Data Analysis 3: Assignment 2

Muhammad Arbash Malik – 2202071
GitHub

1. Introduction:

The purpose of this project was to help a company in setting their price for their newly launched apartments in Lisbon, Portugal. This company operates and deals with small and mid-sized apartments hosting 2-6 guests. The goal was achieved through building multiple predictive models based on the data we extracted from Inside Airbnb. There were 5 models that were created, namely OLS Linear Regression, LASSO, Random Forest, Cart, and GBM, and their performance was evaluated based on their RMSE, and the final price recommendation made was based on the model with the lowest RMSE. These models consider variables like property type, neighborhoods, and whether they have certain amenities. The variables that were fed into the machine learning algorithms were shortlisted from the most complex model through LASSO, where it removed the variables with zero coefficients.

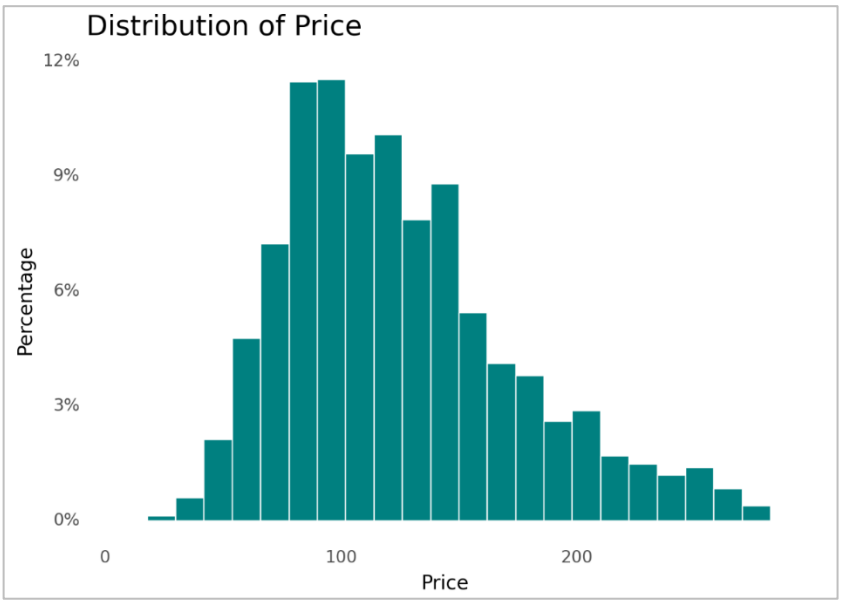
2. Data Cleaning and Data Preparation:

The data set that I initially downloaded from the Airbnb website contained 22,605 listings, however it had to be cleaned and filtered down to only the listings that were relevant and similar to the apartments that my company was about to launch, making sure that the data we used in our models was a result of conscious decisions made, ensuring that we were not feeding any irrelevant data into the models that made us compromise on the quality and accuracy of the models. Hence, a lot of time and detailed thought process was spent on the data cleaning and data preparation. But before performing any data cleaning, I saved the dataset to my GitHub repository to read it directly in my code.

The first decision I made was to filter for the number of accommodates between 2 and 6 so that the data is more relevant. The data set had a lot of irrelevant property types for our business objective for e.g. *room in serviced apartment, private room in home, room in hostel* etc. Since our company wants to operate apartments, it makes sense to filter for property types that you can book entirely. Using my domain knowledge, I know that the number of bathrooms is an important feature for predicting the apartment price, so I extracted that information as a numeric variable from a column named *bathroom_text*.

The next hurdle in the data cleaning was the amenities column, which had a list of values. The main problem with creating dummy variables for each amenity was that there were over 2000 unique values in that column, and to tackle this problem, I figured that I could group similar amenity values into a common category amenity, which then can be used in modelling. So, for example, there were a lot of values for Wi-Fi "*Fast Wifi 123 Mbps"*, "*Fast Wifi 512 Mbps"* with only the numbers changing, and I grouped them into Wi-Fi category. The same approach was used for all the amenities creating 47 dummy variables for each amenity.

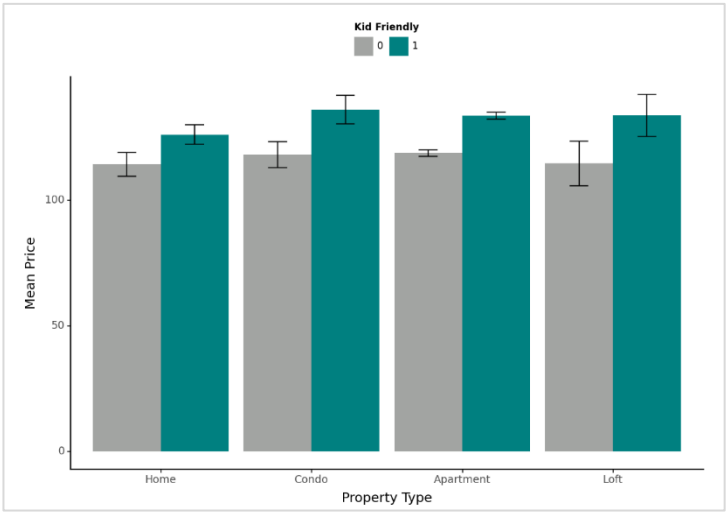
Moving further, I cleaned the important variables to get them in a form that can be fed into the models. Firstly, the target variable, *price*, was cleaned by removing the $ sign and making it into a integer column. I also converted the True/False values into 1s and 0s for *instant_bookable* and *host_is_superhost* variable. After that I changed the values in the property type to make them cleaner and combined the property types to "Home", "Apartment", "Loft", and "Condo" as per my domain knowledge. I changed some of the variables to category type and dropped irrelevant variables that I did not need. For the final step, I checked how many missing values my dataset had. Only 3 variables had missing values namely, *n_beds, n_bathrooms, n_review_scores_rating*. Since, they had approximately 2000 missing values combined, I dropped them instead of creating flag variables since we still had sufficient observations for our models. The final step was to filter out extreme values in my target variable so that our model has a fair data to work with. The price distribution after filtering out extreme values can be seen below.

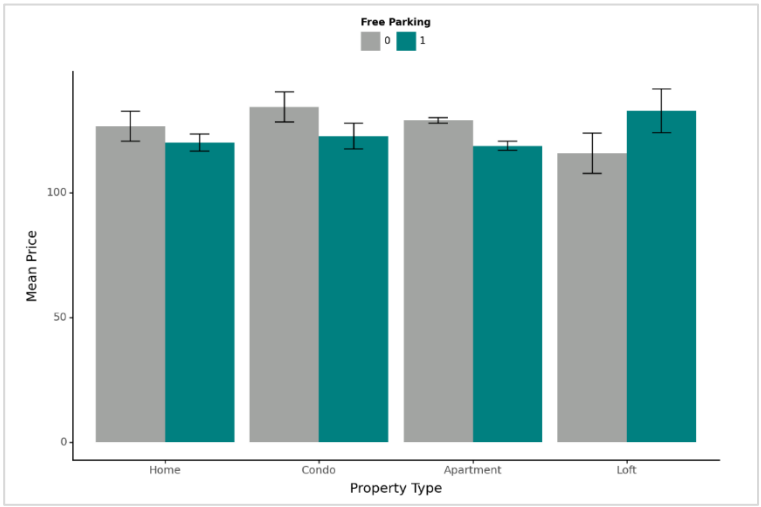


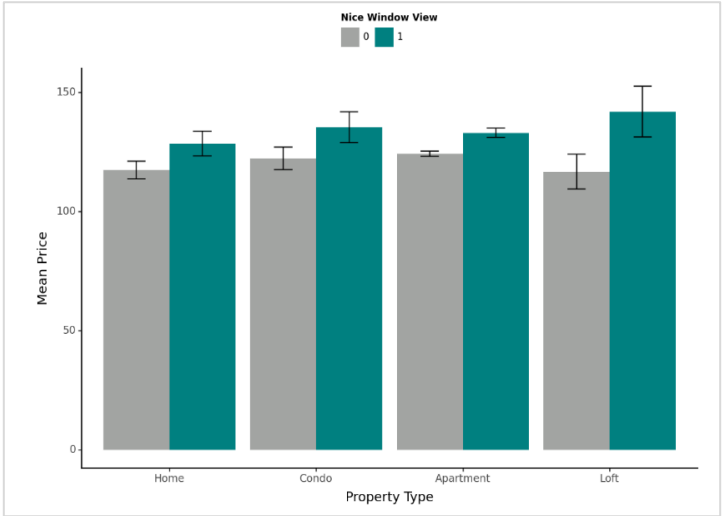
3. Modelling Decisions:

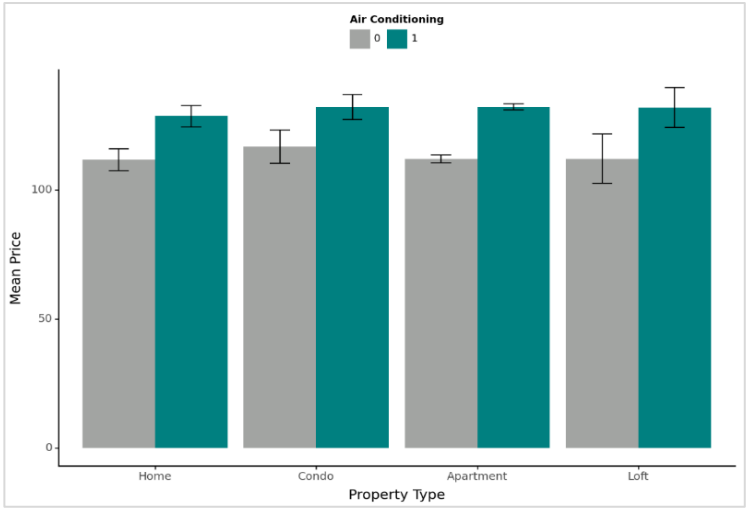
3.1 Lasso:

The next step in the process was to select our predictor variables. I chose to select my predictor variables through Lasso shortlisting. I first categorized my predictor variables into variables such as *basic, reviews, dummies,* and *all_interactions*. Then I combined them and used them in Lasso using the Naive Grid Search method. The shortlisting also helped identify some important interactions.









For the interaction terms given by LASSO, boxplots of property types against price were plotted, conditioned on amenities to see the impact it had on the prices of listings.

3.2 Cross Validation & Holdout

After all the data cleaning there were 11,365 observations remaining. Of these listings, 30% were randomly selected as a holdout set (test set), while the rest were used as a training set. The training set underwent 5 test fold cross validation, allowing our models to perform more efficiently and provide the coefficients with the least amount of overfitting. The result of these cross-validated RMSEs were used to then evaluate the model performance and was used as primary criteria to select the best model.

4. Models

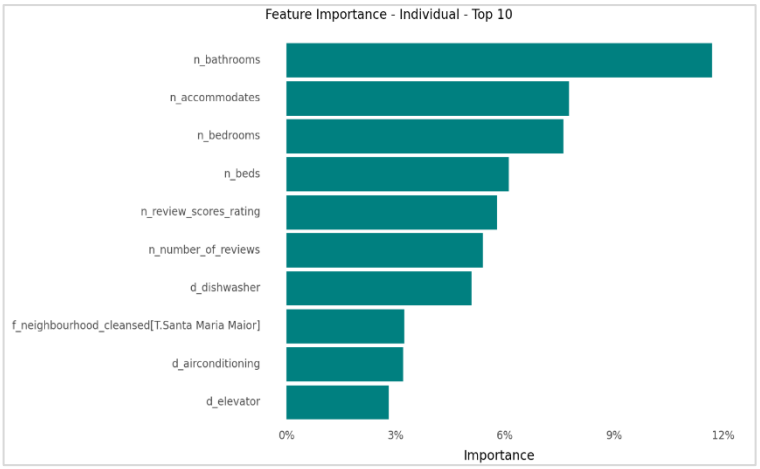
4.1 Model 1: Random Forest:

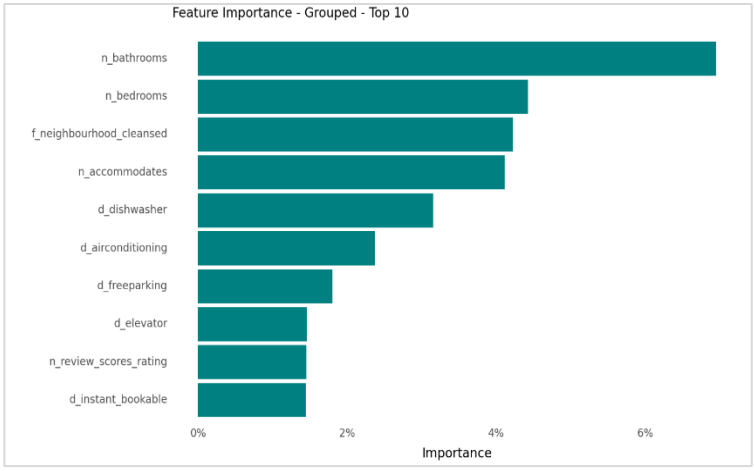
For the random forest model, there were tuning parameters provided It is a supervised learning algorithm where the end result is a combination of different trees, that were trained with the bagging method. So, the end result is essentially a result of averages of numerous trees, meaning that it is quite accurate of the actual representation. This was cross validated as well. The tuning parameters that I provided for the first one was as following:

- Maximum number of features were from: 10, 12, 14, 16.
- Minimum Node Size was from: 5, 10, 15.

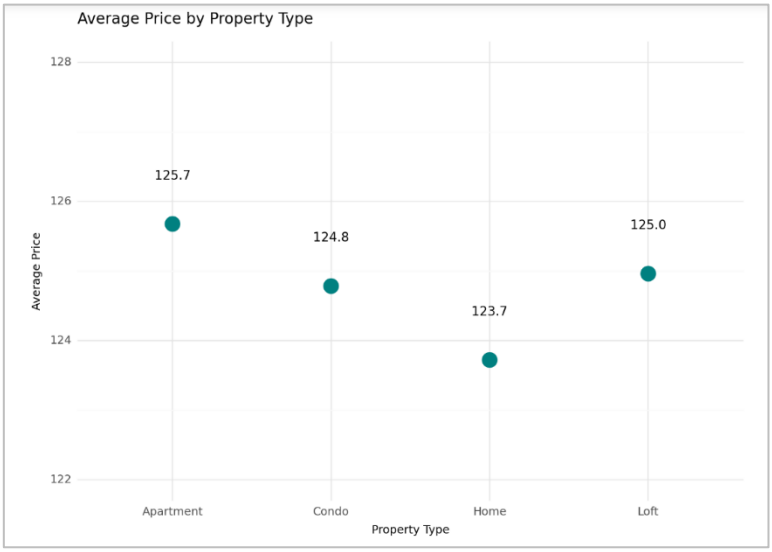
The number of features range was given according to the theoretical recommended number of features which was 13.19 for me. This model performed the best when the maximum number of features was equal to 16 and the minimum node size equals to 5.

Variable importance graphs were constructed that showed which variables were the most important in predicting the prices. First these were done without any grouping, and during the second iteration some of the variables were grouped together. For both of the graphs I only used the top 10 features so it is easier for getting insights as well.





Partial dependance plots were also constructed from the constructed random forest pipeline. It is a good insight to see on average, how our target variable changes with certain predictor variables. For the plots, I used property type and accommodates as the predictor variables.



From the number of accommodates graphs, there is a very small increase in price when the number of accommodates increases from 2 to 3. But there is a drastic price increase when the number of accommodates go from 3 to 4. Comparing the property types, the cheapest property type is Home while the most expensive property type is Apartment, although the difference is just 2 Euros.

4.2 Model 2: OLS:

For the OLS model, I first planned to run 3 models where the next model would be more complex in turn of addition of more predictor variables (see below).

```
## basic variables
basic_vars = ["f_property_type","n_accommodates","n_bedrooms","n_bathrooms","n_beds","f_neighbourhood_cleansed"]

## reviews
reviews = ["n_number_of_reviews","n_review_scores_rating",]

## dummies
dummies = list(clean_df.filter(regex="^d_.*"))

## interactions
# for interactions I interacted property types and accommodates with all the dummies
interactions = []

for dummy_col in clean_df.filter(regex="^d_.*"):

    interaction_f = 'f_property_type' + '*' + dummy_col + ' + '
    interactions.append(interaction_f)

    interaction_n = 'n_accommodates' + '*' + dummy_col + ' + '
    interactions.append(interaction_n)

all_interactions = ''.join(interactions)
all_interactions = all_interactions[:-3]


# Predictor variables

predictor_variables1= basic_vars
predictor_variables2= basic_vars + reviews + dummies
predictor_variables3= basic_vars + reviews + dummies + all_interactions
```

But thinking in terms of actual business objectives where you are constrained by deadlines, I ran a Lasso model that is mentioned above to shortlist variables from the most complex combination of predictor variables *(predictor_variables3)***.** The number of variables before Lasso was 155 and after shortlisting the variables were 113.

4.3 Model 3: Lasso:

A Lasso predictive model was also created to compare. For this, I combined Lasso (L1) and Ridge (L2) together. Lasso and Ridge are some of the simple techniques to reduce model complexity and prevent over-fitting which may result from simple linear regression. I also gave a pure Lasso penalty of 1 (*l1_ratio = 1)*.

4.4 Model 4: CART:

How CART with pruning works is that the training data is split randomly into several folds for instance 10, after which the pruning level is selected. 9 folds are used for creation of 9 pruned trees and an error is estimated on the last fold. The second step is repeated until all the pruning levels are used. The smallest error is found, and the pruning level assigned to it is used. Until the pruning level is reached, all terminal nodes in the lowest tree level are removed and decision class is assigned to parent node. Decision value is equal to class with higher number of cases covered by node. For this model, I used an automatic complexity pruning path and filtered the alphas above 2 to ensure better performance.

4.5 Model 5: GBM:

Gradient boosting machine is a model which creates trees, and, with each iteration, it improves upon the previous tree. Hence, in principle this method is better than CART. In our predictive models, this was also the case in terms of the RMSEs of the models.

5. Model Performance

As per the table below, the best model performance was given by GBM, followed by OLS and LASSO. Surprisingly, random forest came in 4th overall while CART was the least performing model.

The recommendation to the company according to the final results is that they should invest in apartments and lofts, which is loaded with at least the basic amenities (the more the better), with the option of accommodating the maximum possible number of persons, as it will allow them to price their listings the highest and potentially earn highest profit.

	Model	CV RMSE
1	Random Forest	39.61
2	LASSO	38.74
3	OLS	37.50
4	CART	42.87
5	GBM	36.92

6. Prediction

I used our best model for prediction and compared it to the actual prices on the holdout set.