Assignment 2 - Predict Airbnb prices in Bangkok

Ádám Vig (1903211)
25 October 2021

In this assignment, I build a model to predict Airbnb prices in Bangkok. I use data from September, 2021 and an XGBoost model, to predict prices of apartments for 2 to 6 people. All code can be found in this Github repository.

Data and feature engineering

The data contained daily rent for the apartments - this is my target variable - an variables such as *number of accommodates, number of beds, number of bathrooms, the neighborhood, average review score, number of reviews, minimum and maximum nights allowed.* The database contained text describing additional amenity objects in the apartment (eg. TV, *elevator*, $refrigerator$ etc.), I created dummy variables from the text and dropped those, that were present in less than 10 observations. In the *neighbourhood* variable, I pooled those into an 'other' category, that had less than 50 observations. I also created the *distance (km) to the Grand Palace*, one of the most famous tourist attraction in the city center using longitude and latitude. I created interactions but I only used them in a LASSO model. My initial database contained 17500 observations, but some had missing prices, I also dropped single rooms, hotel rooms etc. and filtered apartments for 2 to 6 people, leaving 8162 observations and 125 features and in my final sample. The original price was given in thai local currency, I converted it to EUR for easier interpretation.

Error handling in the target variable

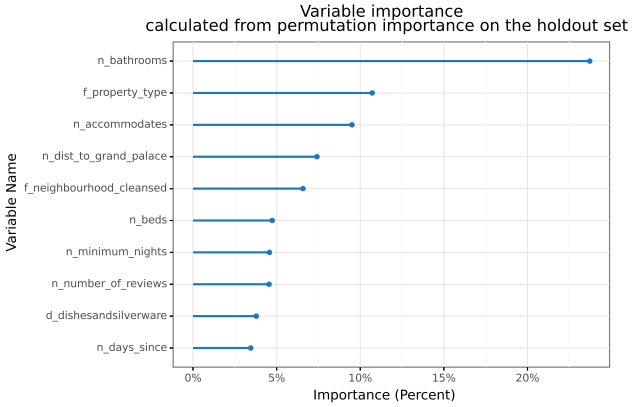
The target variable appears to have some errors. In some cases instead of daily rent, the yearly rent is given resulting in extreme values otherwise unexplainable. I checked some of these apartments on airbnb.com with the *id* variable and these are mostly small condos with price 3-5000 EUR per night while there are much better apartments in the city center for 80-100 EUR per night. I decided to drop these observations based on the price in my training set, while leaving in the test set to see how my model works in live data. Additional, I calculated model performance on the truncated test set, and I advise the company if it wants to enter the market and label its own apartments just rely on the train and truncated test results while if it wants to find good deals on the market on future, live data, rely on the full test set results.

Model selection

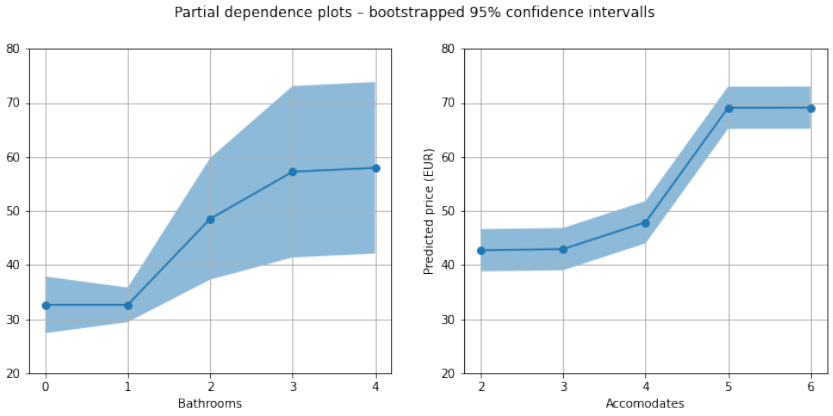
I used a 70-30 train test split, trained an OLS reference model and a cross validated LASSO, Random Forest and XGBoost on a set of reasonable hyperparameters. Then, compared models with the best hyperparameter configurations to choose my final model, an XGBoost model. This model produces 17.93 EUR RMSE on the train set, 81 EUR RMSE on the test set, ant 17.13 EUR RMSE on the truncated test set. The reference OLS model's numbers were the following accordingly: 19.5, 82, 19.4.

Diagnostics

The variable's importance to loss reduction are displayed in the following figure. I used permutation importance to calculate variable importances on the holdout set – traditional feature importance calculates this score only on the train set. Interestingly, the most important feature is the *number of bathrooms*, followed by *property type*, *number of accommodates* and the *distance to Grand Palace*.



I calculated Partial Dependence Plots for the two most important numeric variable, these are displayed in the following figure. As confidence intervals are not implemented in scikit-learn, I estimated bootstrapped standard errors of the PDP-s, and calculated a 95% confidence interval. Both variable is nonlinear in price, and there is a greater variation by bathrooms (although the estimates are less precise).



I calculated subsample performances on the holdout set for neighbourhoods (districts), and the model performs substantially better in Sathon, Khlong Toei and Vadhana (normalised RMSE: 0.35, 0.41, 0.46). On the other hand, the following districts have the worst performance: Din Daeng, Bang Rak, Ratchathewi (normalised RMSE: 1.21, 2.29, 2.32). Note, that these are calculated on the holdout set where I left error values of yearly prices.