**Group 3 – Final Report**

Trieu Vo

Yash Tadiyal

Ashutosh Kumar Singh

College of Professional Studies, Northeastern University

EAI 6000: Fundamentals of Artificial Intelligence

Prof. Kasun Samarasinghe

December 10th, 2021

**Table of contents**

[**I. The use-case 3**](#_heading=h.gjdgxs)

[**II. Explanation of the data 3**](#_heading=h.30j0zll)

[**III. EDA and Feature Engineering 6**](#_heading=h.1fob9te)

[1. Data Cleaning 6](#_heading=h.3znysh7)

[2. Data Manipulation 6](#_heading=h.2et92p0)

[*a. Categorical features 6*](#_heading=h.tyjcwt)

[*b. Numerical features 7*](#_heading=h.3dy6vkm)

[3. Data Visualization 8](#_heading=h.1t3h5sf)

[4. Data Reduction 9](#_heading=h.4d34og8)

[**IV. Models Building 10**](#_heading=h.2s8eyo1)

[1. Preparation 10](#_heading=h.17dp8vu)

[2. Feature Transformation 11](#_heading=h.3rdcrjn)

[3. Models Building 11](#_heading=h.26in1rg)

[*1) Linear Regression 11*](#_heading=h.lnxbz9)

[*2) Decision Tree Regression 12*](#_heading=h.35nkun2)

[*3) Random Forest Regression 12*](#_heading=h.1ksv4uv)

[*4) XGBoost 12*](#_heading=h.44sinio)

[**V. Hyperparameter Tuning 13**](#_heading=h.2jxsxqh)

[1) Linear Regression 13](#_heading=h.z337ya)

[2) Decision Tree Regression 13](#_heading=h.3j2qqm3)

[3) Random Forest Regression 13](#_heading=h.1y810tw)

[4) XGBoost 13](#_heading=h.4i7ojhp)

[**VI. Performance Measurement 13**](#_heading=h.2xcytpi)

[R2 score 13](#_heading=h.1ci93xb)

[1) Linear Regression 14](#_heading=h.3whwml4)

[2) Decision Tree Regression 14](#_heading=h.2bn6wsx)

[3) Random Forest Regression 15](#_heading=h.qsh70q)

[4) XGBoost 16](#_heading=h.3as4poj)

[**VII. Model Explanation 17**](#_heading=h.1pxezwc)

[1) Linear Regression 17](#_heading=h.49x2ik5)

[2) Decision Tree Regression 18](#_heading=h.2p2csry)

[3) Random Forest Regression 19](#_heading=h.147n2zr)

[4) XGBoost 21](#_heading=h.3o7alnk)

# I. The Use Case

As more people choose Airbnb as their primary accommodation provider, Airbnb has effectively challenged the traditional hotel business. Since its establishment in 2008, Airbnb has experienced tremendous development, with the number of rentals listed on its website increasing at an exponential rate each year. We use the Inside Airbnb dataset to identify significant trends in customer interest and predict the pricing of rentals in the US and Europe. This problem is similar to a classical use case of machine learning: house price prediction. We can then provide solutions to business difficulties.

There are several questions that we want to find answers to:

* How to predict the price for each listing?
* Can we find out how the features of listings affect the price locally and globally?

# 

# II. Explanation of the data

The Inside Airbnb dataset contains approximately 1.2 million listings from various countries across the world. It also has 84 distinct listing characteristics.

In this project, we use the dataset of US listings and European listings in the Inside Airbnb dataset. It contains 821,335 listings and we select 27 useful features to explore.

There are 27 columns in this dataset and their description are as follows:

* ***accommodates***: The maximum capacity of the listing.
* ***amenities***: Furniture that the listing has.
* ***availability\_365***: The availability of the listing 365 days in the future as determined by the calendar. Note a listing may not be available because it has been booked by a guest or blocked by the host.
* ***bedrooms***: The number of bedrooms.
* ***beds***: The number of bed(s).
* ***calculated\_host\_listings\_count***: The number of listings the host has in the current scrape, in the city/region geography.
* ***host\_has\_profile\_pic***: boolean [t=true; f=false].
* ***host\_id***: Airbnb's unique identifier for the host/user.
* ***host\_identity\_verified***: boolean [t=true; f=false].
* ***host\_name***: Name of the host. Usually just the first name(s).
* ***host\_response\_rate***: the rate the host respond to their messages.
* ***host\_response\_time***: the amount of time the host takes to reply to their messages.
* ***id***: Airbnb's unique identifier for the listing.
* ***instant\_bookable***: [t=true; f=false]. Whether the guest can automatically book the listing without the host requiring to accept their booking request.
* ***last\_review***: The date of the last/newest review.
* ***latitude***: Uses the World Geodetic System (WGS84) projection for latitude and longitude.
* ***longitude***: Uses the World Geodetic System (WGS84) projection for latitude and longitude.
* ***maximum\_nights***: maximum number of night stay for the listing (calendar rules may be different).
* ***minimum\_nights***: minimum number of night stay for the listing (calendar rules may be different).
* ***name***: Name of the listing.
* ***neighborhood***: the neighborhood of the listing.
* ***number\_of\_reviews***: The number of reviews the listing has.
* ***property\_type***: Self-selected property type. Hotels and Bed and Breakfasts are described as such by their hosts in this field.
* ***reviews\_per\_month***: The number of reviews the listing has over the lifetime of the listing.
* ***review\_scores\_rating***: average rating.
* ***room\_type***: [ Entire home/apt | Private room | Shared room | Hotel ].

***Target variable***:

***price***: the daily price of a listing.

# III. EDA and Feature Engineering

## 1. Data Cleaninggo

We calculate the number and the percentage of missing values per feature and find features that have missing values. Then we consider whether we should fill those missing with previous values of that feature or remove them completely. After trying different combinations with other later steps, we decide to remove all the rows that have missing values. This helps us decrease the number of observations from over 800,000 to over 200,000, which saves us a lot of time when training models and improving model accuracies.

## 2. Data Manipulation

We evaluate categorical features and numerical features separately to make sure that they have proper data types before importing them into models.

### a. Categorical features

We get categorical features, evaluate them and convert them to numerical features. Those categorical features are amenities, host\_has\_profile\_pic, host\_identity\_verified, host\_name, host\_response\_rate, host\_response\_time, instant\_bookable, last\_review, name, neighborhood, price, property\_type and room\_type.

***amenities***: because they are lists of amenities the listings have, we try to convert them to the number of amenities those listings have

***host\_has\_profile\_pic, host\_identity\_verified, instant\_bookable***: because these features have 2 boolean values ‘t’, ‘f’ so we convert them into binary values ‘0’, ‘1’.

***host\_response\_rate***: we remove the ‘%’ sign and convert them into real numbers from 0 to 1.

***host\_response\_time***: it has 4 classes: a few days or more, within a day, within a few hours, within an hour. So we use LabelEncoder to convert them into 0, 1, 2, 3 based on its priorities.

***price***: We remove the ‘$’ sign and convert them into real numbers.

***room\_type***: this feature has 4 classes (entire\_home\_apt, hotel\_room, private\_room, shared\_room), so we used a function called get\_dummies() to convert them into dummy variables. In the end, we had 4 more new columns, and we remove the original feature ‘room\_type.

Clean names of columns: we clean the names of columns again after adding new dummy columns, and we get 3 new columns (room\_type\_entire\_home\_apt, room\_type\_hotel\_room, room\_type\_private\_room)

### b. Numerical features

We get all numeric features, including categorical features that we turn into numeric features. Then we deal with outliers. At first, we try to use InterQuartile Range to detect and remove outliers of each feature that are outside of Q1 - 1.5 \* IQR and Q3 + 1.5 \* IQR. However, this step removes so many outliers that the dataset reduces to only 50,000 observations left and decreases the accuracies of models we build later. So we decided to modify the range a little bit. We calculate the upper bound (97.5%) and lower bound (2.5 %) of values of each feature, and remove observations that have values outside those bounds. After doing this step, the remaining dataset has around 150,000 observations, which is pretty suitable to train and test ML models.

## 3. Data Visualization

We use the histogram to visualize the distribution of each feature in the dataset to assess normality. Here we can see that most of the features don’t have a normal distribution. They are usually skewed left or skewed right. Some of them only have 1 value after removing missing values and outliers. We will remove those features before importing the dataset into ML models.

A picture containing window, shoji, crossword puzzle

Description automatically generated

## 4. Data Reduction

We can see that there are some features with high cardinalities, such as ‘host\_id’ and ‘id’. Therefore, we remove them.

From Data Visualization we also see that there are some features that have only 1 value, such as 'host\_has\_profile\_pic', 'room\_type\_hotel\_room', 'room\_type\_shared\_room'. They will not make changes to the price of listings. So we remove them.

So after removing unnecessary features, we have 20 features left. We visualize them again to make sure that they look all right.

A picture containing crossword puzzle

Description automatically generated

# IV. Models Building

## 1. Preparation

We separate predictor variables and the target variable (‘price’ column). Then we create the training set and the test set with the ratio 70%:30%, respectively. The training set has 106,731 observations with 18 predictor features. The test set has 45,742 observations with 18 predictor features.

## 2. Feature Transformation

Because we have a lot of predictor features that don’t have a normal distribution, and also there are still several outliers in the dataset, we decide to try to use StandardScaler and RobustScaler to deal with these problems.

StandardScaler standardizes features by removing the mean and scaling to unit variance. It uses the z-score formula. Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data.

RobustScaler is a scaler that is robust to outliers. It removes the median and scales the data according to the quantile range. Standardization of a dataset is a common requirement for many machine learning estimators. However, outliers can often influence the sample mean/variance negatively. In such cases, the median and the interquartile range often give better results.

After trying both methods and observing the accuracies of models, we decide to use RobustScaler to scale predictor features to get the best results.

## 3. Models Building

### 1) Linear Regression

We build a linear regression model to see whether the model can fit the dataset well or not. Because we have 19 predictor variables, we don’t hope too much that this linear regression model can give us a high result.

After training, we get model coefficients as follow:

Text, letter

Description automatically generated

We can see that ***accommodates, bedrooms, latitude, longitude*** affect the price most.

### 2) Decision Tree Regression

We build a Decision Tree model to predict the price of a listing. The thing we are concerned about is that this kind of model usually gets overfitting. Therefore it may have a high result on the training set and a low result on the test set.

### 3) Random Forest Regression

To solve the overfitting problem of the Decision Tree model, we build a Random Forest model. We try different hyperparameters to get the best results from this model.

### 4) XGBoost

Also, XGBoost is a boosting algorithm that was used to win in many Kaggle challenges years ago. We build an XGBoost model to predict the price of the listing in the hope that it will also give us high results.

# 

# V. Hyperparameter Tuning

## 1) Linear Regression

For the Linear Regression model, we use the default hyperparameter.

## 2) Decision Tree Regression

We try to change different hyperparameters such as criterion, max\_depth, random\_state, and finally, we set criterion = "squared\_error", max\_depth = 10, random\_state = 42 because they give us the best result on the test set.

## 3) Random Forest Regression

We try to change different hyperparameters such as n\_estimators, criterion, random\_state and finally we set n\_estimators = 100, random\_state = 42, criterion = “squared\_error” because they give us the best result on the test set.

## 4) XGBoost

We try to change different hyperparameters and finally set n\_estimators = 100, learning\_rate = 0.08, gamma = 0, subsample = 0.75, colsample\_bytree = 1, max\_depth = 10 because they give us the best result on the test set.

# 

# VI. Performance Measurement

## R2 score

The R2 score is a very important metric that is used to evaluate the performance of a regression-based machine learning model. It is pronounced as R squared and is also known as the coefficient of determination. It works by measuring the amount of variance in the predictions explained by the dataset. Simply put, it is the difference between the samples in the dataset and the predictions made by the model.

If the value of the R2 is 1, it means that the model is perfect and if its value is 0, it means that the model will perform badly on an unseen dataset.

## 1) Linear Regression

On both the training set and the test set, the Linear Regression model achieves R^2 = 0.37. This is a pretty low result, but it can show us that the model doesn’t suffer from overfitting. Maybe this low result is because of underfitting and we need to try more complicated models.

We also visualize the first 25 observations in the test set to have a look at the difference between the original values and our predicted values.

Chart, line chart

Description automatically generated

## 2) Decision Tree Regression

On the training set, the Decision Tree model achieves R^2 = 0.57, which is acceptable. On the test set, the model achieves R^2 = 0.49. This result is better than what we have tried before. We try different hyperparameters and at first get the R^2 around 0.23. We think that this comes from the weakness of the Decision Tree, which is overfitting. But after decreasing the max\_depth to 10 we see an increase in the R^2 on the test set, which is 0.49.

We also visualize the first 25 observations in the test set to have a look at the difference between the original values and our predicted values.

Chart, line chart

Description automatically generated

## 3) Random Forest Regression

On the training set, the Random Forest model achieves R^2 = 0.95, which proves that the model fits the training data very well. On the test set, the model gets R^2 = 0.62. This result comes from many changes in the input features, data cleaning, data manipulation, data transformation, as well as hyperparameter tuning. Because at first, we try a lot of techniques but the R^2 is only going around 0.53. After all, we can increase it to 0.62. This is a big achievement.

We also visualize the first 25 observations in the test set to have a look at the difference between the original values and our predicted values.

Chart, line chart

Description automatically generated

## 4) XGBoost

On the training set, the XGBoost model achieves R^2 = 0.8, which is a pretty good result. On the test set, the model gets R^2 = 0.64. This result is even better than that of the Random Forest model (R^2 = 0.62). This also comes from changes in the input features, data cleaning, data manipulation, and data transformation, as well as hyperparameter tuning.

We also visualize the first 25 observations in the test set to have a look at the difference between the original values and our predicted values. We can see that the orange line is much more fit than the blue line.

Chart, line chart

Description automatically generated

# 

# VII. Model Explanation

## 1) Linear Regression

We use LIME and SHAP to interpret the model's predictions at the local and global levels. We take a random test vector from the test set and set the number of the top features as 5, the number of perturbed samples as 5000.

Chart

Description automatically generated

The comparatively higher market value of the house reflected by the provided vector (depicted by a bar on the left) can be related to the following reasons:

* The high value of ***room\_type\_private\_room*** on the negative side indicates that because it’s a private room, the price will reduce.
* The high value of ***longitude*** on the negative side indicates that the house is not in a suitable place such as city centers or crowded neighborhoods.
* The high value of ***bedrooms*** on the positive side indicates that the house has a bedroom, which is a private room and people prefer it.
* The high value of ***review\_scores\_rating*** on the positive side indicates that the house has a high rating.

Chart, bar chart

Description automatically generated

At the global level, we can see that ***longitude, bedrooms, accommodates, latitude, beds, review\_scores\_rating*** affect the price most.

## 2) Decision Tree Regression

We use LIME and Feature Importance to interpret the model's predictions at the local and global levels. We take a random test vector from the test set and set the number of the top features as 5, the number of perturbed samples as 5000.

Chart

Description automatically generated with low confidence

The comparatively lower market value of the house reflected by the provided vector (depicted by a bar on the left) can be related to the following reasons:

* The high values of ***longitude*** and ***latitude*** on the negative side indicate that the house is not in a suitable place such as city centers or crowded neighborhoods.
* The high value of ***room\_type\_private\_room*** on the negative side indicates that because it’s a private room, the price will reduce.
* The high value of ***bedrooms*** on the positive side indicates that the house has a bedroom, which is a private room and people prefer it.
* The high value of ***accommodates*** on the negative side indicates that the house is small.

Chart, bar chart

Description automatically generated

At the global level, we can see that ***longitude, accommodates, latitude, bedrooms*** affect the price most.

## 3) Random Forest Regression

We use LIME and Feature Importance to interpret the model's predictions at the local and global levels. We take a random test vector from the test set and set the number of the top features as 5, the number of perturbed samples as 5000.

A picture containing chart

Description automatically generated

The comparatively lower market value of the house reflected by the provided vector (depicted by a bar on the left) can be related to the following reasons:

* The high values of ***longitude*** and ***latitude*** on the negative side indicate that the house is not in a suitable place such as city centers or crowded neighborhoods.
* The high value of ***room\_type\_private\_room*** on the negative side indicates that because it’s a private room, the price will reduce.
* The high value of ***review\_scores\_rating*** on the positive side indicates that the house has a high rating.
* The high value of ***accommodates*** on the negative side indicates that the house is small.

Chart, bar chart

Description automatically generated

At the global level, we can see that ***longitude, accommodates, latitude, bedrooms*** affect the price most.

## 4) XGBoost

We use LIME and Feature Importance to interpret the model's predictions at the local and global levels. We take a random test vector from the test set and set the number of the top features as 5, the number of perturbed samples as 5000.

Chart

Description automatically generated with low confidence

The comparatively lower market value of the house reflected by the provided vector (depicted by a bar on the left) can be related to the following reasons:

* The high values of ***longitude*** and ***latitude*** on the negative side indicate that the house is not in a suitable place such as city centers or crowded neighborhoods.
* The high value of ***bedrooms*** on the positive side indicates that the house has a bedroom, which is a private room and people prefer it.
* The high value of ***review\_scores\_rating*** on the positive side indicates that the house has a high rating.
* The high value of ***accommodates*** on the negative side indicates that the house is small.

Chart, bar chart

Description automatically generated

At the global level, we can see that ***bedrooms, room\_type\_entire\_home/apt, accommodates, longitude, latitude*** affect the price most.