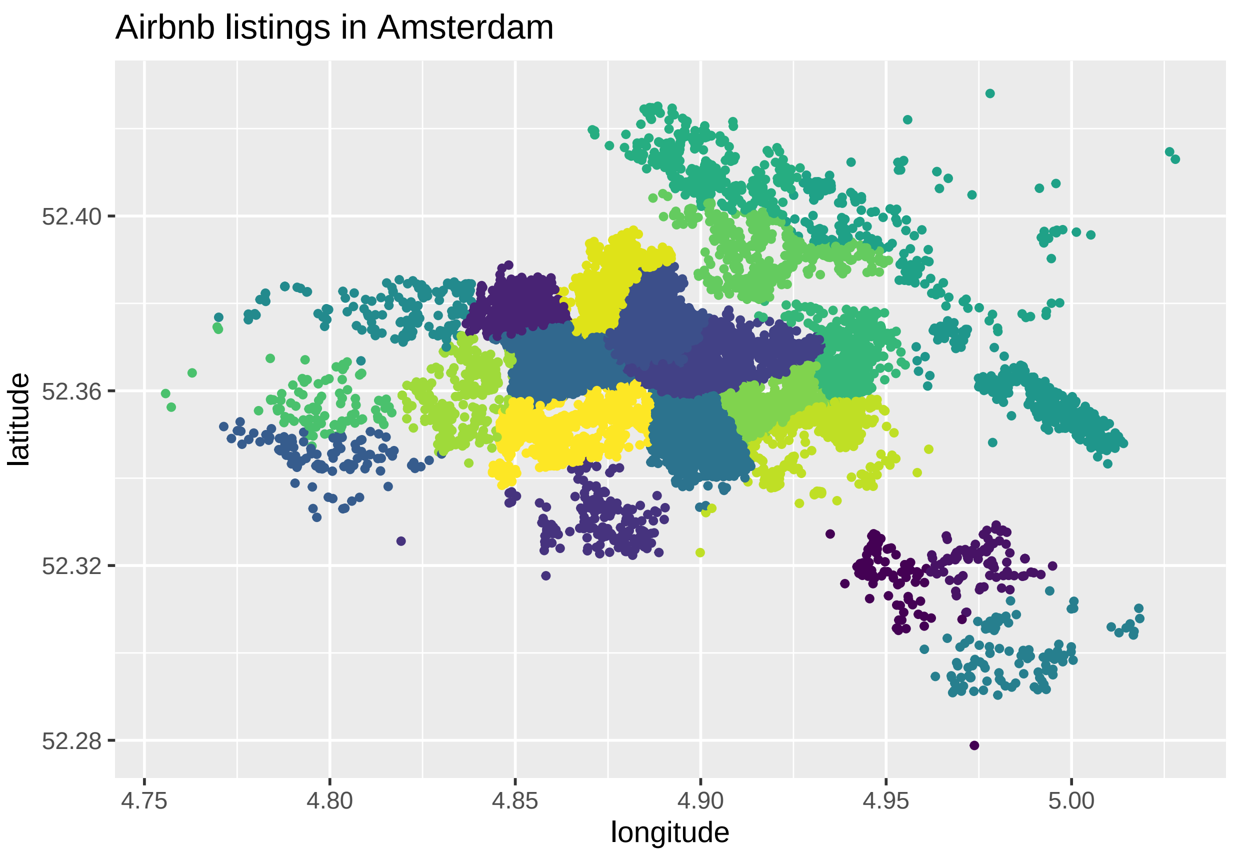
CYOP - Predicting Airbnb prices

Nicolás Sandoval

28-07-2021

Abstract

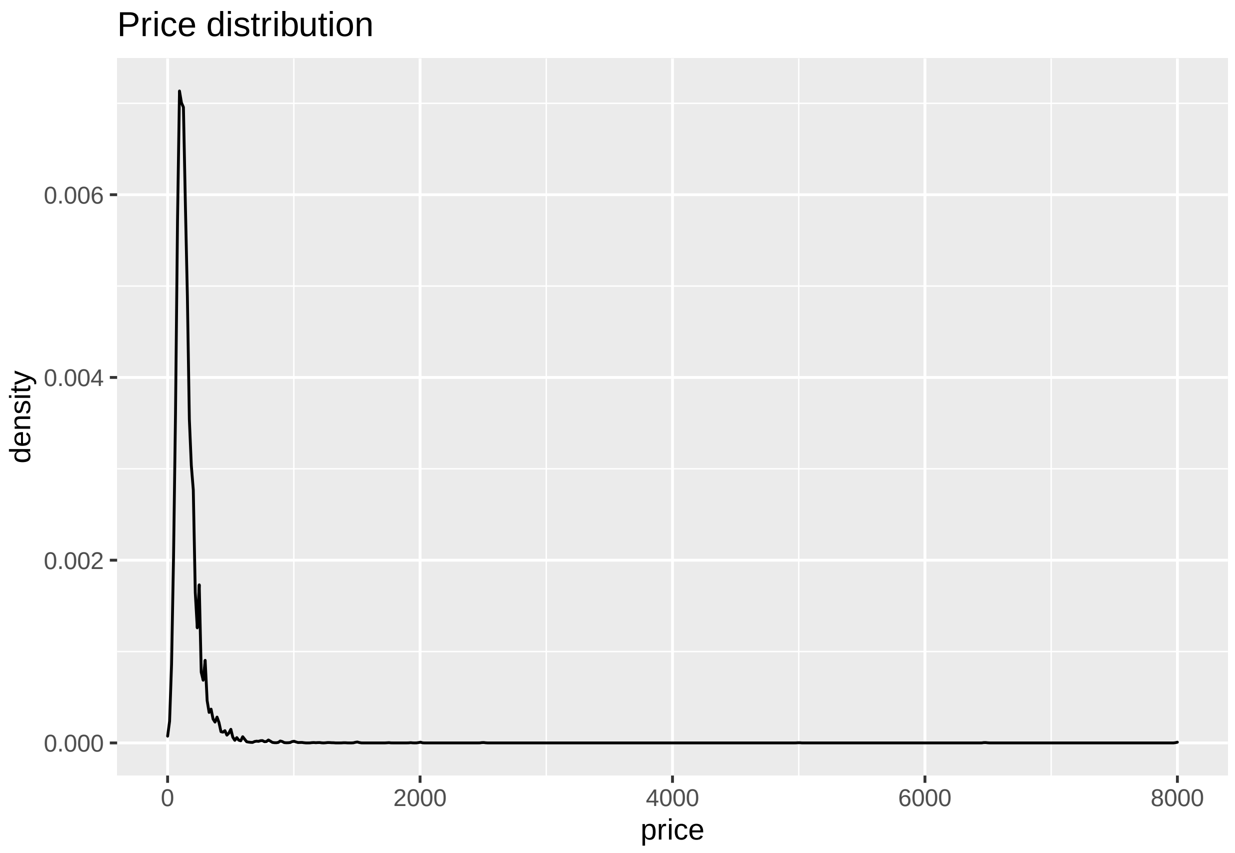
This report is part of the capstone course of the HarvardX Data Science Professional Certificate program. The objective of this project was to build explore a new dataset and then build a regression system. The dataset used was Airbnb's Amsterdam listing, scraped on 2021/07/04 and available here. I decided to predict listing prices based on the rest of the information in the listing, which included location, reviews, number of rooms, among others. To generate predictions I used 2 different algorithms, regularized linear regression via glmnet and gradient boosted trees, using the XGBoost. While linear regression was much faster to run, the gradient boosted trees method gave a significant improvement to the target metric (RMSE). The best linear model's RMSE was 0.407, while the XGBoost model had an RMSE of 0.389.



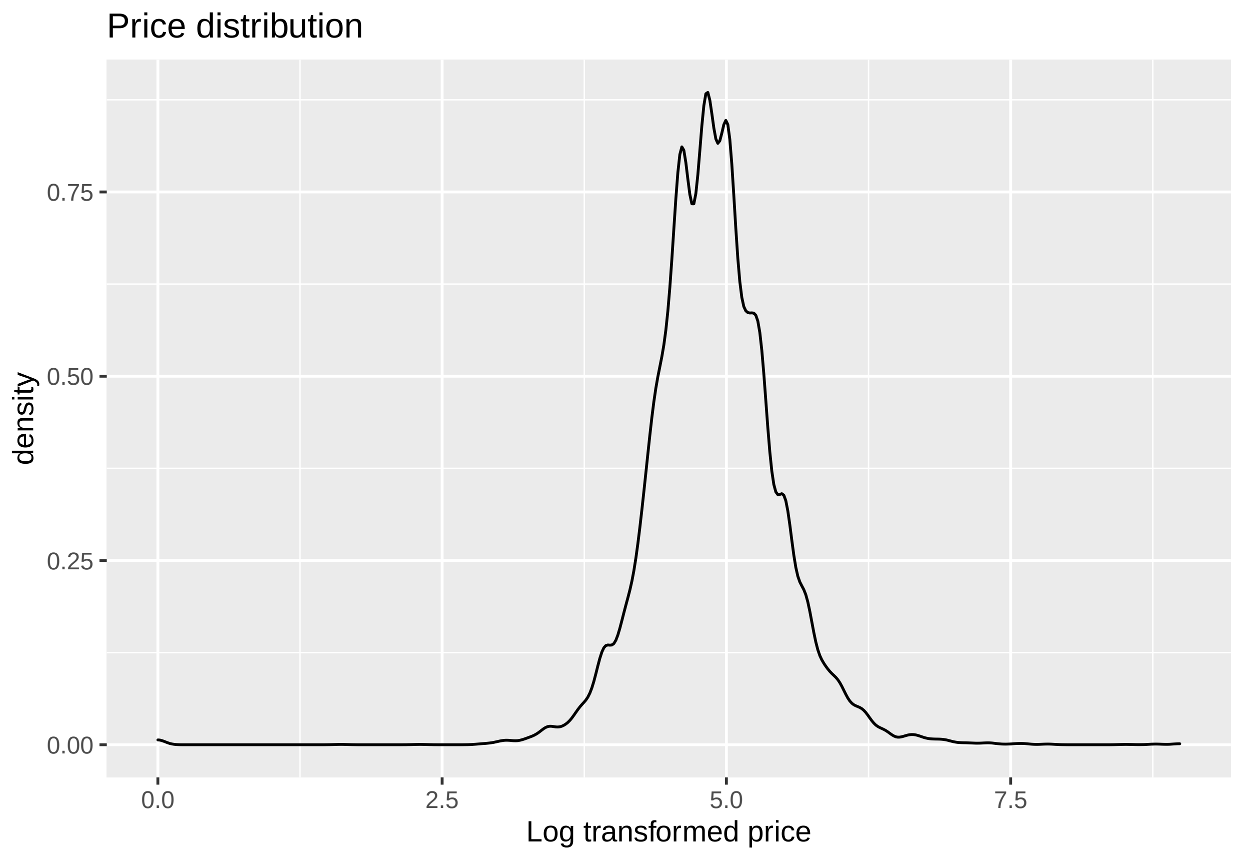
Introduction

The dataset contains 16724 observations of 74 variables, with a mixture of categorical and numerical information, covering Airbnb listing for Amsterdam and surrounding areas as it appeared on the Airbnb website on July 4th, 2021. The data was split into training and testing sets using a 0.85/0.15 split. The goal of this project is to predict the price of each listing, based on the rest of the listing information.

The first column to look at is `price`, which is the outcome the models will attempt to predict. The column has character data ("$150.00"), but it's easy to convert to numeric via parse_number.

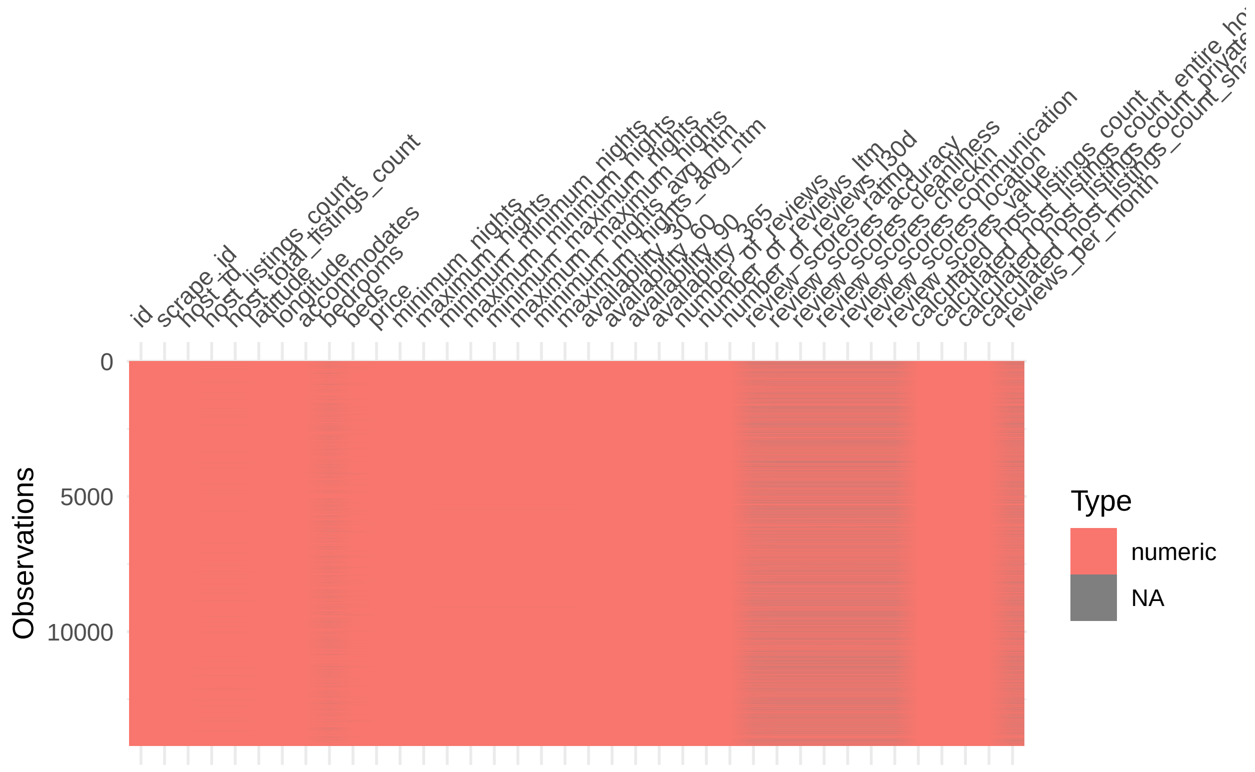


Prices go from 0 to 8000 USD per night and distribution has a significant skew, which goes against the normally-distributed assumption made for linear models. For that reason I log-transformed the price information. I also added 1 to every value in order to avoid listings set to $0 from returning NA.



After the transformation the data looks much close to a normal distribution.

To check for missing data I used the vis_dat function, which let me inspect the columns at a glance.



The numeric rows with missing data seem to be related to reviews and bedrooms, but it might not matter. This will be determined in the methodology section.

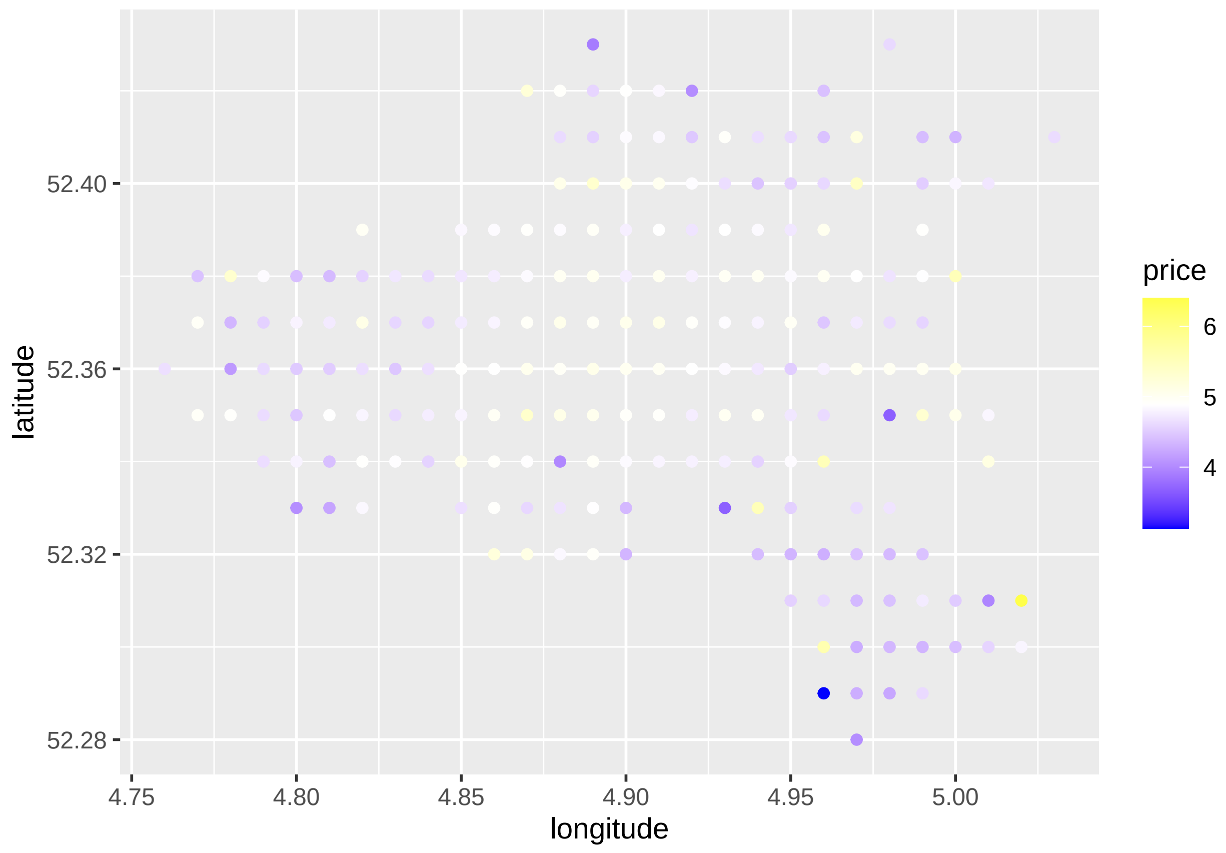
Methodology and analysis

Exploratory Data Analysis

Since I hadn't worked with geographical data in the context of machine learning, I was interested in testing if it had predictive power, so I started by visualizing it, this is the graph that appears on the first page.

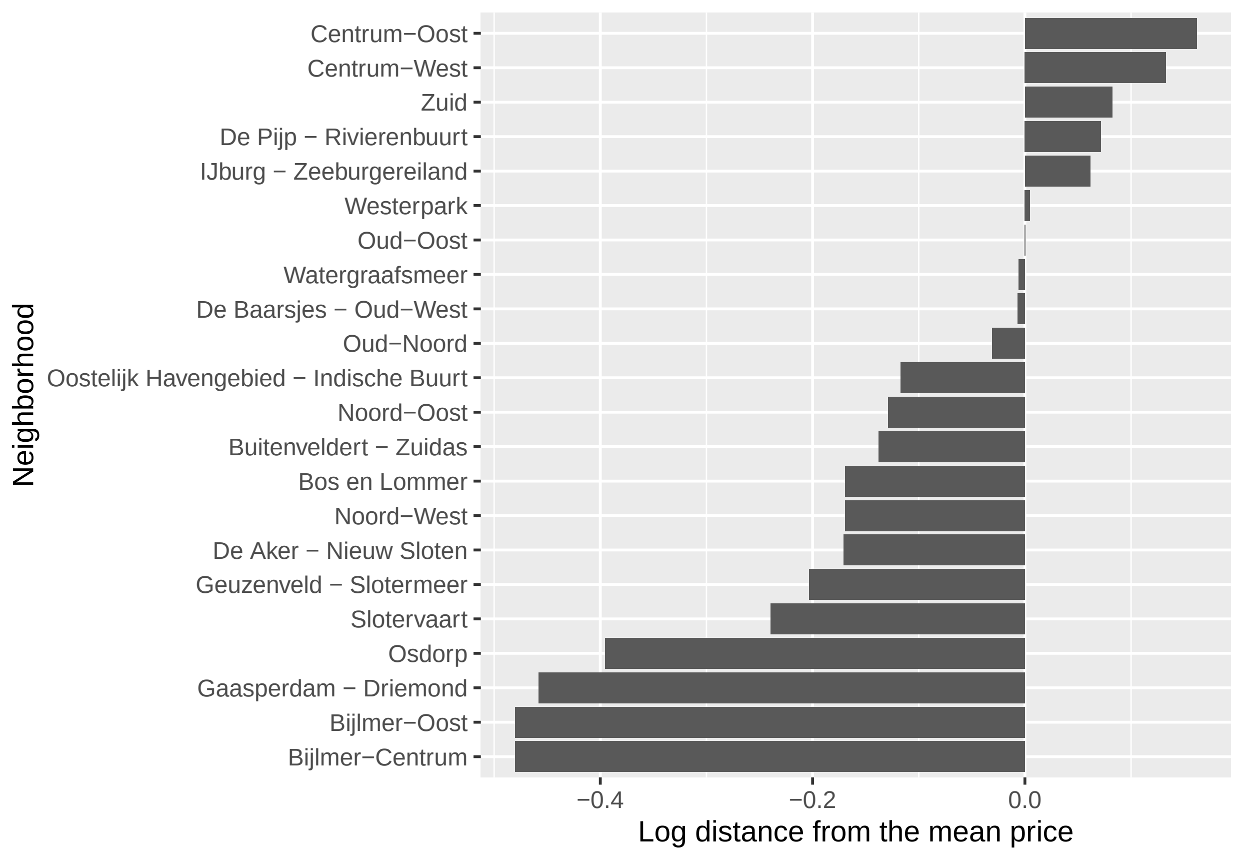
The neighborhoods appear clearly clustered, so I decided to check if there were geographical patterns in the price data.

```
## `summarise()` has grouped output by 'latitude'. You can override using the `.groups` argument.
```



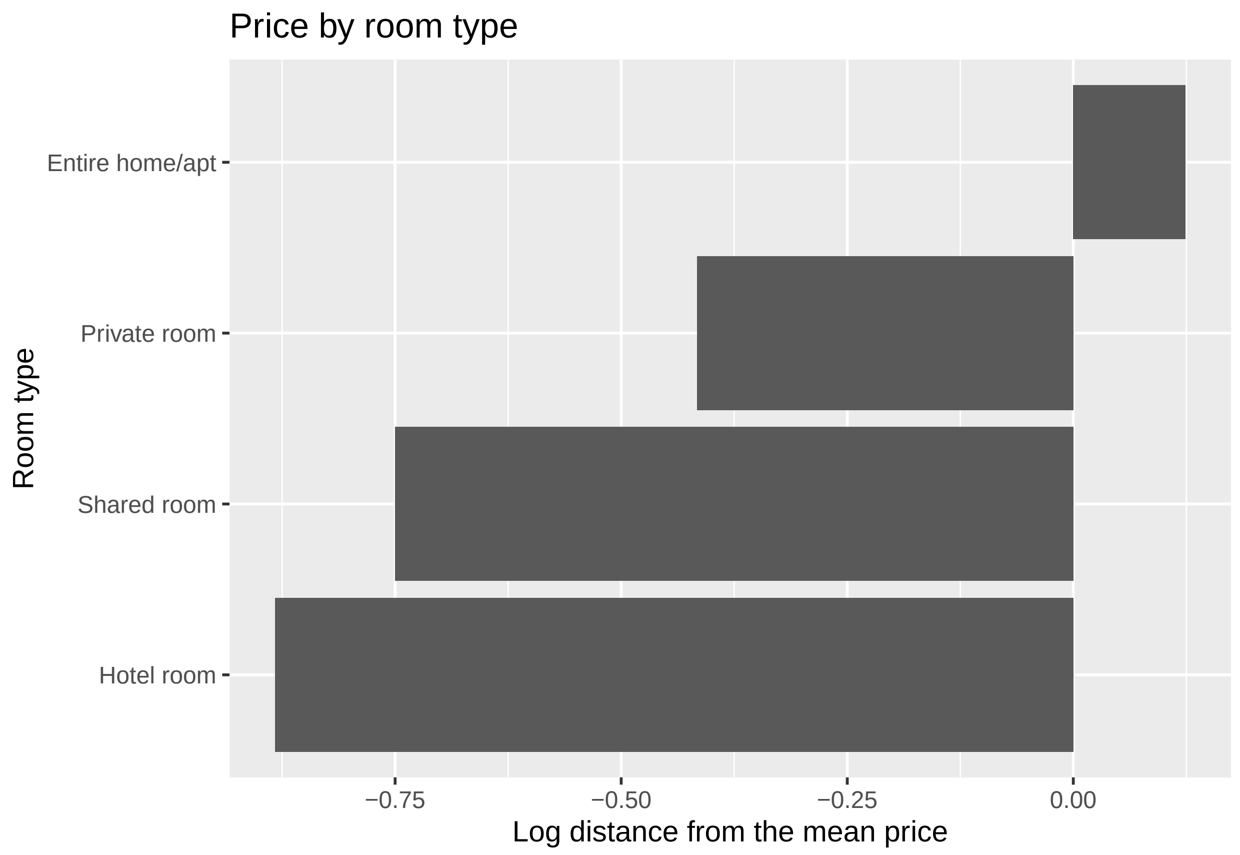
There appear to be some high and low prices regions, but the effect does not seem large.

The next geographical variable I looked at was neighborhood_cleansed, which has the information used for colors in the first graph.



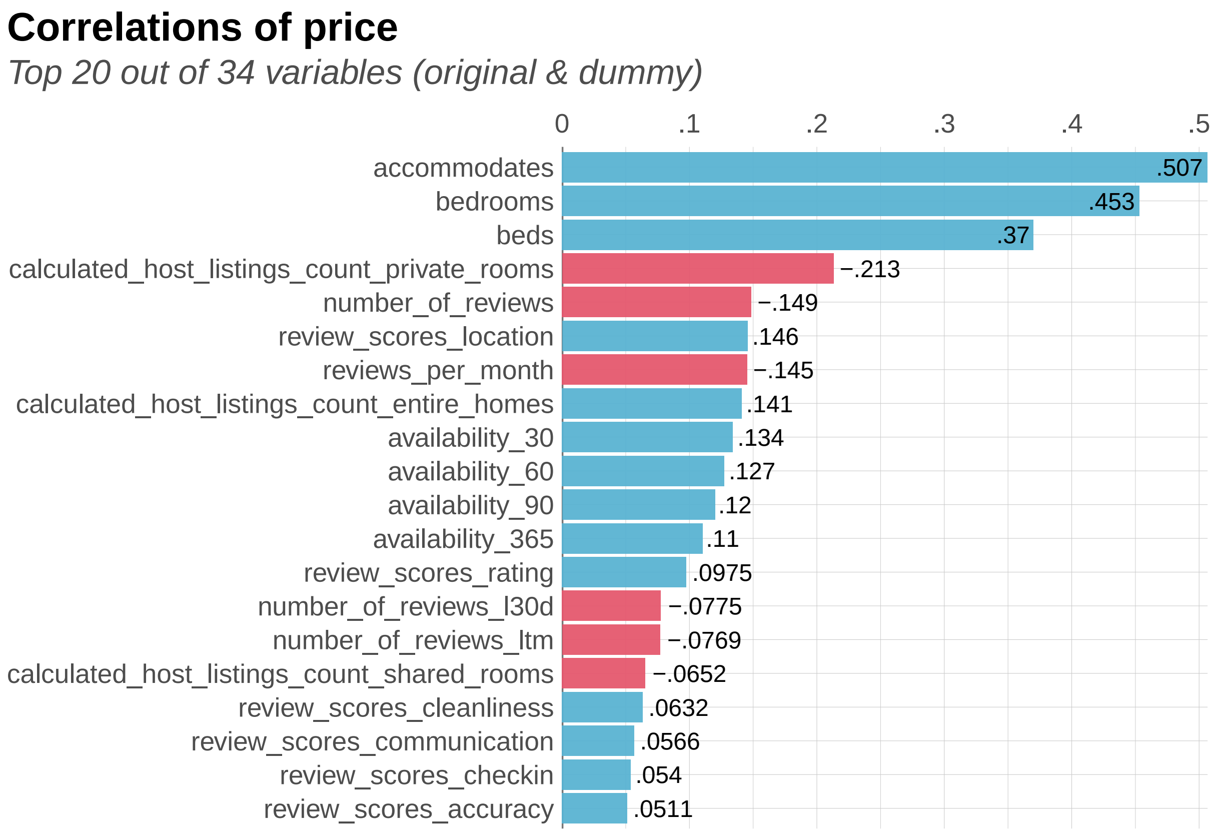
This appears to have a more significant effect than raw location data.

The next variable I checked was room type.



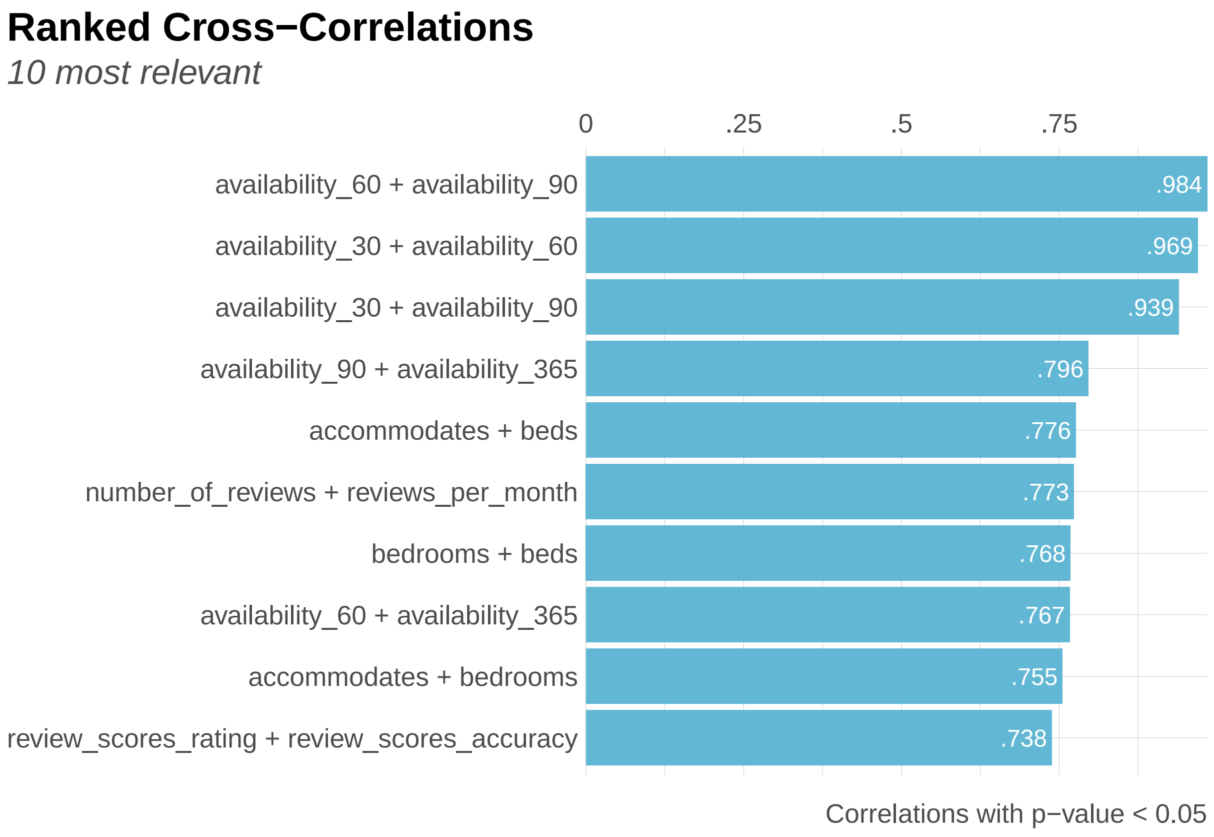
It appears only entire properties have above average prices.

For numeric variables I calculated the correlation between price and the rest of the variables.



A significant number of these variables also have high correlation between each other, for example the top 3 also show up when checking for cross-correlation. The same thing happens when looking at the availability columns or the ones related to reviews. This helps simplify potential models as it's not necessary to include every variable with high correlation to price.

```
## Returning only the top 10. You may override with the 'top' argument
```



After this initial review of the variables I started building models, testing performance using 10-fold cross validation. The first model I tested used only location information (neighbourhood_cleansed, longitude, latitude) and served as a benchmark for the rest of the models. The regularization penalty was tuned via cross validation but the optimal value was 0, that is, no regularization. This initial model returned a mean RMSE of 0.545.

Next I tested property type and room type as predictors, which decreased the RMSE to 0.496. Once again the best regularization penalty was 0.

My next attempt was combining the 2 models which did not affect the RMSE (0.496).

Because availability data appeared to have high correlation to price, I added that to the combined model, but it actually increased the RMSE to 0.467

Adding `accomodates`, which also had a high correlation with price decreased the RMSE to 0.419, by far the best of the linear models.

After this step I tried adding `calculated_host_listings_count_private_rooms` which had the highest correlation with price outside of the already included predictors (and the columns highly correlated with them).

After this I moved onto gradient boosted trees, using the same predictors as the best linear model. The model was also tuned via cross validation coupled with grid search for several parameters (mtry, number of trees and learn rate). During training the best model returned an RMSE of 0.388, a significant improvement over the best linear model.

Results

After finalizing the workflows for each model with the selected tuning parameters, I ran both models on the test set. Both models used seven predictors (property_type, room_type, neighbourhood_cleansed, longitude, latitude, availability_30 and accommodates). The best linear model's RMSE was 0.407, while the XGBoost model had an RMSE of 0.389. Surprisingly, the linear model actually outperformed the RMSE obtained during cross validation, while the boosted trees model performed very closely to how it did during training. This might indicate that the boosted trees model might have been overfitting to the training set, at least relative to the linear model, but it's hard to be conclusive with only a single test.

Limitations

While the XGBoost model gave better results, these are not easily interpretable, and boosted trees models took longer to tune compared to linear models. Having said that, once the final tuned model is generated, generating predictions does not take close to the amount of time tuning took, which makes the final model much more usable.

Future work

Because Airbnb constantly uploads the new compiled listings on their site, it's possible to continue to refine the model with new data or test how it runs using the data from other cities. It could be interesting to see what predictors are important across datasets, and not just locally relevant, as it could allow more generalizable models to be built. One thing that has a lot of potential is building an ensemble model using multiple models to improve these results, for example using variables not considered for this analysis, like text fields (property descriptions, names, etc.). This can be done using the tidymodels framework without having to rework the modeling process, which makes it an attractive idea to test.