THE UNIVERSITY OF SYDNEY

QBUS 3820 Machine Learning and Data Mining in Business

Group Assignment

Group 12

SID: 470056172

SID: 470134067

SID: 480101240

SID: 500375833

Table of Contents

QBUS3820 Group assignment

Introduction

Based on the data from Airbnb rentals in Sydney, we are going to construct predictive models for the daily rental prices and obtain some useful insights to help stakeholders in this market. This report develops by discussing the supervised learning process first, explaining and justifying our models to clients, and finally discussing some of the interesting facts that we have found.

We use six models to predict the price: Linear Regression, Regression Tree, Bagging, Random Forests, XG Boosting, and Model Stacking. Based on scores from Kaggle competition (validation set), the model that perform best is the stacking model, which naturally, is be used as our final model for the further model evaluation. In addition, the regression model is utilized to provide three practical suggestions for our clients, which mainly focuses on how to improve their revenues.

Data processing and exploratory data analysis

Data processing

In order to benefit our further exploratory data analysis, we need to do some data processing to make our predictors interpretable and visualizable.

Firstly, we checked the basic information of the dataset. We found that there are some columns such as ‘summary’ and ‘description’ which include various and complex expressions, leading it hard to extract information from these folumns. Thus, we decide to exclude such variables. However, it does not mean that this kind of variables are not useful, if such kind of inputs could be organized structurally, we may also gain some useful insights from them.

Secondly, we noticed that the ‘host_since’ variable is stored as a date structure and cannot directly and quickly convey information, thus a new column ‘host_years’ was created, indicating how many years the hosts have rent their houses out, which is simply calculated by using the current year 2020 minus host since years.

Then, we checked if there existed some missing values and the types of each of those variables. The result shows that we have 43 numerical predictors and 21 categorical predictors that could be used for predications. Basing on these data types, we will do exploratory data analysis for numerical predictors and categorical predictors respectively.

Exploratory data analysis

Part A: EDA on response variable

Firstly, we conducted exploratory data analysis for the response variable daily rental price.

The daily rental prices have a mean of $217.97 per day and it should be remarkable that more than 75% of the data are below $231.00. In addition, from the figure below, it could be observed that the distribution of daily price in Airbnb is extremely right skewed with some significantly expensive houses being rented, driving the whole mean larger than the median value. Such pattern indicates that we should do the log transformation for our benchmark model -- linear regression, so that we could improve the overall model performance.

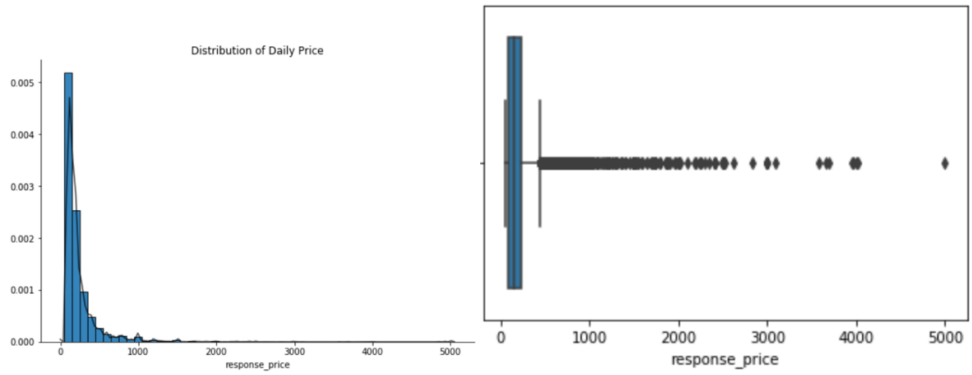


Figure 1 Distribution of response variable

Part B: EDA on numerical predictors

Next, we conducted exploratory data analysis for numerical predictors. On the first stage, the correlation of them with daily rental prices are being checked. Most of the variable, in fact, has a relatively weak correlation with the response variable. Only a few predictors, such as the number of bedrooms and accommodates are highly correlated with the price per night. Then, in order to conduct deeper and accurate analysis, we use the regression plot to visualize their relationships. Here, we select four representative variables among them to discuss in detail.

As figure 2 illustrates, the first representative variable is the longitude. It is quite interesting to see that when the longitude is lower than the 151.1, almost all the daily rental prices are quite low, while once the location of this house is above that longitude, the number of expensive houses start increasing. Such complex pattern can also be discovered from other variables such as the number of reviews but with a different trend, suggesting that more advanced model is required to capture these complex relationships. Then, the next variable deserved to mention is the square feet. Although in the correlation table, it shows a relatively strong and positive relationship with the rental price, from the plot, we could see that it has a serious issue of lacking the data, which may make the square feet become a less powerful or even useless predictors in this case and should be carefully considered in the feature engineering process.

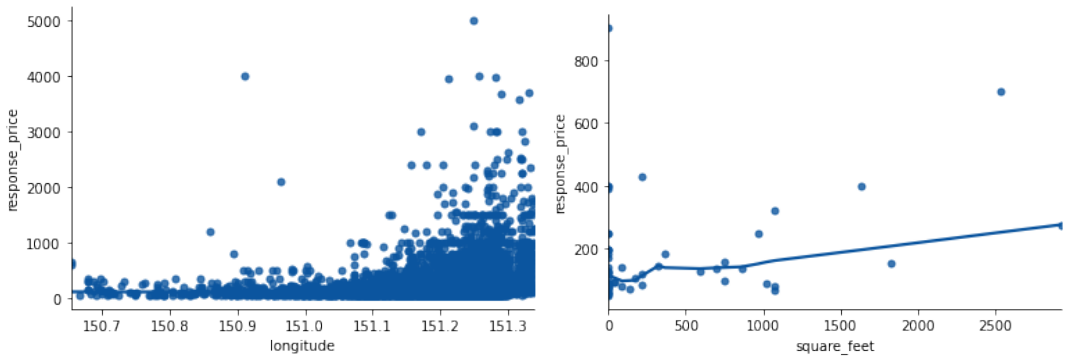


Figure 2 Representative numerical variables with daily price

The next important variable the requires explanation is the number of beds. It is reasonable to judge that when there are more beds in that house, the price per night will naturally be higher. This relationship is quite significant at the beginning, however, when the bed number increases from three to four, the average rental price shows a decreasing trend. Afterwards, it increases again but the price is not that sensitive to the number of beds than before. The pattern like this may indicate that the linear splines model could be hard to see whether we could get a good model performance. The final representative is the host years, which generates a straight line with almost zero slope suggests that the per night rental price in Airbnb may not be correlated with how many years the host has rented their house. Such kind of features, actually, is quite common to see in our dataset, but we cannot imprudently drop them because although the variable independently may not be that useful, it could have some significant effect when combining with other inputs. Especially if the tree-based model such as the regression tree, which could approximate complex interactions, is considered. We should let the model itself to choose the inputs that being used.

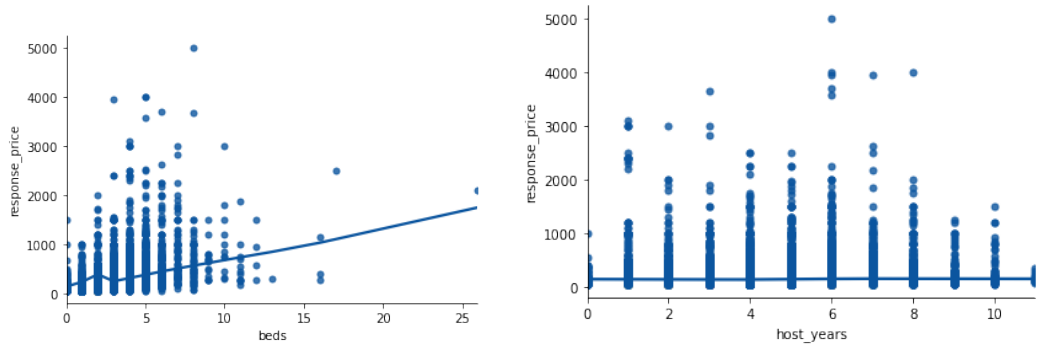


Figure3 Representative numerical variables with daily price

Part C: EDA on categorical predictors

At last, the exploratory data analysis for categorical predictors is being conducted. Due to the existence of some extremely expensive per night rental prices, the plots are being 'pressed' and it is hard to directly use the boxplot observe the differences between each class of the categorical variables. Thus, just for visualization purpose, we extract the rental prices that are less that $300 per night, in this range, we capture more than 80% of the total

data. Again, under the constraint of too many predictors, we will still choose to discuss some remarkable ones and to benefit our further engineering process, we roughly divided the whole categorical dataset into four different types. The first kind of variable's plot is clear and easy for us to capture some information. One typically example is the room type. As the first figure below illustrates, the 'room types' consists four different categories and each category has some identifiable difference with others. Also, the number of observations is sufficiently enough in each class, thus in feature engineering process, for such kind of variable, we may not need to do something specially expect creating dummy variables for it. Then the next type is represented by bed types. Although from the graph below, it could be observed that the house with real bed has a quite higher rental price per night than the other groups, it should be noticed that the observations in group couch is so sparse and creating a separately dummy variable will be meaningless. Thus, we should merge them with other classes that are also sparse or quite similar.

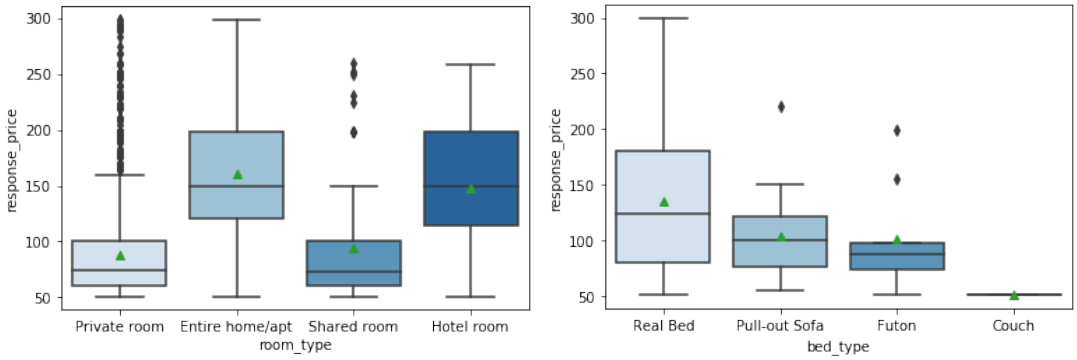


Figure4 Representative categorical variables with daily price

The third type of categorical variable is rather messy. It is obvious to see that there are too many classes in the property type, which certainly will affect our model performance. Therefore, we decide to merge categories based on their mean value of per night price and the number of observations in each class. Finally, the characteristic of the last type of features is the number of classes, and the observations in each class is quite reasonable. However, the differences of mean, median and range between them are not so obvious. For such variables, still, we can not simply remove them and should let the model itself decide whether the predictors are useful or not.

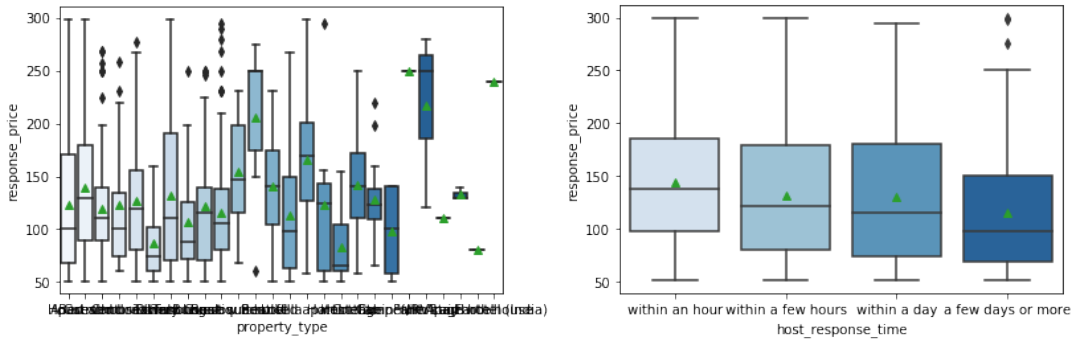


Figure5 Representative categorical variables with daily price

Lastly, if we go back to look at the whole picture with all the values of rental pieces included, it is remarkable that some inputs only have one class such as the variable 'is business travel ready'. If all the observations we have belong to the same category, this predictor may not play a significant role.

Another interesting phenomenon to be noticed is that most of the rooms with extremely high price do not have the requirement to provide guest profile pictures or phone verification, while for the rental houses that are lower than $1000 per night, there are two types expressed as true and false.

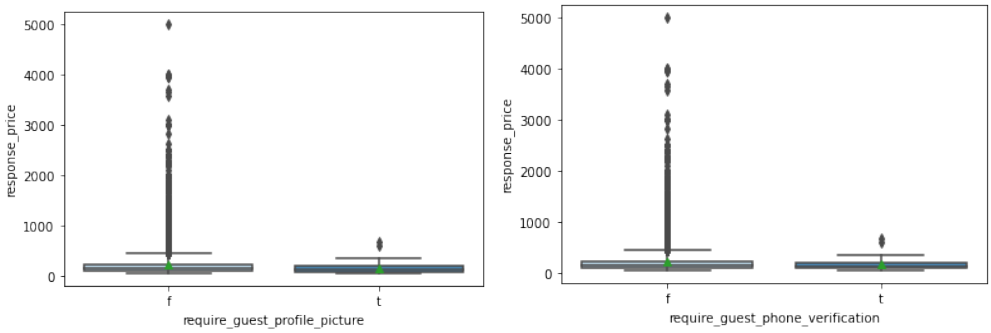


Figure6 Representative categorical variables with daily price

Feature engineering

The previous exploratory data analysis has informed and provided some directions for the feature engineering process. Feature engineering is the process that could contribute to good model performance and it should be noticed that different models require different kinds of feature engineering to adapt to its learning algorithm. In practice, we basically develop two sets of data to do two slightly different feature processes. One is for the linear model while the other is for some tree-based models. To make our steps clearer and easier to explain, we will introduce the process through the following structure:

(1) Response variable transformation
(2) Handle missing values
(3) Merge sparse or similar category
(4) Deal with outliers
(5) Create dummy variables
(6) Drop not useful columns

Task 1 Response variable transformation

As discussed in exploratory data analysis, the daily rental price in Airbnb has an extremely long right tail, and to improve the performance of our benchmark model – linear regression,

we do the log-transformation to reduce the skewness in errors. As is depicted in the graph below, even after transforming, the response variable is still right skewed and non-normal, raising the concern that there may be potentially no good performance of the linear model.

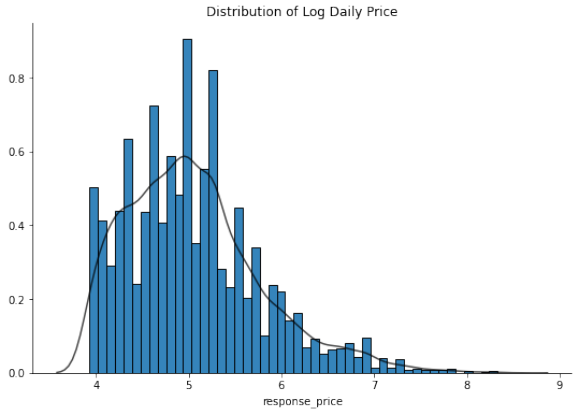


Figure7 Log transformation for rental price

Task 2 Handle Missing value

One of the limitations of this part is that although we use the distribution plot of each variables that are derived from EDA (see Appendix in detail) to judge how to fill in the null values based on some criteria, the combining of data driven and subject opinions may still result in some inaccuracy and bias.

We checked and handled the missing value for numerical data and categorical data respectively.

Part A: numerical predictors

When handling numerical features, basically we used the method of filling in them with the mode, mean, or zero.

The criteria of filling mode are that:

(1) The data type of these predictors is integer.
(2) The numerical values of these predictors are likely to concentrate on some certain amount.
(3) The missing value does not mean that the corresponding characteristic not apply to that Airbnb listings.

Thus, we selected the inputs that satisfy these requirements and summarized as the table below. Note that in terms of missing values for these predictors in the test set, we still used the train mode to fit them, as we always derived information from the train dataset rather than the test or validation set.

Integral numerical predictors	mode
bathrooms	1.0
bedrooms	1.0
beds	1.0
review_scores_accuracy	10.0
review_scores_cleanliness	10.0
review_scores_checkin	10.0
review_scores_communication	10.0
review_scores_location	10.0
host_response_rate	100.0
host_acceptance_rate’	100.0

Next, the criteria of filling a mean value is that:

(1) The values of these predictors are acceptable for fractional part.
(2) The numerical values of these predictors are quite widely ranged and unlikely to concentrate on some certain amount.
(3) The missing value does not mean that the corresponding characteristic not apply to that Airbnb listings.

Only the ‘review_scores_rating’ satisfied these three criteria and we use train mean to fill in both train and test dataset.

Finally, we select the variables that are reasonable to have zero values. For example, the missing values in weekly discount may indicate that this host does not provide any weekly discount for renters, and such logic also apples tot monthly discount, security deposit percentages on the nightly rate, cleaning fee percentage on the nightly rate and the reviews per month. Thus, we filled in those variables with zero.

Part B: categorical predictors

The fillings of categorical predictors basically follow the same logic as above. We filled the missing values of ‘host_response_time’ with its modes since we considered them as lack of information rather than do not apply and we assume that the most hosts’ response time are quite similar with each other or do not vary from each other to a large extent.

Task 3 Merge sparse or similar categories

The main issue that should be carefully considered is that some sparse categories occur only in the train set while the other appear only in the test set. Thus, when we merge the sparse categories, we should consider the category of train and test and the same time or otherwise, the mismatch of dimension will lead to failure in making predications for test set.

The variable selected to be merged are based on the information derived from EDA, and the merge process consists of the following steps:

(1) For each variable, we print each category's train mean, train median and number of observations.
(2) Keep the category with a large number of observations.
(3) Merge small categories basing on their train mean and median values.
(4) Check the category in test set to see whether there are some missing classes.
(5) Merge the extra category from test set

Task 4 Deal with outliers

Basing on the distribution plot of each numerical variable, we noticed that the security deposit percentages on the nightly rate and cleaning fee percentage on the nightly rate are significantly not consist with the rule defined by Airbnb (2020), and the values in both categories on average, are 100 times and 10 times respectively larger than the normal range of the charging fees. Therefore, we initially guess that there may be some decimal issues here. Then in practice, we had to let the data indicate whether the change of decimal places is reasonable, so we prepared two set of data, one without change and the other with changes the decimal places. Then we kept other variables constant, fit the linear regression model and put it in the validation set and found that actually, the one with adjustment gives us a relative better score. Hence, we keep this adjustment, but this is a roughly solution and more information is needed to correctly fix these errors.

In addition, the other potential issue is that the variables -- weekly discount and monthly discount have some negative values, which may probably due to different representation of discounts by different hosts, so we unified all the values to be positive for both train and test sets.

Task 5 Create dummy variables

Creating dummy variables are quite simple but one thing to notice is that for our benchmark model -- linear regression, we dropped the first column to avoid the multicollinearity issue while for the other methods used, do not have such concern, and we should let the model itself choose the most proper predictors. Hence, we did not drop the first column for these advanced models.

Task 6 Drop not useful predictors

Although we know that it is usually better if the variables are naturally selected by learning algorithm, here, some predictors are unacceptable and should be handle manually. The dropping criteria are shown as below:

(1) There are more than 90% of missing values such as the 'square feet', implying that we cannot find an appropriate fill in values.
(2) The predictors have other alternative better predictors in this dataset. For instance, when we tried to extract information from one of the text data -- host location, we may find the numerical variables – longitude and latitude are more adaptable to the learning algorithm in practice.
(3) The categorical variable that only contains one category.

Methodology

The initial six models we selected are the simple linear model, regression tree, bagging, random forest, XG boosting and model stacking. Other models that are covered in this unit such us KNN are not considered because the number of predictors is large relative to our training sample size, which in turn, is easily subject to the curse of dimensionality.

In this part, basing on the validation set score on the Kaggle, we selected the three models - Random Forest, the Model Stacking, and Regression tree to discuss in detail.

Model 1 Random Forest

Part A: Rational of choosing the model and how they make sense for the data

The rational of fitting the random forest model is that firstly, it is a tree-based model and we can take advantages of the trees. Considering our dataset, we summarized three main points about how the tree-based models benefit our fitting process.

(1) There are some outliers in the predictors space such as the reviews per month. Additionally, the distributions of them are quite skewed. When we used the tree based, it is totally robust to these outliers and we do not need to do any feature engineering for numerical predictors to correct the skewness.
(2) When we predict the rental house prices, there should be some significant interaction effects. For example, the marginal effect of the number of bedrooms on per night rental prices could depend on the location of that house. The tree-based method naturally handles these complex interactions.
(3) Another important issue is that we have too many predictors in our dataset and we expect to let the model itself to choose the proper variable for us. The tree-based method will automatically perform the variable selections by recursive binary splitting and handle this issue really well.

Building on the tree-based method, the random forest even further improves it by reducing the instability and increasing the predictive accuracy. Specially, it averages many trees so that the model variance will be reduced. Then, it also adds a tweak to decorrelates the trees, before splitting each node, we only considered a sample of predictors to be candidate variables, which reduces the correlation of each trees and hence reduce variances. Although such method may increase some bias, hopefully, the variance reduced could offset the bias increased and hence improve the generalization ability.

Part B: How the model is fit

In terms of fitting random forest, the most important thing is to choose the tuning parameters. The first hyperparameters we need to think about is the number of trees that growths. Increasing the number of trees will not lead to overfitting as there are more tress

that could be averaged, and the variance will tend to decrease. Therefore, we set a relatively wide range for the number of trees. Next parameters we selected is the max depth, this one, however, is different from the number of trees because when we increase the depth, the model will tend to be overfitting and capture some unnecessary small variations. Thus, we did not choose its range to be some relatively small values. After initially choosing the tuning parameters, considering the computational cost, we used random search method to find the best hyperparameters in this range. Next, we put our predictions into the validation set and kept changing the range of the hyperparameters in order to improve our model. The number of trees selected by random search is 101 and the maximum depth is 15

Part C: Interpretations of the models

Although Random forest improves the predictive ability comparing with the regression tree, it compromises the interpretability. Therefore, in order to roughly interpret the result of our models, the variable importance plot is being produced, which reflects how one predictor helps to reduce training error and it is measured by adding up the total decrease in the mean squared error by using that single predictor, and then average over all the trees. From the figure 8, we could observe that the number of bedrooms plays a far more significant role in predicting the rental price per night in Airbnb, which is followed by the cleaning fee percentage on the nightly rate and the number of bathrooms.

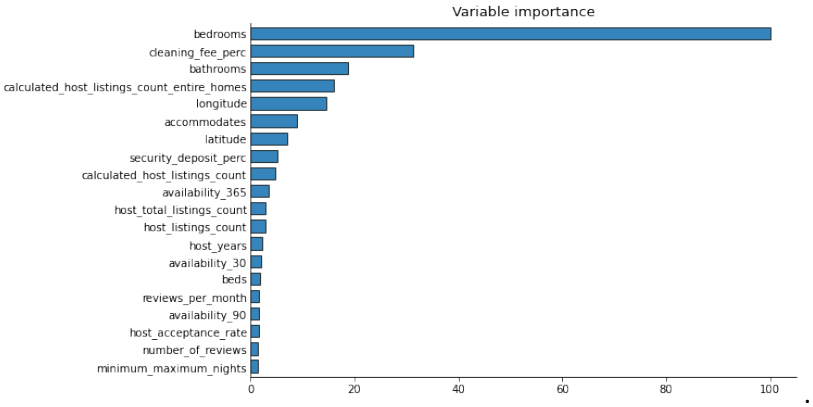


Figure8 Variable importance

Model 2 Model Stacking (Random Forest + XG boosting + Linear regression, meta model: Linear regression)

Part A: Rational of choosing the model and how they make sense for the data

The idea of fitting the model stacking is quite simple, as we want to combine predictors from different models to improve our overall model performance. The advantages of using model stacking is as following:

(1) It can reduce the variance of the model.
(2) We do not need to select the best model from the validation set any more. Instead, we can stack them together.

Part B: How the model is fit

It is believed that we should combine the models that are as accurate as possible and at the same time as diverse as possible so that the correction between them are relatively weak ,which in turn may generate a model with relatively low variance and make a better trade-off between variance and bias. Thus, based on the validation set score on Kaggle, we chose Random forest, XG boosting and linear regression to do the stacking. It should be noticed that, in fact, on the validation set, it is the bagging and the random forest that predict better than others, however, as the bagging and random forest are quite similar. We used the next two best models- XG boosting and linear regression to stack. Finally, in terms of the choice of meta model, the validation set method is used for selection. It is found that as we change the model from the linear regression to other more complex models, the validation performance does not improve that much but it makes the model more sophisticated than before. Thus, the simple linear model is finally used as a meta model.

Part C: Interpretations of the models

One of the limitations of stacking models we believe is that it is hard to interpret the results because we used the predictions from the random forest and XG boosting as the inputs of the linear regression, which makes it extremely hard to gain some useful insights from it.

Model 3 Regression Tree (data mining model)

Part A: Rational of choosing the model and how they make sense for the data

It may seem that the regression tree model are not that useful because as the validation set score shows, it performs even worse than our benchmark linear model, however, the regression tree has a greater interpretable ability over the other models and are easy to use it to explain with the person that do not know much about the machine learning. Therefore, in the data mining section, we would use this model to derive some insights and make suggestions to hosts.

Besides, the other main reason we choose it is because it could naturally capture the interaction effect and could automatically perform the variable selection, which is suitable for our house pricing dataset that has may features.

Part B: How the model is fit

In terms of fitting the regression tree, the overall algorithm consists two process. Firstly, it grows a maximal tree by the recursive binary splitting, under the constraint of minimum leaf size. In our case, we control the number of samples in each leaf be not less than 100. We start from the top of the tree, for the first node, the algorithm quickly scanning all the inputs and choose the one that gives the lowest mean squared error, partition the Airbnb data into two regions and then we keep fitting until we reach the leaf size of 100. It should be mentioned that the recursive binary splitting is a greedy algorithm as it immediately chooses the best predictors at that node without consider the further fitting process. Secondly, the cost complexity pruning is done to make penalty for the size of the trees so that the model

will not be overfitting. One more thing to mention is that the regression tree scales well for our Airbnb datasets since it only takes a second to fit the model.

Part C: Interpretations of the models

The regression tree graph we get is quite straight-forward and easy to understand. Similar to the result we get from the random forest model, the number of bedrooms is still the most powerful inputs as it is be selected as our first split nodes. In addition to this, the number of bathrooms, cleaning fee percentage on nightly rate and the location (longitude and latitude) occurs frequently at the split nodes. The more detailed analysis will be presented in the data mining section.

Model validation

Below is the summary table of the validation score from Kaggle and the computational cost of these six models.

Models Name	Score on Leaderboard	Run Time
Linear Regression	0.397	0.000s
Regression Tree	0.441	0.000s
Random Forests(default)	0.456	0.001s
Random Forests (best estimator)	0.380	11min 2s
Bagging	0.381	230.324s
XG Boosting	0.387	28.834s
Stacking ('RF','XG Boosting' to 'LR')	0.362	2min 19s

We utilized the last pair of models that we selected in stacking to conduct model evaluation.

As we discussed before, according to score on Leaderboard, the model of stacking linear regression, XG Boosting and random forest performs best, followed by random forest and bagging. However, it should be noticed that they all spend much more time than others, leading to relatively higher computational costs.

Data Mining

In this section, we are going to provide some market advices for hosts, real estates investor and other stake holders, particularly focusing on how to let them rent their house at higher prices and generate more revenues. Our insights are gained from the regression tree model, which, as mentioned above, has a good interpretability.

From figure 9, we could see that at the first node of the regression tree, the split variable is the number of bedrooms, which is an extremely useful forecaster for predicting the rental prices and it roughly determines that in which range of prices the hosts could charge for their houses. Thus, the insights we provided will target on hosts with different numbers of bedrooms. (Note that the numbers of bedrooms below have a half value, such as 2.5 and 3.5, and the half value will be treated as a small study room or sofa bed that can accommodate people.

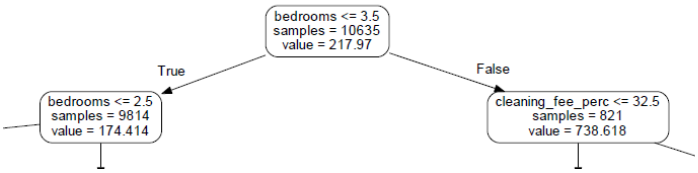


Figure9 the first node from regression tree

Insight 1 For the hosts that have more than 3.5 bedrooms

If a host's house has as a quite larger number of bedrooms, it is likely that they could charge a per night price between the range of $ 661.195 and $1432.087. Afterwards, based the part of tree as shown below, we could observe how the hosts that charge the highest price do.

Firstly, one of the key thresholds is that the cleaning fee percentage on the nightly rate are lower than 32.5%, suggesting that the host with large number of bedrooms should charge a relative lower and reasonable cleaning fee. Then, the next important variable is the calculated host listing counts, which measures the experience of hosts or more specifically, distinguishes the business from individuals. The business, normally, will know more about renters and can have their needs satisfied, and that may be potentially one of the associations to let the business party charge a higher per night house price.

Similarly, if hosts' or investors' houses have a relatively large number of bedrooms, it should charge a lower cleaning fee and consider cooperating with business to generate larger revenues.

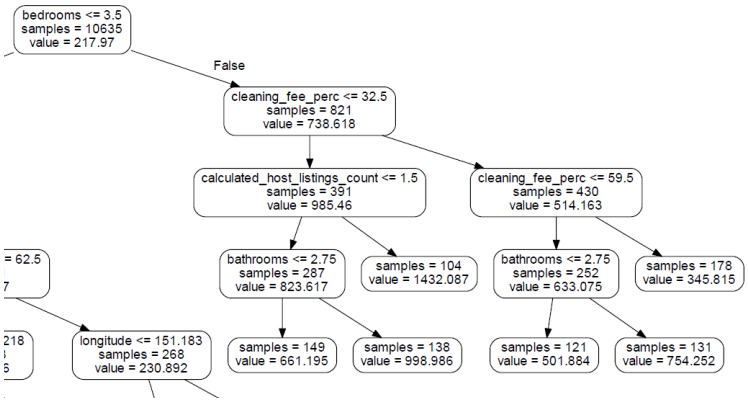


Figure10 Part of the regression trees for bedrooms more than 3.5

Insight 2 For the hosts that have the number of bedrooms between 2.5 and 3.5

If hosts have 2.5 to 3.5 bedrooms, their daily prices on Airbnb should fall in the region of ($181.874, $694.963). The hosts that can potentially get the highest revenue in this group charges a cleaning fee percentage less than 38.5%

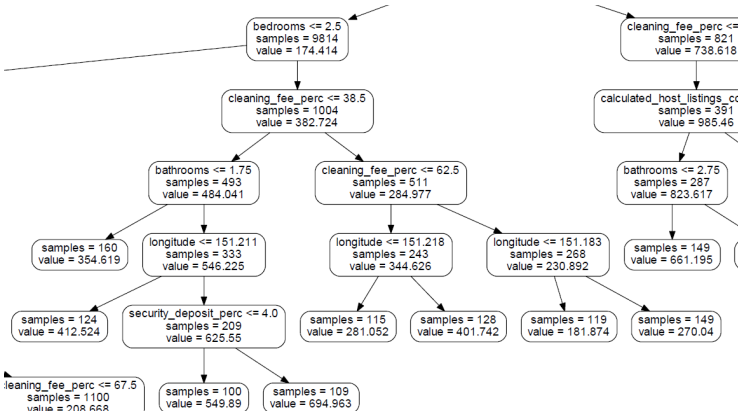


Figure11 Part of the regression trees for bedrooms between 2.5 and 3.5

Insight 3 For the hosts that have the number of bedrooms less than 2.5

Our data suggests that a quite large number of hosts have less than 2.5 bedrooms. Thus, number of leafs for this group is larger and the per night house price depends on various predictors. Hence, we summarize five key suggestions based on the features that occur quite frequently on the split nodes or occur at the top of branch:

(1) Allow for more than 4 clients checking-in.
(2) Choose the house position wisely.
(3) Consider cooperating with some rental companies.
(4) Charge lower cleaning fee.
(5) Response your potential renters as soon as possible.

Conclusion

In conclusion, our group constructed some models in machine learning in order to predict house prices on the Airbnb platform. The process is to firstly look into the data given to us so that we can get a glance of it and further dig the similarities as well as differences, which may contribute to feature selecting. Subsequently, exploratory data analysis is carried out based on some figures such as boxplot. Feature engineering is performed followed by methodology of the models we used. We selected the two best performed model according

to Python output as well as the score on Kaggle's leaderboard. Data mining is depicted at the end to provide some insights and give further suggestions to clients.

We have found that adding the extra number of bedrooms or bathrooms may be not that easy due to the limitation of area for those individual hosts. Nevertheless, lightering cleaning fees, reducing unnecessary listings count, and allowing reasonable accommodates are relatively easier for them to boost profits by renting at the Airbnb platform. Besides, collecting more data, such as the quality of photos the hosts posted online, the level of detail describing to the room, etc., are considered to be important when booking in our real-life experience.

There are still some limitations of our models due to many other reasons such as incorrectly filled missing values. In the future, we can better help the clients if we have more accurate data. As a word saying, 'More data beats cleverer algorithms.' (Domingos, 2012), a larger size of data is of great contribution in most of the time. This depicts the importance of data scale in machine learning. If we have a perfect algorithm that is generated from a small training dataset, there is no meaning even if we have 99.9% of precise. The other half of the word is, 'Better data beats larger data.', which shows the importance of data quality. Machine learning based on large data size will not drop out of time but can be improved by the introduction of new learning models.

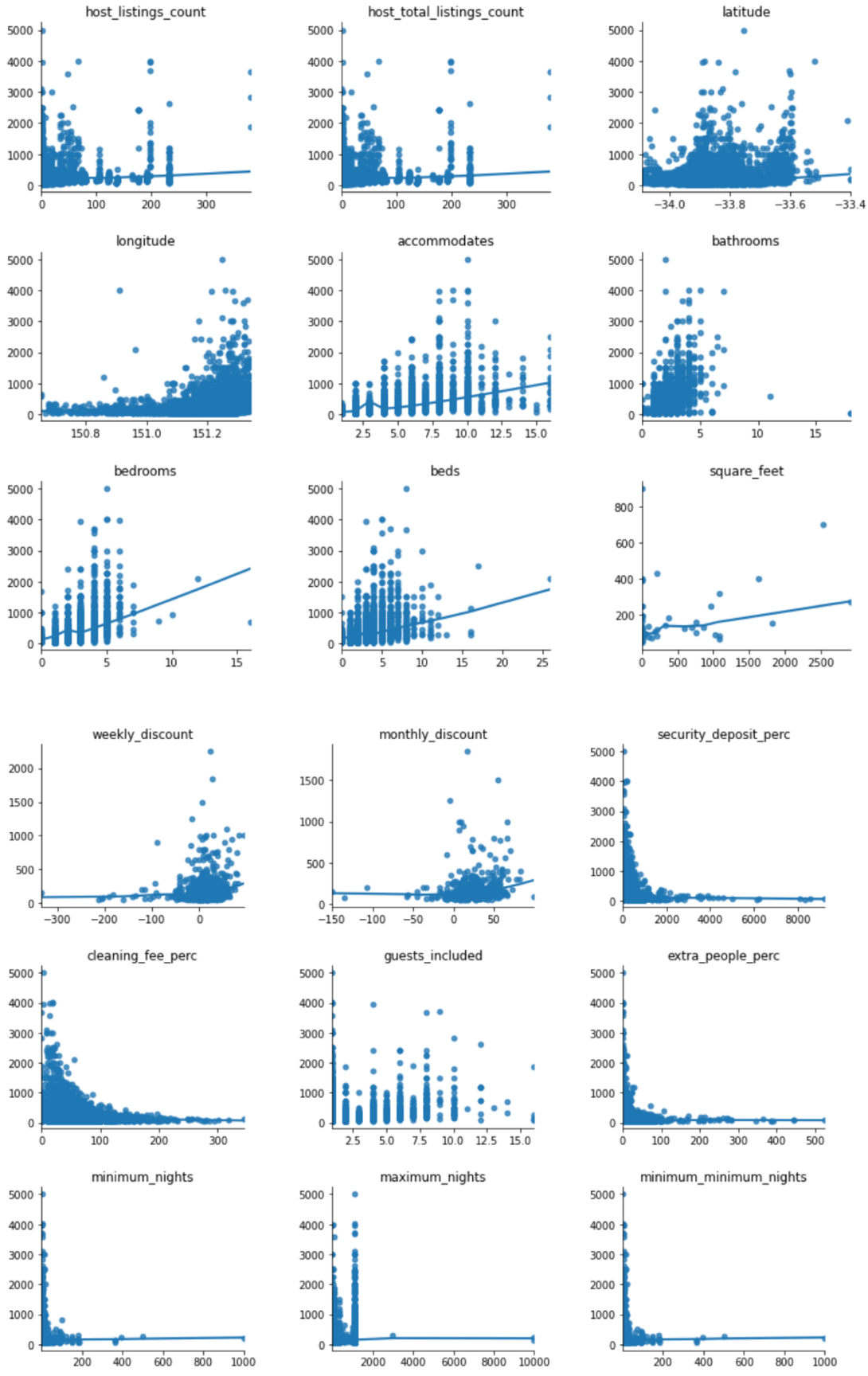
Reference

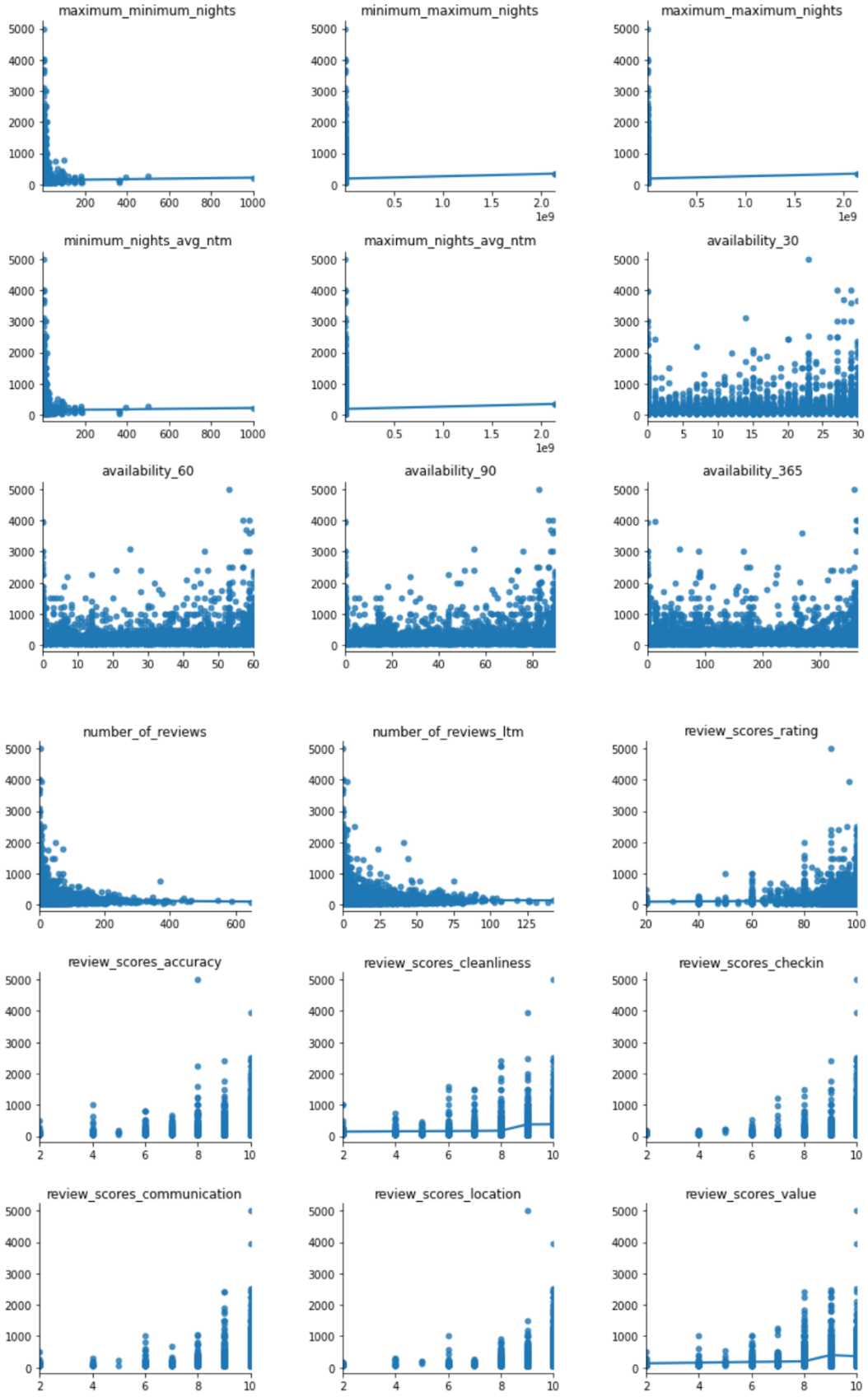
(2020). Retrieved 30 May 2020, from https://www.airbnb.com.au/help/article/2526/as-a-host- what-should-i-know-about-security-deposits

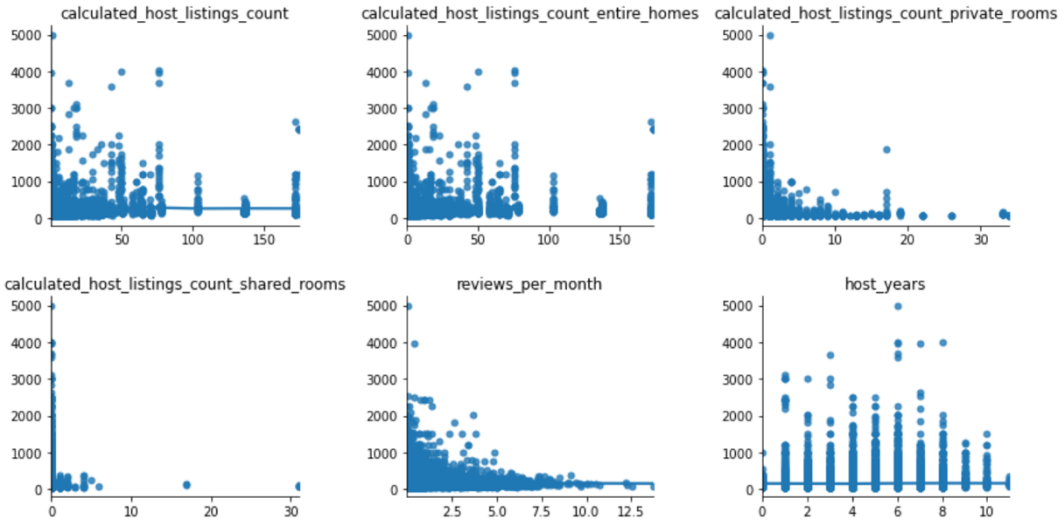
(2020). Retrieved 30 May 2020, from https://www.airbnb.com.au/help/article/2526/as-a-host-what-should-i-know-about-security-deposits

Domingos, P. (2012). A few useful things to know about machine learning. Communications Of The ACM, 55(10), 78-87. doi: 10.1145/2347736.2347755

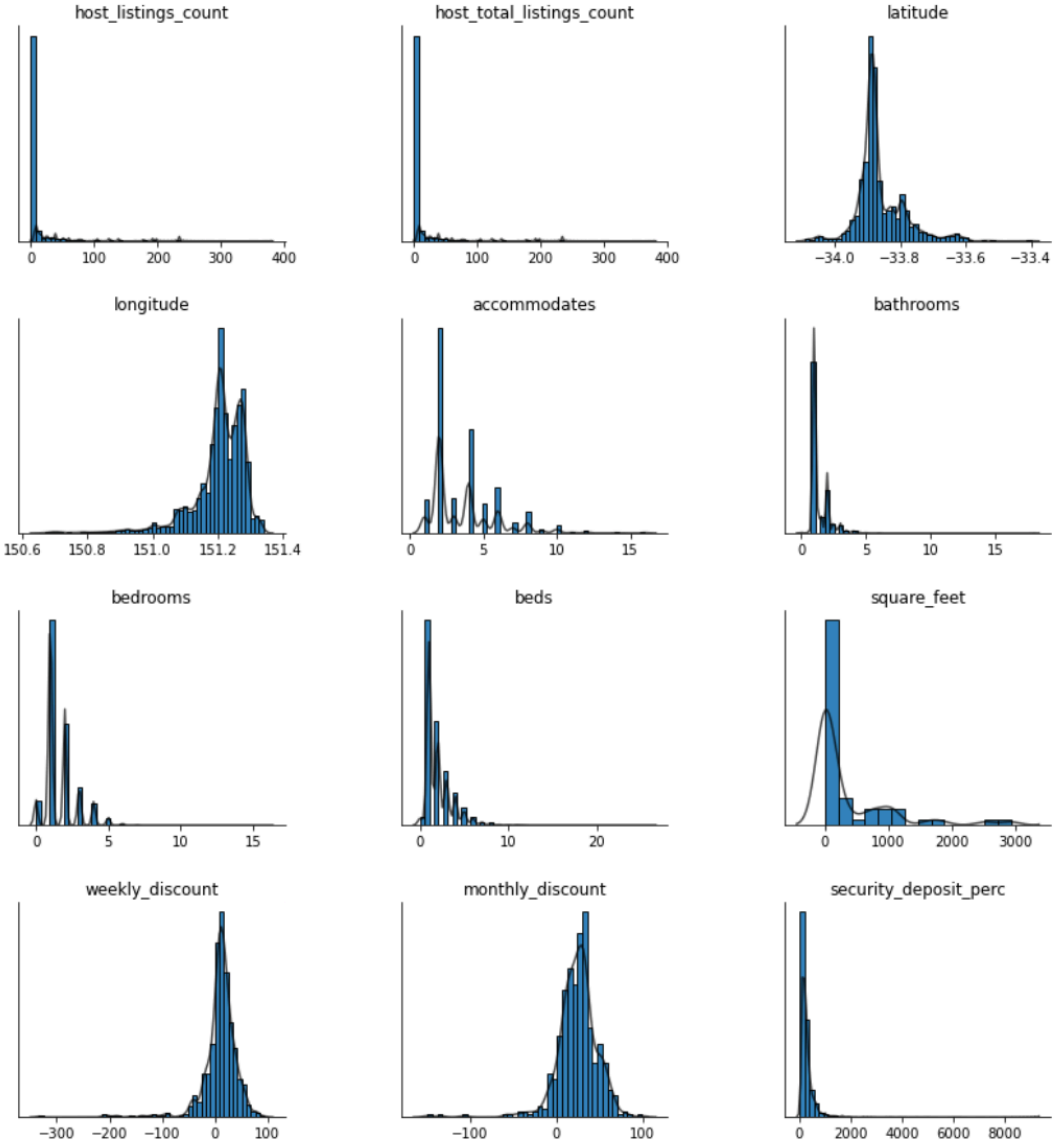
Appendix

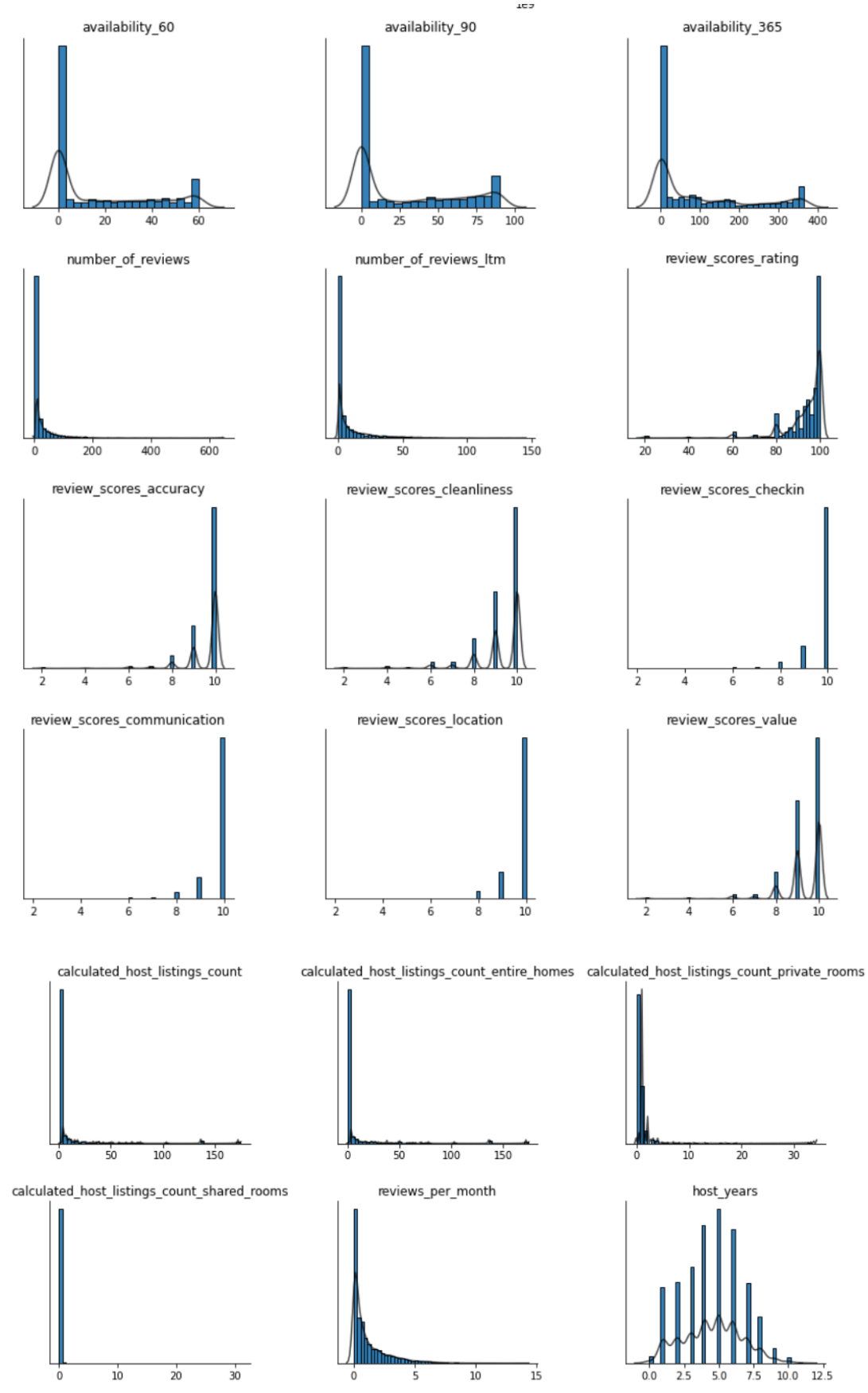

host_listings_count
host_total_listings_count
latitude
longitude
accommodates
bathrooms
bedrooms
beds
square_feet
weekly_discount
monthly_discount
security_deposit_perc
cleaning_fee_perc
guests_included
extra_people_perc
minimum_nights
maximum_nights
minimum_minimum_nights


maximum_minimum_nights
minimum_maximum_nights
maximum_maximum_nights
minimum_nights_avg_ntm
maximum_nights_avg_ntm
availability_30
availability_60
availability_90
availability_365
number_of_reviews
number_of_reviews_ltm
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
1e9


calculated_host_listings_count
calculated_host_listings_count_entire_homes
calculated_host_listings_count_private_rooms
calculated_host_listings_count_shared_rooms
reviews_per_month
host_years


Correlation matrix
host_listings_count
host_total_listings_count
latitude
longitude
accommodates
bathrooms
bedrooms
beds
square_feet
weekly_discount
monthly_discount
security_deposit_perc
cleaning_fee_perc
guests_included
extra_people_perc
minimum_nights
maximum_nights
minimum_minimum_nights
maximum_minimum_nights
minimum_maximum_nights
maximum_maximum_nights
minimum_nights_avg_ntm
maximum_nights_avg_ntm
availability_30
availability_60
availability_90
availability_365
number_of_reviews
number_of_reviews_ltm
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
calculated_host_listings_count
calculated_host_listings_count_entire_homes
calculated_host_listings_count_private_rooms
calculated_host_listings_count_shared_rooms
reviews_per_month
host_years
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


host_listings_count
0
100
200
300
400
host_total_listings_count
0
100
200
300
400
latitude
−34.0
−33.8
−33.6
−33.4
longitude
150.6
150.8
151.0
151.2
151.4
accommodates
0
5
10
15
bathrooms
0
5
10
15
bedrooms
0
5
10
15
beds
0
10
20
square_feet
0
1000
2000
3000
weekly_discount
−300
−200
−100
0
100
monthly_discount
−100
0
100
security_deposit_perc
0
2000
4000
6000
8000


availability_60
availability_90
availability_365
number_of_reviews
number_of_reviews_ltm
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
calculated_host_listings_count
calculated_host_listings_count_entire_homes
calculated_host_listings_count_private_rooms
calculated_host_listings_count_shared_rooms
reviews_per_month
host_years