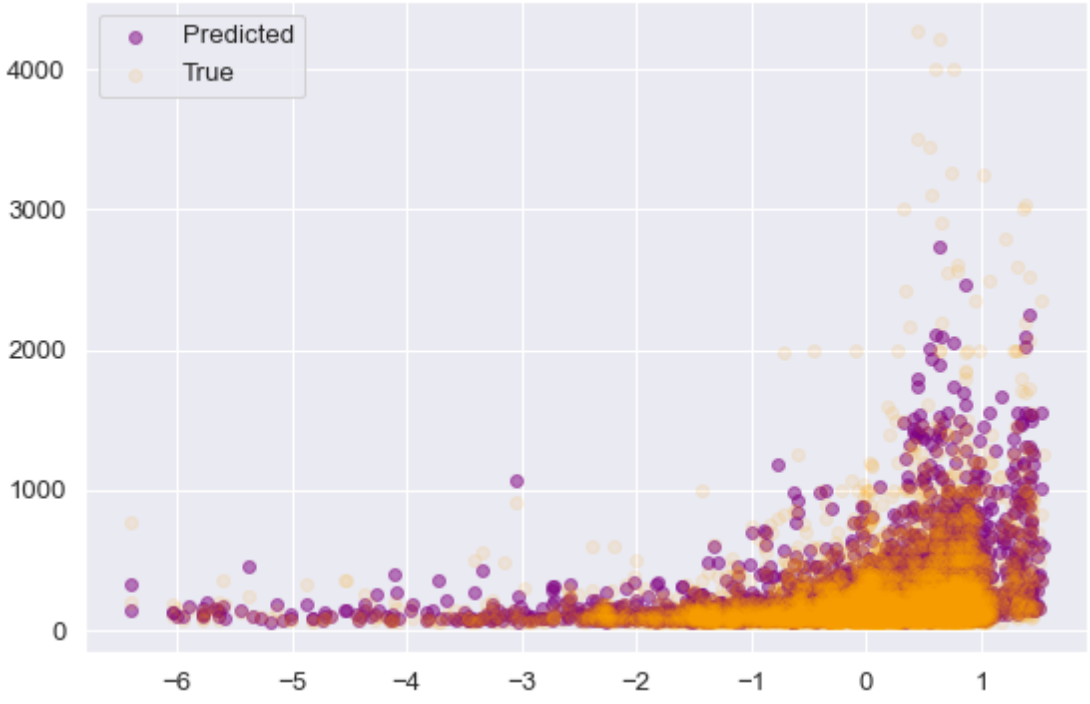
2.2 Light GBM

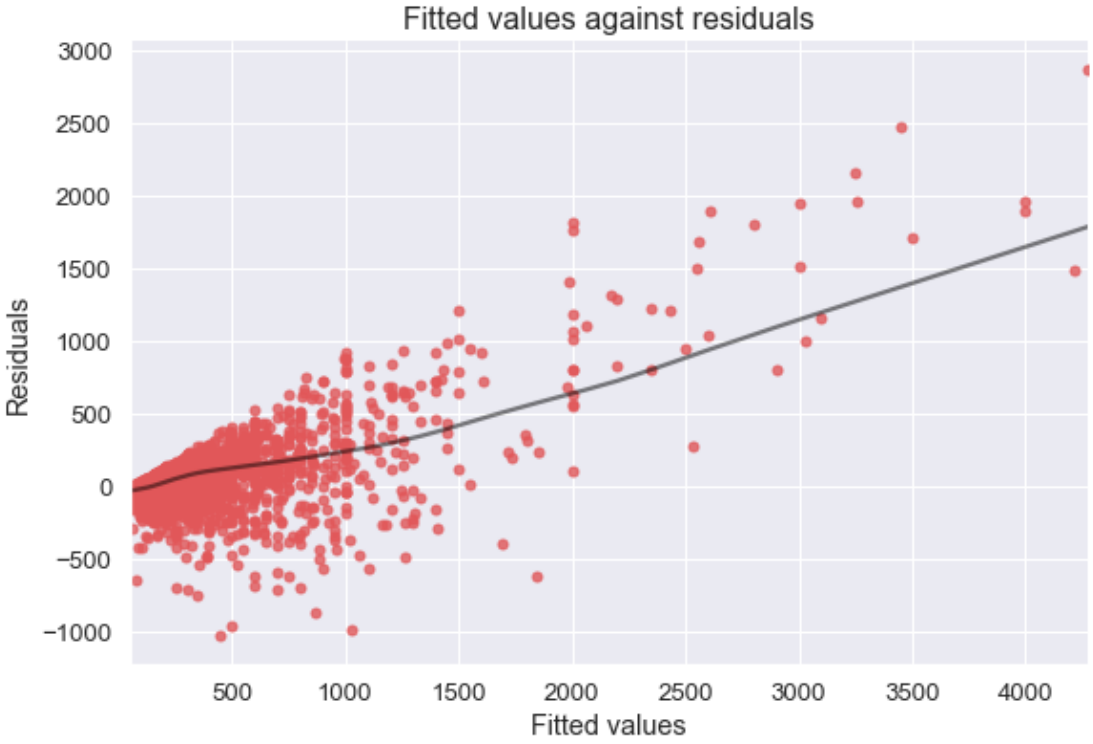

Metric during training
Training
Validation
l2
0.6
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
Iterations


Predicted
True
4000
3000
2000
1000
0
−6
−5
−4
−3
−2
−1
0
1


Fitted values against residuals
Residuals
3000
2500
2000
1500
1000
500
0
−500
−1000
500
1000
1500
2000
2500
3000
3500
4000
Fitted values

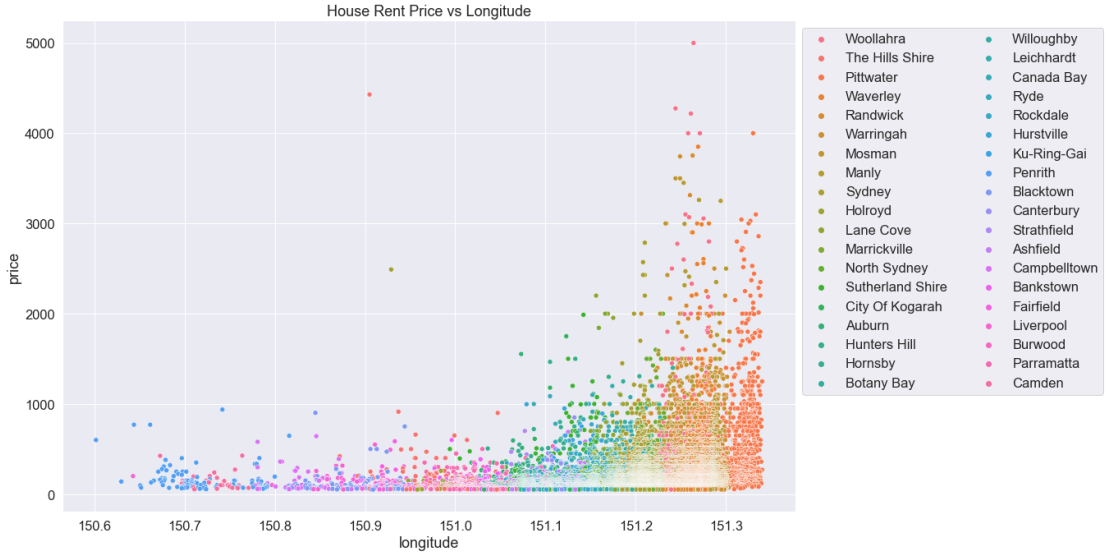
2.3 Model Stacking

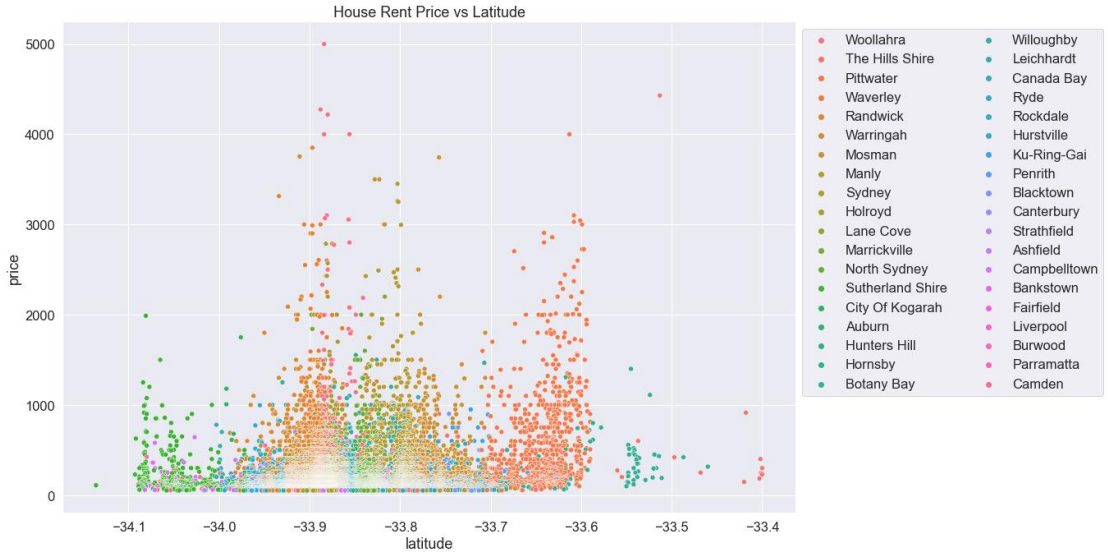




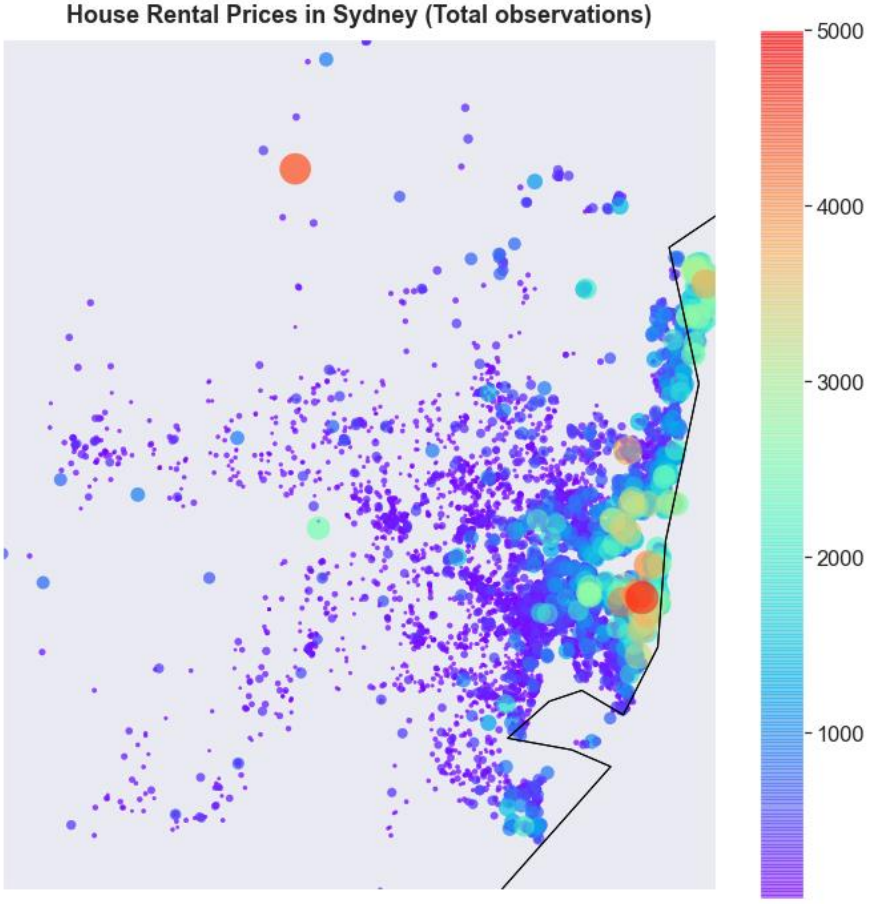
Data mining

3.1 Geographic locations



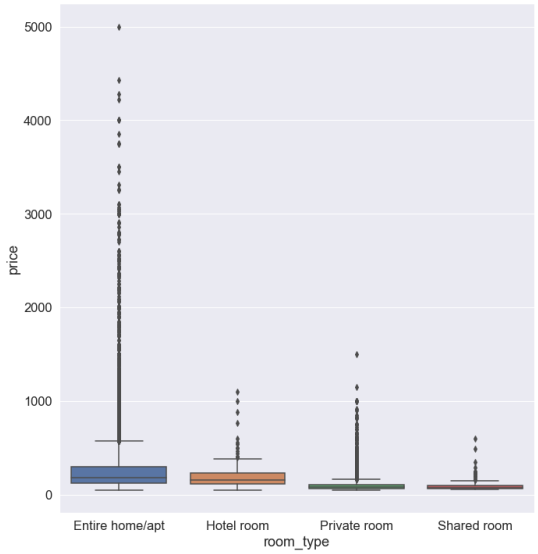


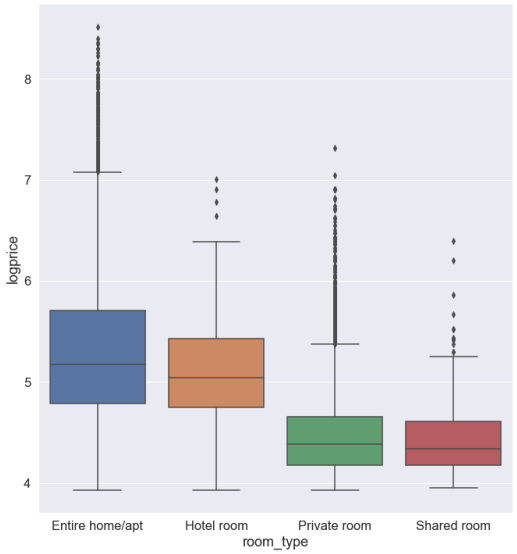
House Rental Prices in Sydney (Total observations)

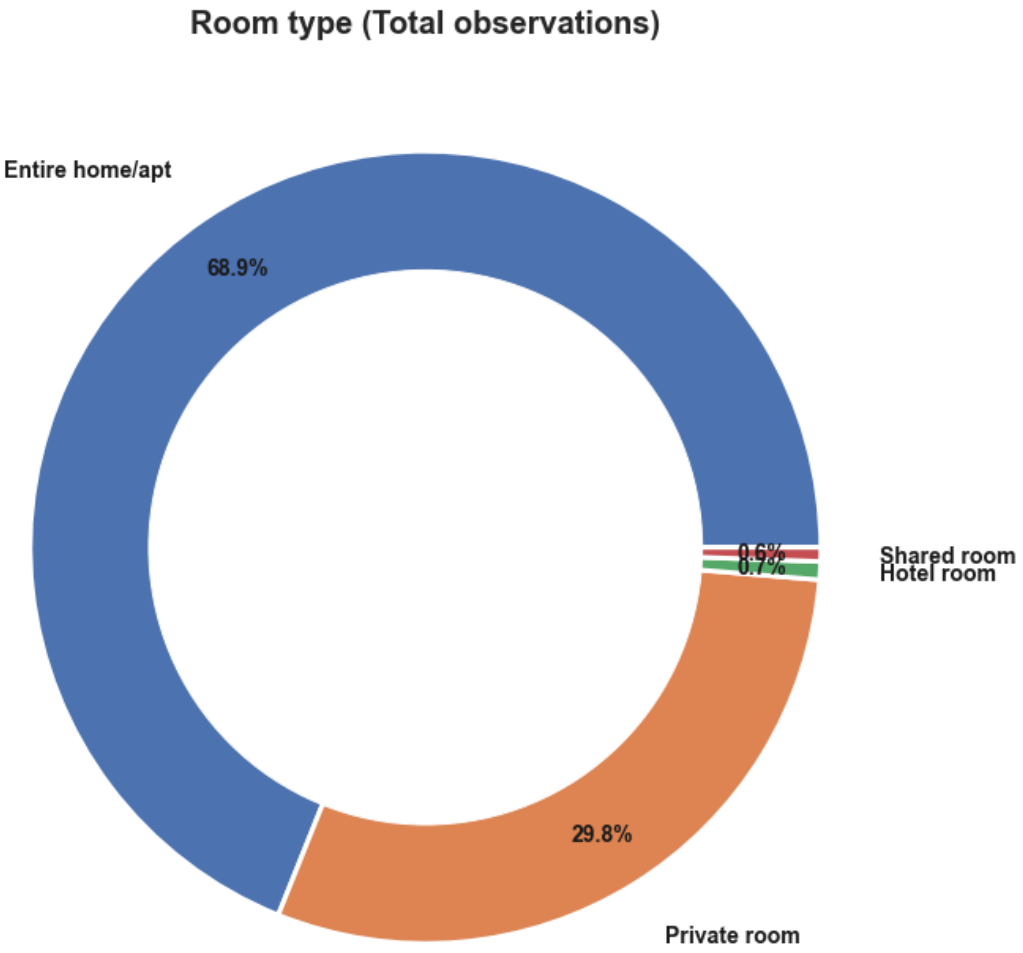


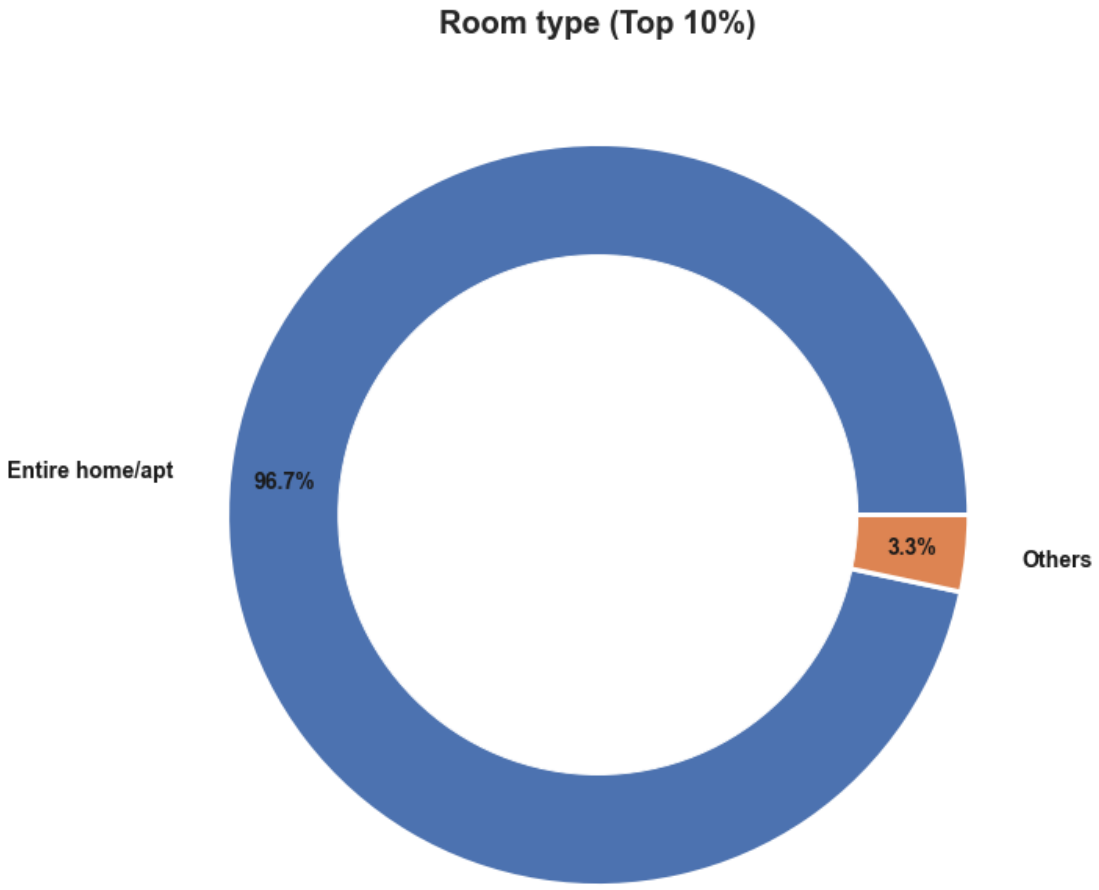
House Rental Prices in Sydney (Top 10%)

3.2 Room types









3.3 Popularity (No. of reviews)