Introduction to Artificial Intelligence

Project: Predicting AirBnB Prices

Mikołaj Jagielski 303860

Michał Łezka 303873

1. Introduction

The aim of this project is to predict Airbnb prices using machine learning algorithms: Linear Regression and Random Forest. Linear Regression algorithm is a simple yet effective algorithm that predicts the price of a new listing by finding relationship between Airbnb features and other independent variables and the price, which is dependent variable in this case. On the other hand, Random Forest algorithm builds a set of tree-like models of decisions where outcomes of each tree are based on the most relevant features for predicting prices, then those outcomes are aggregated into a single result. We applied these algorithms to a dataset of Airbnb listings of offers from many European cities with various features such as number of bedrooms, room type, cleanliness, and so on. After pre-processing the data, we trained and tested the models to evaluate their performance using appropriate metrics and techniques.

2. Algorithms

- The Linear Regression algorithm is a very simple regression algorithm that finds corelation between independent and dependent variables. The algorithm will attempt to find the best-fit line that minimizes the difference between the predicted prices and the actual prices in the training data. The line's equation represents the relationship between the features and the predicted price. Its biggest issue in our context is its weakness in working with nonnumerical data and assumption that a linear relationship between the features and the target variable, which may not always hold true in real-world scenarios.
- Random Forest is an ensemble machine learning algorithm that combines multiple decision trees to improve the accuracy and generalizability of the predictions. The algorithm works as follows:
    1. Randomly select a subset of the training data.
    2. Randomly select a subset of the input features.
    3. Build a decision tree using the selected data and features.
    4. Repeat steps 1-3 to create a set number of decision trees.
    5. When making a prediction for a new data point, pass it through each decision tree in the forest and take the average or majority vote of the predictions as the final prediction.

  The Decision Tree algorithm is an algorithm that builds a tree-like model of decisions and their possible consequences. Each node in the tree represents a decision based on one of the features, and each branch represents one of the possible outcomes of

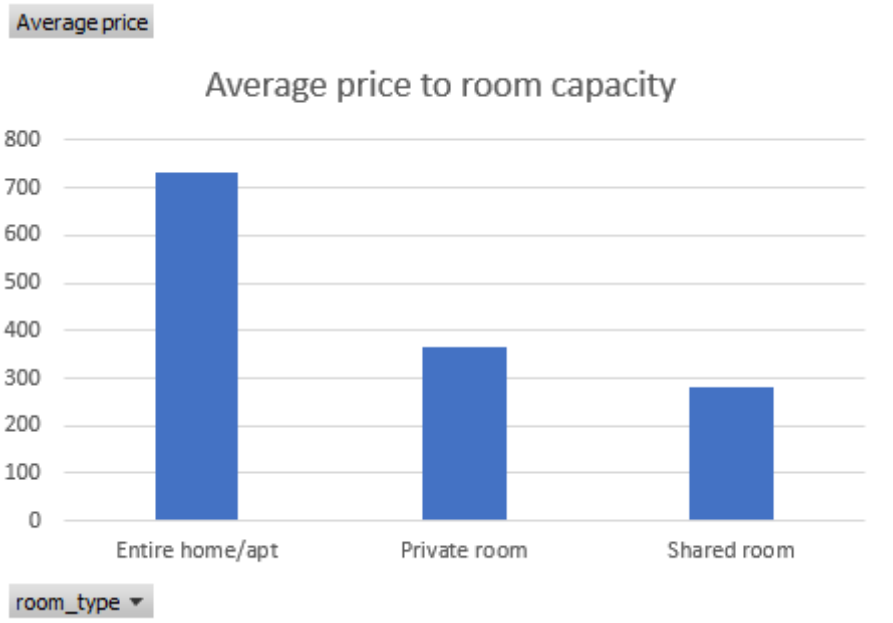
the decision. In the context of predicting Airbnb prices, the algorithm can be used to build a decision tree based on the features that are most relevant for predicting prices. The algorithm then uses the decision tree to predict the price of a new listing by traversing the tree based on the features of the listing and the decisions made at each node.

As both Linear Regression and Random Forest algorithms have their own strengths and weaknesses, we consider this pair very good to implement and compare.

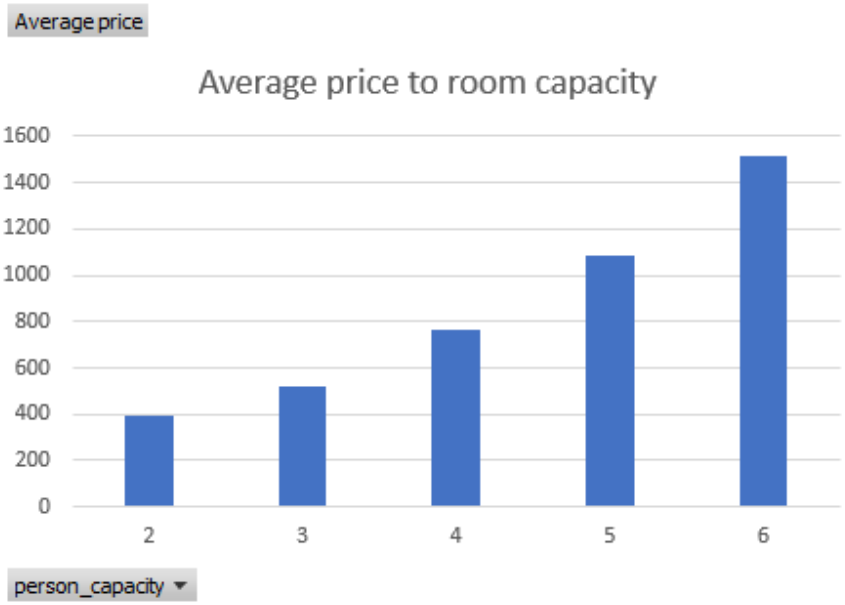
3. Data

We have noticed a few (and some rather obvious) tendencies. Some of the more obvious would be:

- The more privacy/space, the more expensive rental is

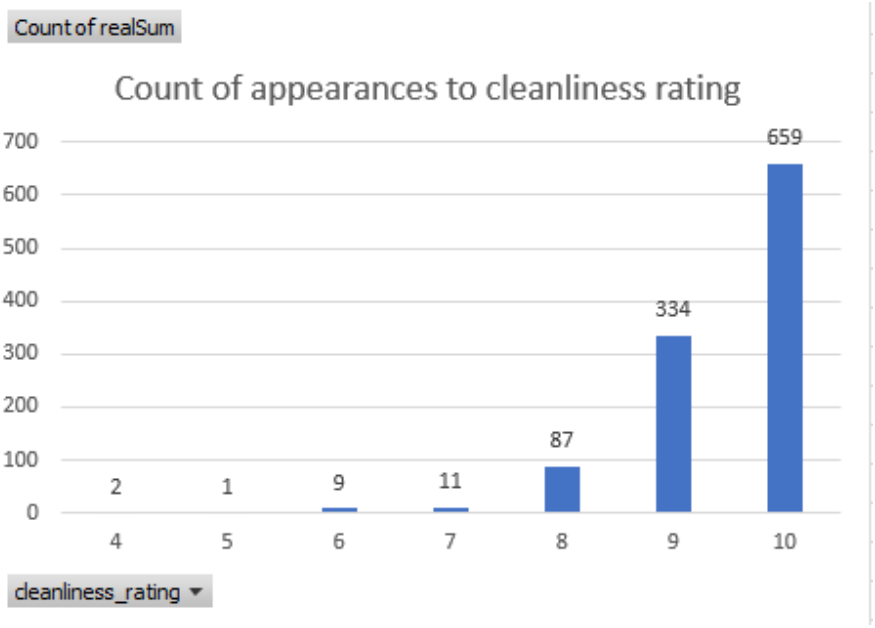


- The higher capacity, the more expensive



- Cleanliness ratings are however more interesting:

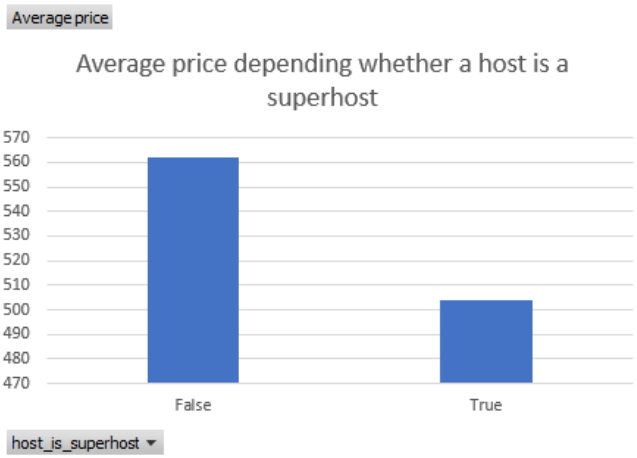
As we can see, there are some outliers, most importantly 4-star reviews having higher prices on average than 5-star reviews. When we analyse this phenomenon, we can see that it is caused by 4 and 5-star cleanliness reviews are hugely underrepresented:



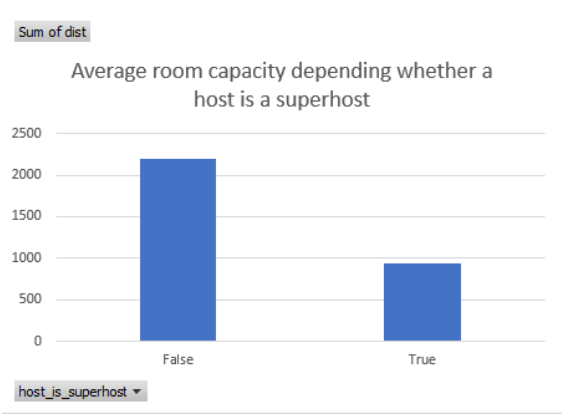
As we can see, we have a major problem in the dataset: imbalance caused by lack of samples with specific properties, like mentioned above cleanliness ratings, if host a super host, room capacities, and a huge dominance of non-business trips. Another example of such imbalance would be the number of superhosts in Amsterdam compared to regular hosts:

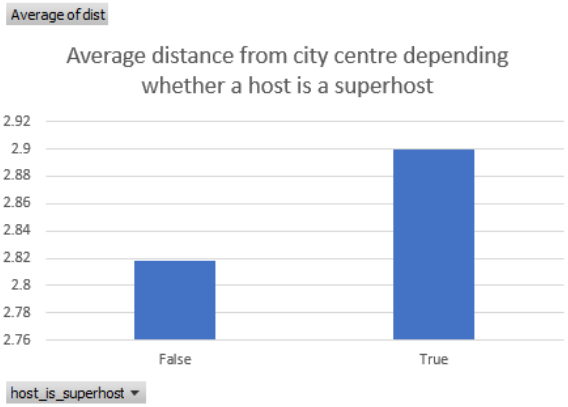
host_is_superhost	Count of Rows
False	780
True	323

Which causes some rather unexpected results:



After a deeper analysis we have observed that superhosts tend to rent smaller apartments that are further away from the city centre:





We plan to address this problem using oversampling, that is - generating synthetic samples for the minority class to balance the dataset, specifically called SMOTE (Synthetic Minority Over-sampling Technique). We can randomly generate new samples for the missing listings by interpolating between existing samples or generating new samples in regions where the missing listings are underrepresented. This can balance the dataset and prevent the model from being biased towards the majority class. We decided to split the data in chunks for each city and database to see how specific set of features, which vary in different cities, affects the results. It is also very important to walk-through the whole data and eliminate invalid data like cleanliness rating above 10 etc.

4. Test and evaluation

To compare the how well the algorithms perform we plan Root Mean Squared Error (RMSE). It is a commonly used evaluation metric for regression algorithms that measures the square root of the average squared difference between the predicted and actual values. Lower RMSE values indicate better performance. The comparison and results will be visualised - we are planning to display the results using appropriate graphs and charts from Seaborn library. The RMSE will be compared on charts for each city.