Berlin Airbnb Price Prediction

Bazzi, Mariam and Stahl, Tamas

2021-02-02

Executive Summary

This project is an analysis of Airbnb's data in Berlin that is leveraged to predict the nightly prices for apartments to be listed on Airbnb. Airbnb offers complete independence to the host to set a price to their properties with minimal guidelines that allow host to compare similar listings to come up with a comparative price. Our primary motivation, therefore, is to assist a property management company to price their new apartments prior to listing them on the market. These are small to medium size apartments that can accommodate at least 2 guests, and up to 6 guests. With that perspective, we can frame our problem statement, which is to accurately predict the price of the new to-be listed apartments that is optimal in terms of the company's profitability and guest affordability.

Data

The data is sourced publicly from Inside Airbnb for property listings in the city of Berlin scraped on 21 December 2020. The dataset contains numerous continuous and categorical variables, for instance, the nightly price in Euros and the neighborhood, as well as rich text data such as a property's description and URL addresses for a listing's page and photos. Initially, the dataset consisted of 74 variables and 20,224 observations. Each observation corresponds to a unique listing id, and thus it is a large representative sample. There are no observed measurement errors.

Data Cleaning & Preparation

1. Variables that were not related to price were dropped. These included the all variable with URLs, scrape id, descriptions, etc.
2. Variables were formatted and parsed into the appropriate data types.
3. Variables with many missing values, such as host_response_rate and host_acceptance_rate, were dropped.
4. Other variables, for instance bathroom_text, was renamed to bathroom.
5. The amenities column which consisted of a list of the property's facilities, was parsed to separate binary variables.
    1. It appeared that many of those variables would be better represented if they were collated into groups. Therefore, similar variables were aggregated and grouped together. For instance, all variables that were related to WIFI or WIFI speed provided were aggregated to one main WIFI category.
    2. Some variables were contained long sentences or just brand names, and therefore, were dropped.
6. Duplicate variable and observations were removed.
7. The dataset was filtered to keep listings that are apartments, serviced apartments, condominiums or lofts that can accommodate 2 to 6 guests.

The final cleaned and processed dataset consist of 76 variables and 9,891 observations. The price in Euro is our y variable and all other variables are considered potential predictors and screened by different methods to pick the likely predictive ones. As a robustness check, the dataset was split to 80% work data, and 20% holdout data. The final work data, after cleaning and screening for potential predictors, is composed of 7914 observations and 76 variables.

Data Limitations

In an attempt to control for noise in the data, we grouped similar variables in a single category. As an example, there are numerous variables related to TV brands & screen sizes, thus, those were grouped in a single "TV" variable. Some of the groups aggregated are TV, Wifi, sound system, stove or oven availability, kitchenware, clothing storage, and whether an apartment has a balcony or patio. We should also shed light on the limitation of variables in the data set, and that there are many other variable that were not included, but could have impacted the nightly price. We were also subject to potential selection bias as we hand-picked the variables to include.

EDA

As mentioned above the data set consists of 76 variables and 9,891 observations. The target variable is price per night per person, expressed in EUR. As we are predicting the price we should first check whether we should take the level or log of the price variable.

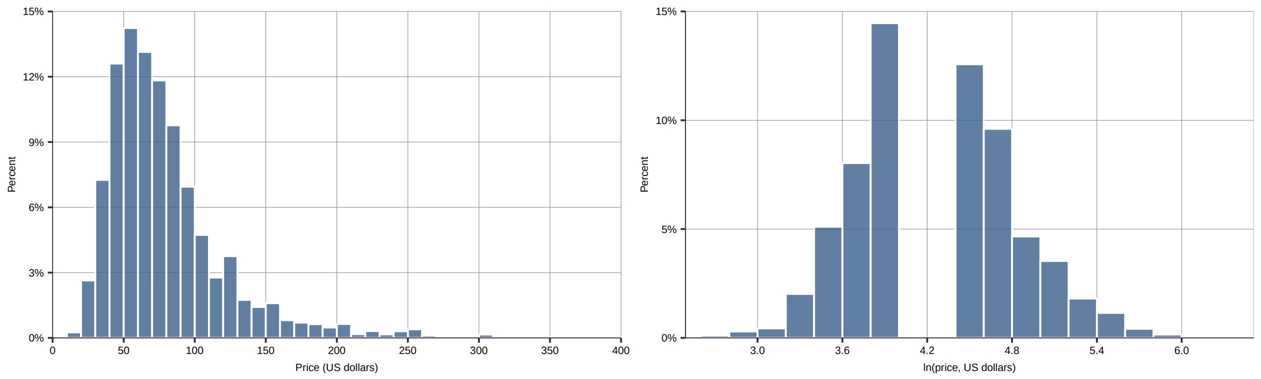

Figure 1: Level and log of Price variable

We could conclude just as in the London use case, that the Airbnb apartment prices are strongly skewed with a long right tail. Log price is close to normally distributed. We chose to use the price variable and not transform it to log as we are more interested in the actual EUR price and not the changes.

Table 1 shows us the descriptive statistics of price for property type.

Table 1: Descriptive statistics of price for property type

f_property_type	mean	median	std	iq_range	min	max	numObs
Entire apartment	77.16	68	49.09	40	10	999	8925
Entire condominium	89.55	76	69.45	50	20	729	303
Entire loft	126.69	98	110.18	65	25	990	327
Entire serviced apartment	126.63	110	81.42	78	26	760	336

Feature Engineering

In order to have some insight in to our data and to start to work on our models, we inspect the interactions between our variables. Interactions between factors and dummies. We chose 6 pairs to inspect, in which 4 factors were the property type and 2 the number of bedrooms. We inspected the intersections of these factors with dummies (air conditioning, dishwasher, elevator, free parking, tv and kid friendly apartment). Out of the 6 intersections visualized the suggested are f_property_type x d_air_conditioning and f_bedroom x d_elevator.

Our variables are the following: Basic variables, which include n_accommodates, n_bathroom, f_property_type, f_bedroom, n_beds, d_air_conditioning and d_dishwasher. In order to prepare more complex models we added several variables, like factorized variable f_bathroom and the review variables. For higher order polynomials we included the square value of the n_accommodates.

Lastly, we have the variable called amenities that includes various words such as air conditioning, tv, free parking, dishwasher, etc. These words were parsed out of text variables, and we created binary variables from those to correspond to each amenity.

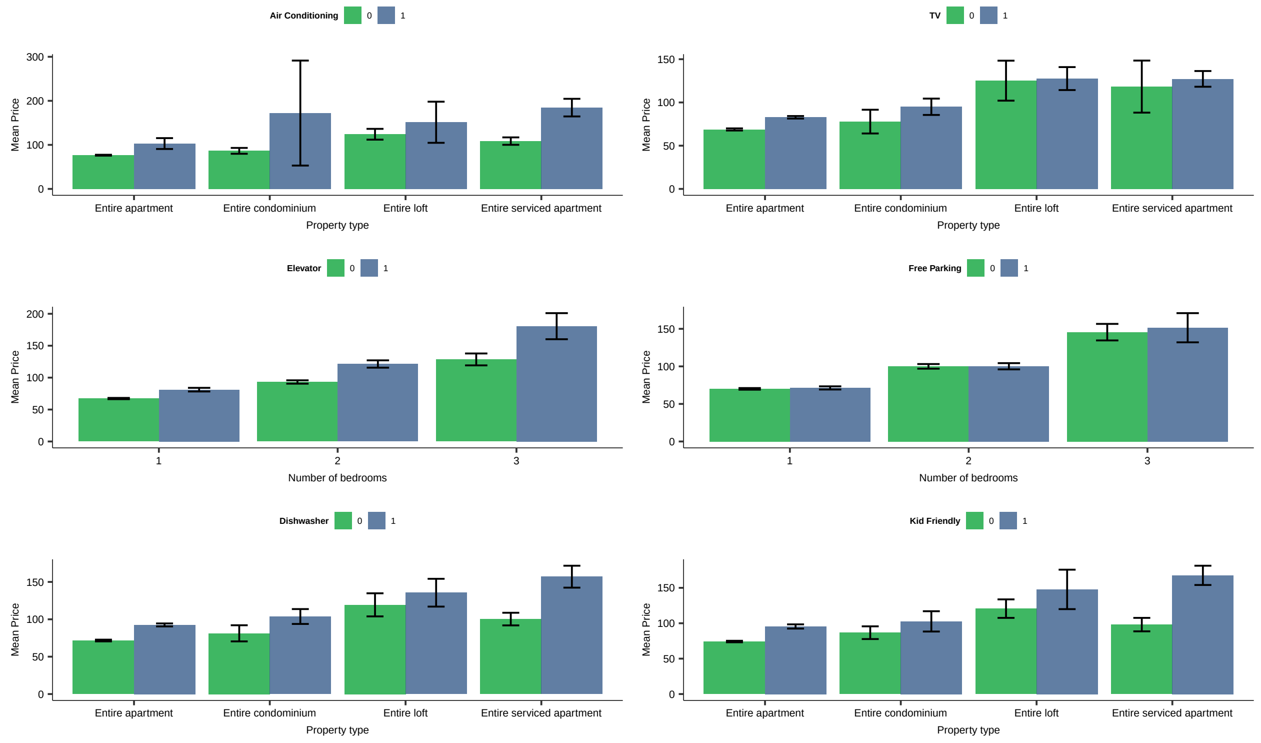

Figure 2: Graphical way of finding interactions

Cross validation - OLS

We prepared 8 models to examine with OLS by adding variables to create more and more complex models.

Table 2 shows the number of variables, R-squared, BIC and cross-validated training set and test set RMSE for the eight regressions. The table shows us two statistics for the entire work set: R-squared and BIC. First we estimated all regressions using all observations in the work set. Then, we estimated models by using 5-fold cross-validation. For each fold we estimated the regression using the training set, and used it for prediction not only on the training set but also in the corresponding test set. For this the Training RMSE and Test RMSE were calculated as the square root of the average MSE on the five training sets and the five test sets.

From R-squared we could say that it is improving as more variables are added. In our case the most complex model explains 37% of the variation in prices. As for BIC it should be decreasing with more complex models, however, after a certain point it increases. But in our case the differences are relatively small. According to BIC in our case the best model is model number 6 as the more complex models have a risk of overfitting the data.

The RMSE in the training set is improving as the model is getting more complex. In the test set it improves until model 7, after it is significantly worse. Model 7 has the lowest test RMSE with 45.95. This model includes all the variables except for the interactions in X3. Model 7 is significantly more complex than Model 5, which was deemed as best by BIC. Model 6 contained the interactions of property type and the additional interactions, meanwhile model 7 included amenities as well.

RMSE suggests that the typical size of the prediction error in the test set is 45.95 euros for model 7, meanwhile it is 46.07 euro for model 6. From a statistical point of view it might be interesting, but if we would look only from business point of view it could be deemed insignificant.

If we have conflict between BIC and cross-validation, cross-validation result should be chosen as it is not based on auxiliary assumptions.

Table 2: Comparing model fit measures

Model	N predictors	R-squared	BIC	Training RMSE	Test RMSE
(1)	1	0.13	84663.28	50.84	50.58
(2)	9	0.26	83426.69	46.78	46.59
(3)	12	0.26	83389.84	46.59	46.43
(4)	13	0.27	83381.99	46.54	46.39
(5)	20	0.27	83379.19	46.32	46.49
(6)	27	0.29	83276.03	45.82	46.07
(7)	66	0.30	83451.79	45.28	45.95
(8)	291	0.37	84625.22	42.57	47.44

LASSO

LASSO has the advantage over building our models with the fact that it is fully automated , with sensible default options for the details of the algorithm. LASSO practically chooses the variables to include, which means that some will be dropped. The important task is to specify the set of x variables. We chose to use the variables of model 7 as that was the best model before.

After running the LASSO algorithm with 5-fold cross-validation we receive the optimal value for lambda. In our case it is 0.8. Furthermore, the algorithm picked 57 predictor variables just like model 7.

The overall RMSE for the five test sets 45.65 which is smaller than the test RMSE of model 7. This is basically due to the fact that the sample size (7913 observations) we used for the prediction is larger than for the cross-validation.

Prediction

The model generates a wide 80% prediction interval. This could result in large errors when predicting a price for an apartment, even with our best model.

Let us show you the uncertainty of the prediction. For an apartment in Berlin which accommodates 4 person (a regular sized family) the predicted average price would be 89.39 euro per night and the 80% PI is between 30.7 euro and 148.1 euro. To reduce the uncertainty we need more observations, therefore it might be useful to revisit our data cleaning and feature engineering. It is important to note that not only the 4-person accommodations have this large uncertainty, all the other subset deal with the same problem. So even if our model is good or near perfect the uncertainty might be still high.

Table 3: Prediction interval results

n_accommodates	fit	pred_lwr	pred_upr	conf_lwr	conf_upr
2	65.15	6.52	123.79	60.73	69.58
3	73.12	14.48	131.77	68.53	77.72
4	89.39	30.70	148.09	84.45	94.34
5	120.00	61.22	178.79	114.29	125.71
6	129.11	70.29	187.93	122.88	135.35

On the first chart, a scatter plot, we could see how predicted prices compare to the actual prices. The range of actual prices are wider than the range of predicted prices as the predicted prices never go above 250, meanwhile actual prices are above 350 euro. This phenomenon might be the result of the fact that regression models have difficulties with predicting extreme values, unless the extreme values are frequent and are in a large number in the data set.

The second chart shows us the prediction interval at 80% level by the number of guests the apartment can accommodate. The Green line represents the lowest and highest predictions, meanwhile the blue bars equal to the predicted value for the apartment. From just a look we could see the high uncertainty as for a 2-person apartment's price for one night could be between 6.5 and 124 euros.

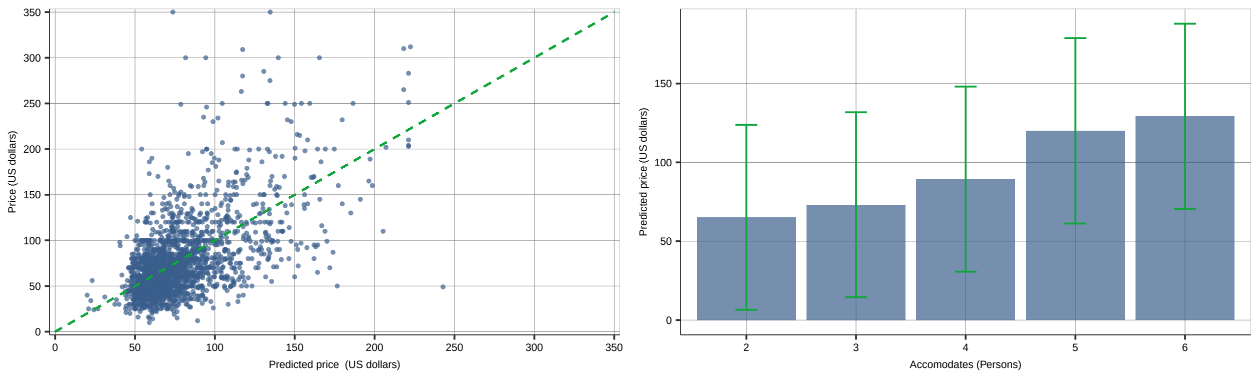

Figure 3: Predicted price plots

Random Forest

We consider all significant interactions that werwe deemed by LASSO for the random forest. We use a 5 fold cross validation to validate if we are overfitting the models into the data.

Model Diagnostics

The variable importance plot on the left shows the top 10 predictor in terms of the largest MSE reduction. Most significant variables are f_bathroom, n_minimum_nights, and n_accommodates. The partial dependence plot on the right suggests an approximately linear association between predicted price and number of guests to accommodate. We then, evaluate the performance of the Random Forest model. The RMSE relative to mean price is measured. The observed relative differences are quite similar for apartment sizes, however, for property type, condominium and loft had higher relative difference of 0.7 and 0.9 respectively.

We added a CART model for comparison with random forest and Lasso. Random forest performed better than the CART single regression tree model.

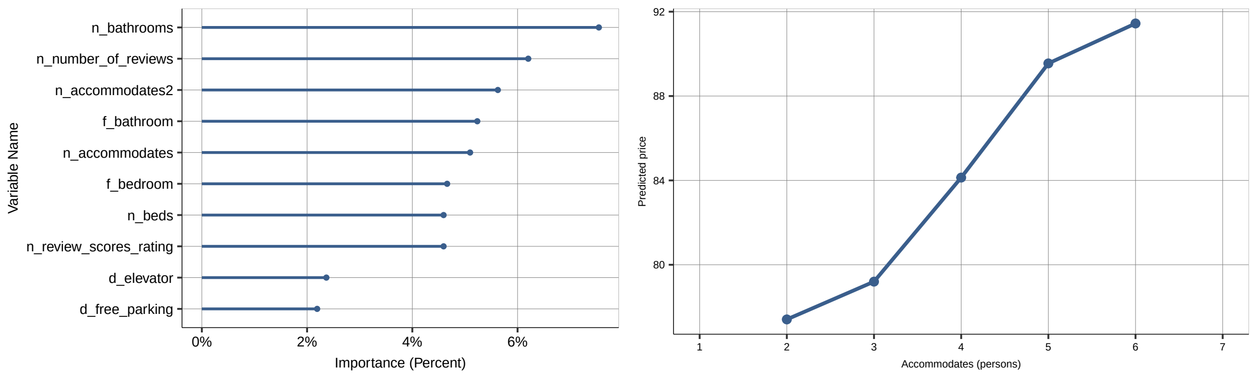

Figure 4: Variable importance plots

Table 4: Performance across subsamples

	RMSE	Mean price	RMSE/price
Apartment size	NA	NA	NA
large apt	62.75	105.5	0.59
small apt	41.46	67.7	0.61
Type	NA	NA	NA
Entire apartment	43.42	76.9	0.56
Entire condominium	70.30	92.1	0.76
Entire loft	132.08	142.6	0.93
Entire serviced apartment	65.19	130.3	0.50
All	50.15	81.3	0.62

Table 5: Horse race of models houldout and CV rmse

	CV RMSE	Holdout RMSE
LASSO	45.652	51.314
Random forest	45.185	50.150
CART	47.998	53.549

External Validity

Since the data set was scraped in December 2020, prices could have changed since then. Thus, affecting the results at the end. Therefore, our results may not be generalizable to the population. If, for instance, the data set was updated on a weekly basis, then this could have provided a higher external validity.

Conclusion

Overall, we have performed extensive data cleaning and feature extraction and engineering, and experimented with various regression and machine learning models. Prediction is not that straightforward. The uncertainty and large prediction intervals make it difficult to conclude exact numbers.

However, this problem could be treated with larger sample size and more observations. We showed that Random Forest slightly outperforms other models, however, it slightly overfits the data compared to other models. This is because the results could not be generalizable; lack of external validity, due to the sample size. Lack of external validity does not deem our findings less significant. We chose model 7 as our best performing model; amongst all 10 models after Random forest.

For the company, we can conclude that best model's RMSE of 45.95. For further improvements, we should be more cautious with feature selection, and split the training set into groups based on the price range. For each group, we would then build a separate model and evaluate performance on each group. This would certainly result in better and more precise predictions.