# Predicting Prices Using Airbnb Dataset

Mahrukh Khan

## Introduction

A real estate company needs prediction for a suitable price per night for their newly built apartments in Thailand. The apartments have a stringent accommodation capacity of 2-6 people. For precise prediction of charges per night I used OLS Linear Regression and three machine learning algorithms; Lasso, Random Forest and Gradient Boosting Method. RMSE was used as the loss function to measure performance fit of the models.

## Data (Feature Engineering and Sample Design)

I selected the dataset from Airbnb, available on Inside Airbnb website, which had approximately 17,000 observations. The target variable was price per night, initially given in thai baht. For an easier understanding of price variation, I converted it to dollars. The predictor variables I selected were linked to size (number of guests, number of beds, number of bathrooms etc), reviews, host information and neighbourhood location. Overall, the data quality would be considered good. This data required intensive data cleaning. Data was selected for certain property types (structurally representing an apartment) that accommodated 2-6 people. Unnecessary columns were removed and those with 'true/false' values were converted into binary columns. All numerical variables were properly formatted. Amenities such as having wifi, tv or parking space for each apartment were given as a vector input in each row of the column. It was unlisted, cleaned and converted into individual binary columns. Certain quantitative variables were pooled into categories: number of bathrooms, reviews, and minimum nights. There was a significant number of missing values in the predictor variables that was dealt with either replacing with a suitable value (mean or median), creating flagged variables, or dropping a column if it did not hold significance. In the dataset, all rows missing the value of price were dropped. Furthermore, I explored the distribution of Airbnb apartment prices. It was strongly skewed with a long right tail whereas log of price was close to normally distributed. I decided to choose level price: makes it easier to interpret and there wasn't significance difference in the prediction when conducted with log of price.

## Model Building

**Patterns of Association:** Patterns of association were observed for average price per night with three predictor variables: number of guests, beds, and reviews. The number of guests showed an approximately linear relationship with potential convexity. Number of beds and reviews signalled towards a weaker link with the target variable. The former seemed almost flat, and the latter showed signs of convexity followed by a slight increase and then a decrease. Considering their patterns, a quadratic term was added for numbers of guests and beds, whereas a cubic term was introduced for number of reviews.

**Interaction Terms:** Using domain knowledge and observing trends of certain predictor variables, interaction terms have carefully been selected. The process involved interacting various amenities with categories of properties and rooms. For instance, if the type of property had parking on its premises or had access to a kitchen. Variation of average price per night within some categories was observed, hence they've been included in our complex model.

### Final Models

**Model 1**: *Number of guests accommodated, Number of guests accommodated (squared term), Number of beds, Number of beds (squared term), Number of bedrooms, Number of bathrooms, Number of minimum nights, Number of days since the first review, Number of days since the owner became a host, property type, room type, neighborhood*

**Model 2**: *M1 + reviews + amenities*

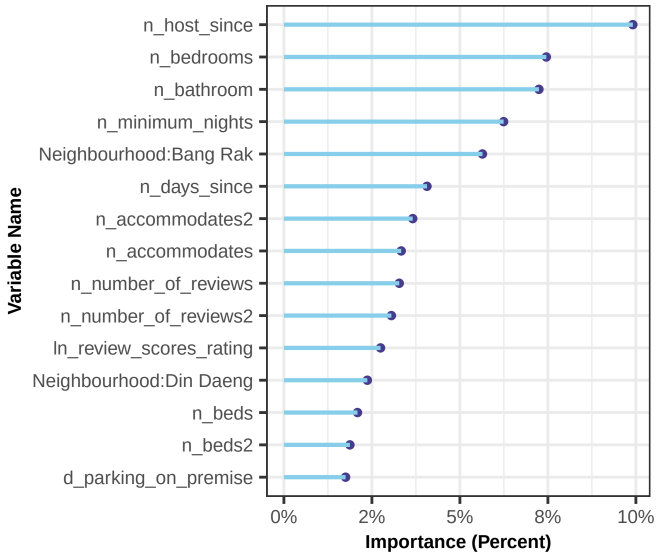
**Model 3**: *M2 + interactions*

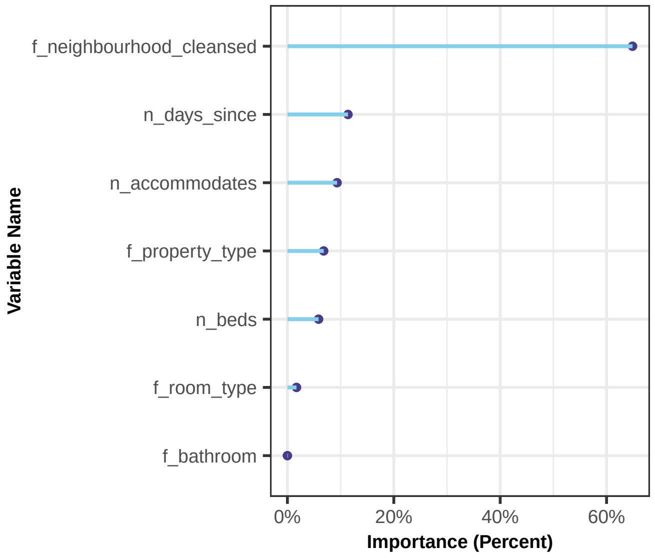
## Random Forest (Benchmark Model)

Random forest uses bootstrap aggregation to give us predicted value of our target variable with the help of the x variables. I first split the sample into two sub-samples, a holdout sample (30%) and the work sample (70%). A 5-fold-cross-validation was done on the work sample. I ran two random forest models: Model 1 (only basic variables) and the second used Model 2 (adding reviews and amenities). The tuning parameters for the models were 500 bootstrap samples, square root of the total variables which was approximately 7 and minimum number of observations in the terminal node was set to 50. Due to the robustness of the algorithm towards tuning parameters, not a lot of variations were tried. The RMSE points to Model 2 being a better prediction model than Model 1. The former had 52.07 RMSE whereas the latter had 53.08, indicating that Model 2 is slightly better. To further understand random forest, our benchmark model, I use diagnostic tools to reveal the patterns of associations that drive the prediction.

### Variable Importance Plots

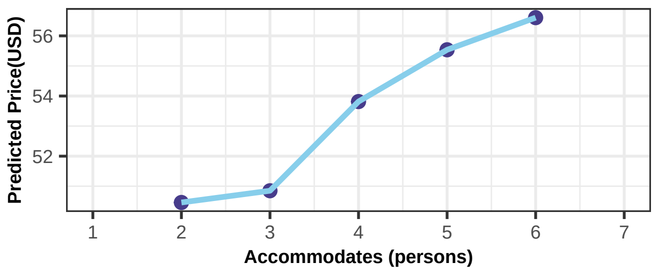
The plot captures which predictor variables matter most to the prediction. The reduction in mean squared error, also known as improvement of the fit, was averaged, and then divided by sum across all predictor variables such as neighborhoods, amenities, reviews etc.

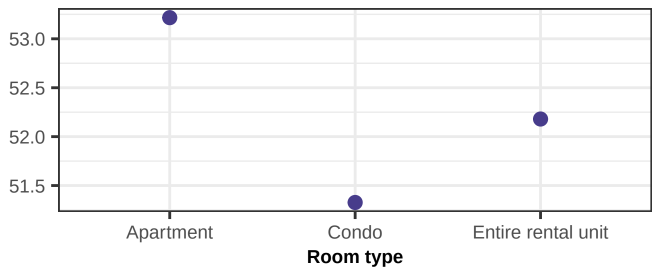




For better results, I decided to view the top 15 predictor variables significant for this model. The variables that matter the most are for the number of beds, bathrooms, reviews, and guests accommodated. It also looks at the neighborhood the apartments are situated in. Amongst the most significant variable is the time-period of the person being a host on Airbnb which could reflect their credibility. In addition, the days passed since their first review which could also reflect on their reputation. I also decided to group the factored variables. The neighborhood, number of days since first review, accommodation size, property and room type matter the most amongst these variables.

### The Partial Dependence Plot





For studying the patterns of association between some predictors and y variable, I use partial dependence plots. I picked two important variables: the number of guests to accommodate and property type. The plot with number of guests shows that average price gets higher with a rise in number of guests. For property type, it can be seen that apartments have a higher average price night followed by entire rental units and then condos. We can see that the variation is approximately 1 to 2 dollars which is hardly significant. It makes sense as they are of a similar structure.

## Comparing with other models:

**OLS Linear Regression:** This regression was conducted using all three models on the training set. Root mean squared error was used to differentiate between the models' prediction power. We can observe that RMSE tends to slightly decrease with addition of variables and their interaction terms. Model 3 produces the lowest RMSE that is 62.04. Since there is slight variation with Model 2, it can also be chosen for price prediction due its lesser complexity.

**LASSO:** This algorithm was given all the predictor variables and their interaction terms as an input. Lasso penalizes those predictor variables that make the smallest reduction in the fit of the model by lowering their coefficient value, turning some to zero (dropping them). For the variables it retained, it returned 180 estimated coefficients for this prediction. In our model we provided it varying tuning parameters. It picked the most suitable and returned an RMSE of 61.39 which is lower than the OLS results.

**CART:** The regression tree built with CART results in keeping the most important predictor variables and showcases their interaction terms. We can observe that the RMSE tends to fall with a more complicated built of our model. However, at a certain complexity parameter which is 0.0077, it rises from 62.55 to 62.66 due to overfitting. Our final stopping rule for the subsequent split is when the improvement in R Squared is less than 0.05. This gives us an RMSE of 62.55.

**Gradient Boosting Model:** For GBM, I decided to use two different models through differentiation in tuning parameters. The first model had a learning 0.1 whereas the second used various values between 0.01 and 0.5. The rest of the parameters were kept same that is 250 number of trees and 20 minimum number of training set samples in a node. The tuning made a very minute difference between both models: first one had an RMSE of 58.70 and the second had an RMSE of 58.52.

## Conclusion:

I used my knowledge of machine learning to fulfill this task of finding a suitable prediction price for apartments accommodating 2-6 people. I used Random Forest Algorithm as a benchmark whilst conducting OLS Linear Regression and using Lasso, CART and GBM algorithms in addition.

Table 1: Horse Race of Models CV RSME

| | CV RMSE |
|---|---:|
| OLS | 63.48005 |
| OLS (model w/ amenities and reviews) | 62.78408 |
| OLS (model w/ interactions) | 62.04336 |
| LASSO (model w/ interactions) | 61.38933 |
| CART | 62.55474 |
| Random forest 1: smaller model | 60.12400 |
| Random forest 2: extended model | 59.35233 |
| GBM (basic tuning) | 58.70258 |
| GBM (broad tuning) | 58.48166 |

For the cross validated RMSE set, we can observe that GBM and Random Forest outperform the rest. This is followed by LASSO. The least favourable prediction model in this set is the simple OLS Linear Regression. In the holdout set it can be observed that there's a reduction in overall RMSE values, which could partly be due to the different sample size. The best fit model is GBM (broad tuning) for the entire data set. In case we have high external validity, we can expect to make a 58.5 dollars error when using our model on live data. Whereas GBM does give a slightly better prediction, the advantage of using random forest algorithm is that variables can be studied in more depth with diagnostic tools.