PROJECT REPORT

Airbnb NYC Price Production

Submitted towards the partial fulfillment of the criteria for award of Post Graduate In Data Analytics by Imarticus

Submitted By:

Jibin George (IL036623)

Course and Batch: PGA 25

Abstract

The Airbnb NYC price prediction project aims to develop a machine learning model that can accurately predict the rental price of a given Airbnb listing in New York City based on various features such as location, property type, and number of bedrooms.

The dataset used for this project contains information about thousands of Airbnb listings in NYC, including features such as the listing's neighbourhood, the property type, number of bedrooms, and other amenities. Using this data, various regression models were trained to predict the rental price of a given listing.

The machine learning model developed for this project utilizes various techniques such as data cleaning, feature engineering, and model selection to achieve a high level of accuracy in predicting rental prices. We use a variety of regression models, including Linear Regression, Decision Tree, Random Forest, KNN, Ada Boost, Light GBM, Cat Boost, Gradient Boost and XGBoost, to determine which one performs best on the data. The model is evaluated using various performance metrics such as mean squared error, root mean squared error, mean absolute error, R-squared, and cross-validation.

The results of this project demonstrate that the developed model can accurately predict the rental price of Airbnb listings in NYC. The model can be used by Airbnb hosts to estimate the price of their listings and by renters to get a better understanding of what they can expect to pay for an Airbnb rental in the city.

Overall, the Airbnb NYC price prediction project shows the potential for machine learning to be used to develop accurate and useful tools for both hosts and renters in the sharing economy.

Acknowledgement

I am using this opportunity to express my gratitude to everyone who supported me throughout the course of this group project. I am thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.

Further, I am fortunate to have **VijayaKumar** as our mentor. He has readily shared his immense knowledge in data analytics and guide us in a manner that the outcome resulted in enhancing our data skills.

I wish to thank, all the faculties, as this project utilized knowledge gained from every course that formed the PGA program.

I certify that the work done by me for conceptualizing and completing this project is original and authentic.

Date : Feb 28, 2023 Jibin George

Place : Chennai

Certificate of Completion

I hereby certify that the project titled "**Airbnb NYC Price Prediction** " was undertaken and completed under my supervision by Jibin George from the batch of PGA-25

Mentor : VijayaKumar Date : Feb 28, 2023

Place : Chennai

Table of Contents

CHAPTER 1 : INTRODUCTION

1.1 Title & Objective of the study

Airbnb NYC Price Prediction

The objective of the Airbnb NYC price prediction project is to develop a machine learning model that accurately predicts the rental price of Airbnb listings in New York City. The model utilizes various features such as location, property type, and amenities to make predictions. The aim is to provide hosts with a tool for setting competitive prices and renters with an estimate of the cost of their stay. The project also aims to provide insights into Airbnb's pricing strategies and the potential for data-driven approaches to improve the sharing economy. Overall, the goal is to enhance the user experience on the platform and help facilitate more efficient and effective transactions.

1.2 Need of the study

The study is needed to address the growing demand for accurate pricing models in the sharing economy. Airbnb's rapid growth has created a need for hosts and renters to have a better understanding of rental prices. By developing a machine learning model that accurately predicts rental prices based on various features, this study can help improve the overall user experience on the platform and provide valuable insights into Airbnb's pricing strategies. Additionally, the study highlights the potential for data-driven approaches to enhance the sharing economy and inform future research in this area.

1.3 Data Sources

- Inside Airbnb

1.4 Tools & Techniques

Tools:

- Python
- NumPy
- Pandas
- matplotlib
- Seaborn
- Plotly
- Folium
- SciPy
- Statsmodels
- LightGBM
- CatBoost
- XGBoost

Techniques :

To evaluate the performance of nine regression models we use mean squared error, root mean squared error, mean absolute error and r2 score as evaluation metrics.

CHAPTER 2 : DATA PREPERATION AND UNDERSTANDING

One of the first steps we engaged in was to outline the sequence of steps that we will be following for our project. Each of these steps are elaborated below:

2.1 Phase I – Data Extraction and Cleaning

- Reading the dataset using Pandas
- Identifying and handling missing values, outliers and duplicates
- Checking for data inconsistencies and correcting them
- Converting data types as necessary
- Dropping irrelevant or redundant columns

2.2 Phase II – Exploratory Data Analysis

- Performing univariate, bivariate and multivariate analysis to understand the data
- Creating visualizations to summarize and present the data
- Calculating summary statistics such as mean, median and standard deviation to describe the data

2.3 Phase III – Feature Engineering

- Selecting relevant features for the model
- Transforming and scaling features to improve model performance
- Encoding categorical variables using techniques such as one- hot encoding or label encoding

2.1 Phase I – Data Extraction and Cleaning

	id	host_id	latitude	longitude	price	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count	availability_365
count	4.889500e+04	4.889500e+04	48895.000000	48895.000000	48895.000000	48895.000000	48895.000000	38843.000000	48895.000000	48895.000000
mean	1.901714e+07	6.762001e+07	40.728949	-73.952170	152.720687	7.029962	23.274466	1.373221	7.143982	112.781327
std	1.098311e+07	7.861097e+07	0.054530	0.046157	240.154170	20.510550	44.550582	1.680442	32.952519	131.622289
min	2.539000e+03	2.438000e+03	40.499790	-74.244420	0.000000	1.000000	0.000000	0.010000	1.000000	0.000000
25%	9.471945e+06	7.822033e+06	40.690100	-73.983070	69.000000	1.000000	1.000000	0.190000	1.000000	0.000000
50%	1.967728e+07	3.079382e+07	40.723070	-73.955680	106.000000	3.000000	5.000000	0.720000	1.000000	45.000000
75%	2.915218e+07	1.074344e+08	40.763115	-73.936275	175.000000	5.000000	24.000000	2.020000	2.000000	227.000000
max	3.648724e+07	2.743213e+08	40.913060	-73.712990	10000.000000	1250.000000	629.000000	58.500000	327.000000	365.000000

The shape of the dataset is : (48895, 16)

The descriptive statistics show that there are outliers in most of the numerical columns. After checking for missing values, it was found that there were some in the columns named "last review" and "reviews per month.

The "reviews per month" is a numerical column with a minimum value of 0.01 and a maximum value of 58.5, with the 75th percentile as 2.02. Due to the high level of variability in the column and the presence of outliers, using simple imputation techniques like mean, median, or mode is not viable. Since the datapoints are missing completely at random, KNN Imputation techniques are used to impute values to this column.

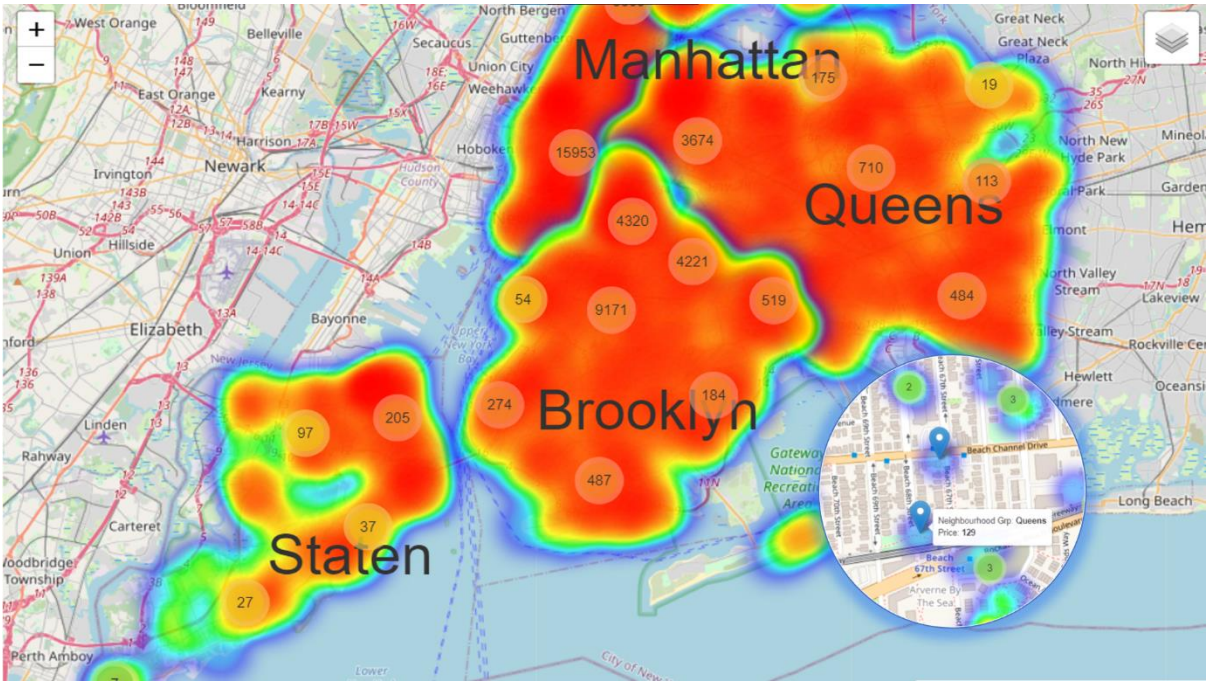
The "last review" is a date column indicating the last date on which the review was posted for the particular listing. Since it is a date column and has no significant value to the price [target], it is better to split the values into month and year separately. The year column contains only data from 2011 to 2019, and there are 12 months in a year. As was done for the above column, in this case also the KNN imputation technique was used to fill in missing values for both columns since in this case also the data points were completely missing at random.

The first four columns (id, name, host name, and host id) in the dataset are excluded from the model-building process as they are considered to provide descriptive or identifying information rather than useful attributes for predicting listing prices.

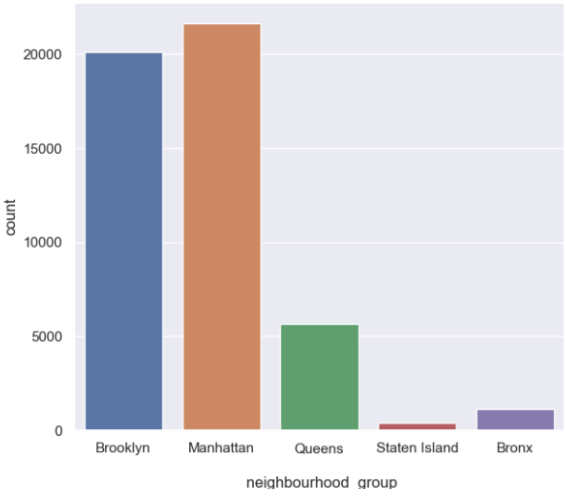
The treatment for outliers is done in the coming sections, where box-cox and power transformations are used to convert the columns into a normal distribution.

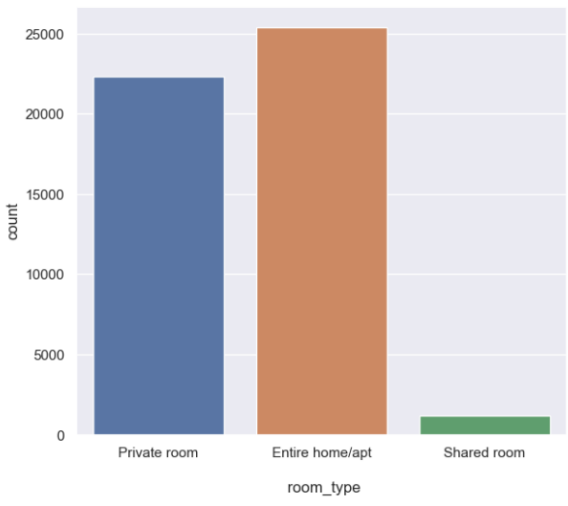
2.2 Phase II – Exploratory Data Analysis

The longitude and latitude columns, along with the neighbourhood group containing 5 areas in NYC (Manhattan, Queens, Brooklyn, Staten Island, and Bronx) and price, are used to create a map using the package folium to understand the areas in NYC where the Airbnb listings are located. If we zoom into these values and hover over any points, we can see the respective neighbourhood group along with its price.

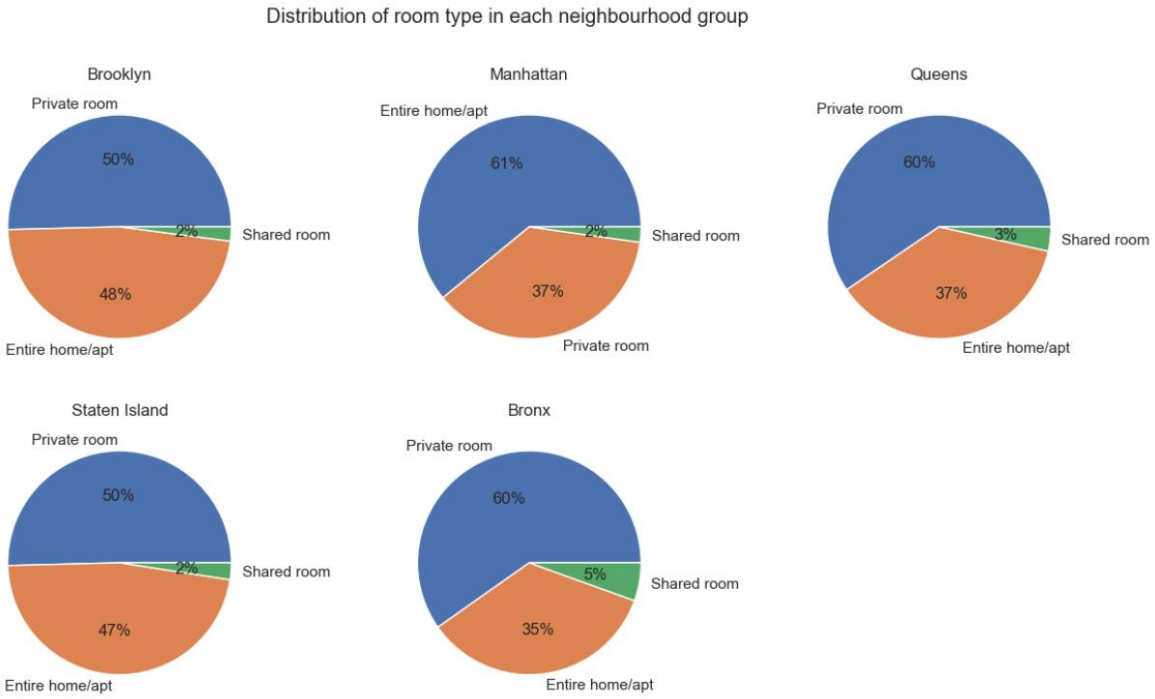


The two main categorical columns that can be seen in the data are neighbourhood group and room type. The frequency plots of these two attributes are as follows :



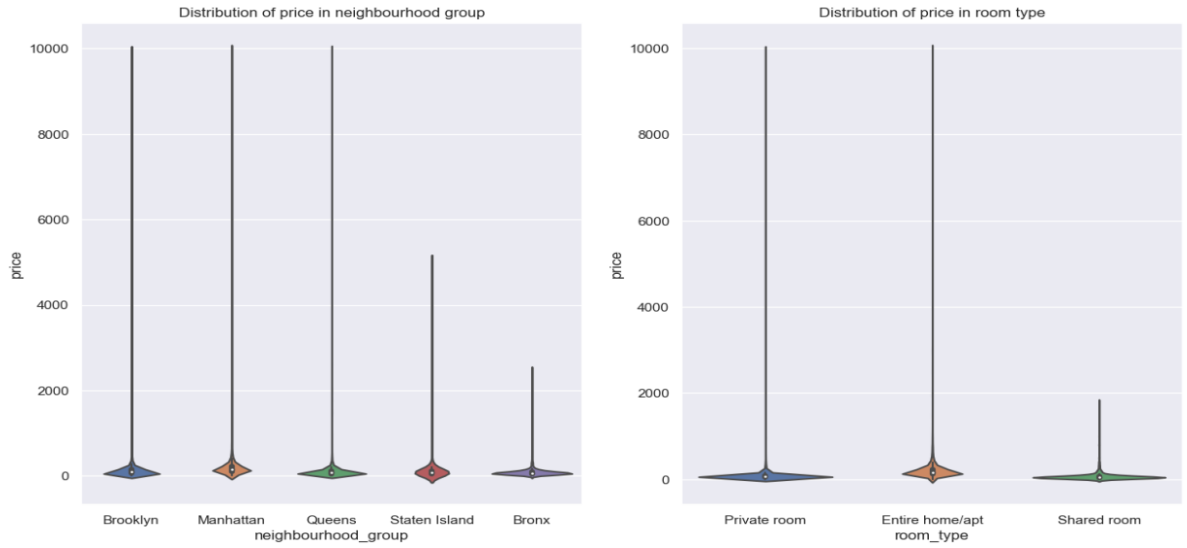


The percentage of room type containing 3 unique values [Private room, Entire home/apt and Shared room] for each neighbourhood can be demonstrated using a pie chart.



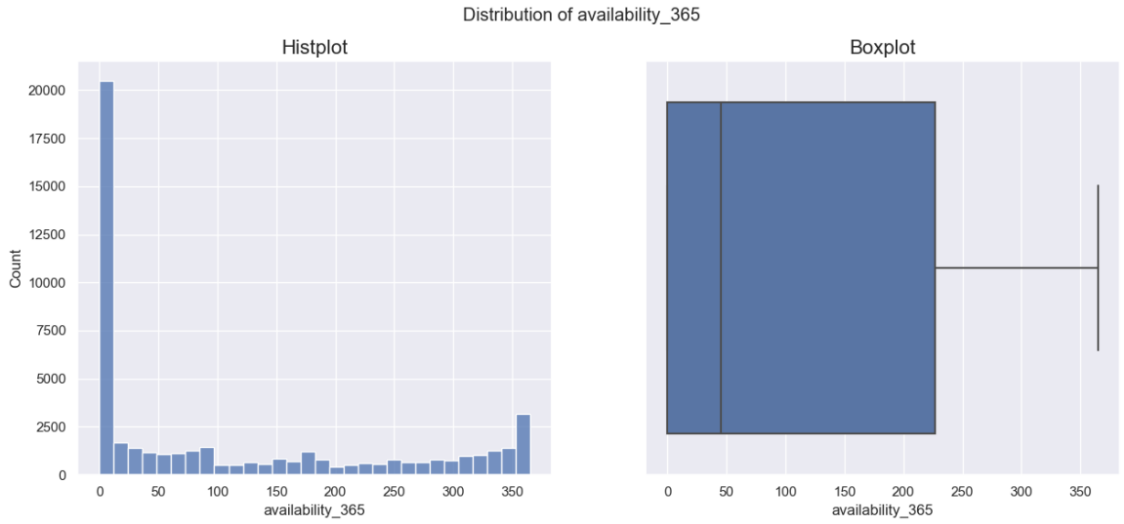
The pie chart shows that almost all the neighbourhood groups contain above 50% of private rooms, except Manhattan, where the proportion of homes and apartments is higher.

Next comes the distribution of the target variable, which is the rental price, in each neighbourhood group and for each room type.



These two violin plots clearly indicates the presence of outliers in the price column.

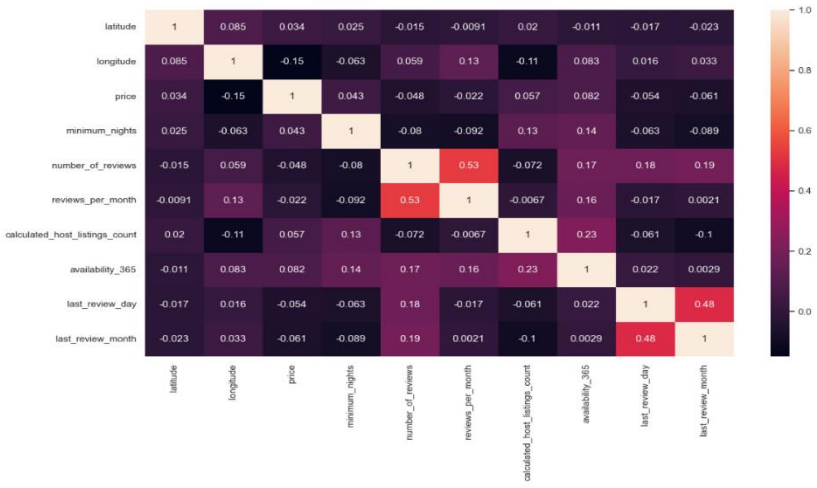
For numerical columns boxplots and histograms can be used to understand the distribution of each attribute. An example of these two plots for the attribute availability 365 is shown below:



The final observations from ploting these two graphs for all the numerical columns are as follows:

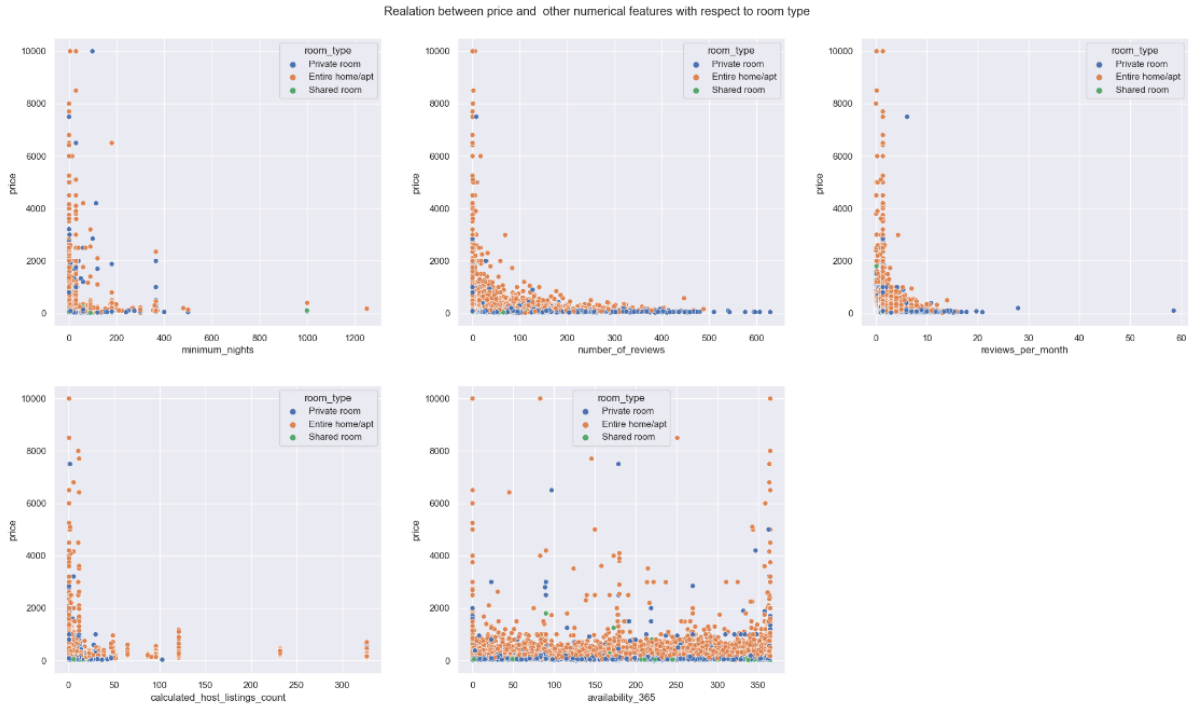
- All the variables except availability_365 is extremely right skewed.
- Except availability_365 all other numerical features have outliers.

The presence of multicolinearity can lead to unreliable, unstable, and inaccurate regression models, which can hinder our ability to make accurate predictions and draw meaningful conclusions from our data. A simple heatmap for the pairwise correlation of each attribute can be used to understand the relationship between them.



All the values in the heatmap are either less than 0.2 or greater than -0.2, which is almost close to zero. This shows that all the attributes are weakly correlated with all other attributes.

A scatter plot can be used to analyse the relationship between two numerical columns. Taking price in the y-axis and rest of each numerical column in the x-axis along with different colour for different categories in room type scatter plots can be drawn as follows:



The problem that arises here is that there is no linear relationship that can be found for any of the numerical columns with respect to the target variable. This problem can be solved by using different transformation methods on the target variable.

2.3 Phase III – Feature Engineering

The number of outliers in the target variable is checked using a box plot and is 2977 of 48895 records. So, it is better to remove these records from the original data for better predictions.

Since all the numerical columns are extremely right skewed with a lot of outliers, box-cox and boxcox1p using power transformer are used to transform the data points into normal curves.

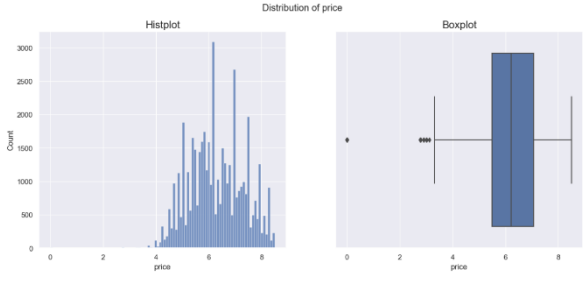
The box-cox transformation is used for attributes that are strictly positive; that is, zeros also cannot be included. The attributes “minimum nights” and “calculated host listings count” are transformed using simple box-cox method.

In situations where the data points contain zero or negative values, boxcox1p along with a power transformer can be used to convert the data into normal curves. First, the power transformer is fitted into the data points to find out the lambda values, which are then used in boxcox1p to transform the respective columns into normal curves. All other numerical columns except the two mentioned above contain zeros and are thus transformed using this method.

A small example of transformation for the column price, before and after is given below:

BEFORE

AFTER



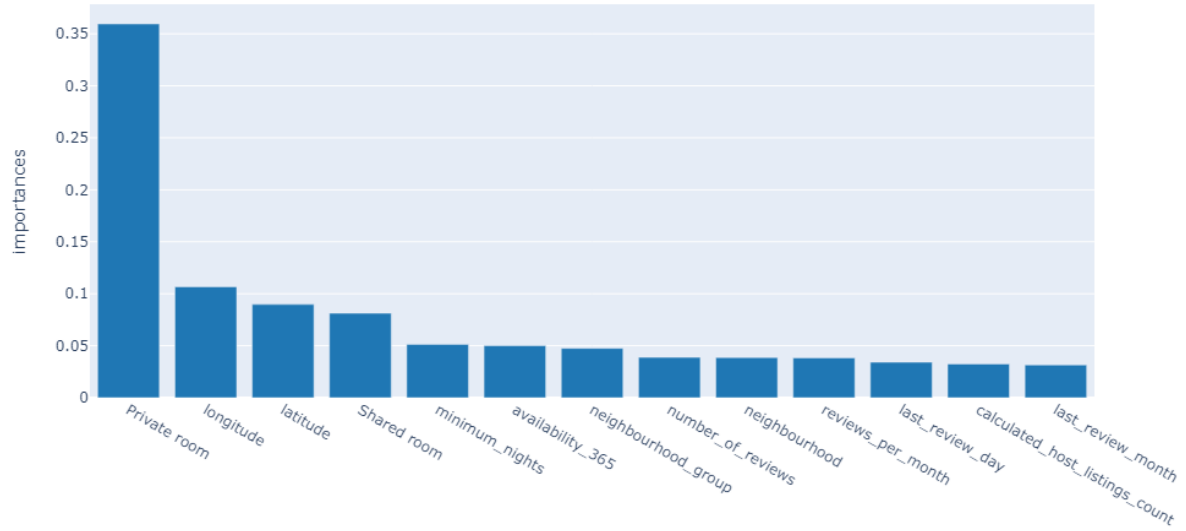
When dealing with categorical columns in nominal scale, such as "neighbourhood" and "neighbourhood group", the method of label encoding was applied. On the other hand, for columns in ordinal scale, such as "room type", one-hot encoding was implemented.

CHAPTER 3 : FITTING MODELS TO DATA

3.1 Train-Test_Split

The train-test split method was used to evaluate the performance of machine learning models. This method involves splitting the available dataset into two parts: a training set and a testing set. The training set, which accounted for 70% of the data, was used to train the machine learning models, while the remaining 30% was used for testing the models. The training set had 32,142 records, and the testing set had 13,776 records. The train-test split allowed for the evaluation of the machine learning models on new, unseen data, which is essential for determining their effectiveness and generalizability.

For a quick head start, an ExtraTreesRegressor was built on the data to understand the features that are important for model building. The result of this model when plotted onto a bar graph was as follows:



The feature private room has a higher contribution for predicting price followed by longitude, latitude, etc...

For each of the models given below, a GridSearchCV or RandomizedSearchCV was used to find the best parameters suitable for the models.

3.2 Linear Regression

A simple linear model that attempts to predict the relationship between a dependent variable and one or more independent variables through a linear equation.

Best Parameters: {'fit_intercept': True}

MAE : 0.54; MSE : 0.50; RMSE : 0.71; R2 : 0.52

3.3 Decision Tree

A tree-structured model that breaks down a dataset into smaller and smaller subsets based on a set of decisions or rules until the subsets contain instances with a single class or value.

Best Parameters: {'max_depth': 5, 'min_samples_leaf': 2, 'min_samples_split': 2}

MAE : 0.52; MSE : 0.45; RMSE : 0.67; R2 : 0.56

3.4 Random Forest

An ensemble model that combines multiple decision trees to improve prediction accuracy and reduce overfitting.

Best Parameters: {'n_estimators': 130, 'min_samples_split': 9, 'min_samples_leaf': 6, 'max_features': 10, 'max_depth': 10, 'bootstrap': True}

MAE : 0.47; MSE : 0.39; RMSE : 0.63; R2 : 0.62

3.5 KNN

A non-parametric model that predicts the value of a data point based on the values of its nearest neighbors in the training data.

Best Parameters: {'weights': 'distance', 'p': 1, 'n_neighbors': 13, 'leaf_size': 44, 'algorithm': 'brute'}

MAE : 0.60; MSE : 0.60; RMSE : 0.77; R2 : 0.43

3.6 Ada Boost

A boosting algorithm that combines multiple weak learners into a strong learner through weighted voting to improve prediction accuracy.

Best Parameters: {'n_estimators': 400, 'learning_rate': 0.013848863713938732, ' base_estimator': DecisionTreeRegressor(max_depth=2)}

MAE : 0.58; MSE : 0.55; RMSE : 0.74; R2 : 0.47

3.7 Gradient Boost

A boosting algorithm that combines multiple weak learners to make a strong learner through an additive model, where each new learner corrects the errors of the previous one.

Best Parameters: {'subsample': 0.8999999999999999, 'n_estimators': 600, 'min_ samples_split': 6, 'max_depth': 7, 'learning_rate': 0.018307382802953697}

MAE : 0.46; MSE : 0.38; RMSE : 0.62; R2 : 0.63

3.8 Light GBM

A gradient boosting framework that uses a tree-based learning algorithm and aims to improve efficiency, accuracy, and speed by using a novel technique called Gradient-based One-Side Sampling (GOSS).

Best Parameters: {'num_leaves': 38, 'n_estimators': 170, 'min_data_in_leaf': 23, 'm ax_depth': 10, 'learning_rate': 0.13219411484660287, 'feature_fraction': 0.8, 'cols ample_bytree': 0.5}

MAE : 0.46; MSE : 0.38; RMSE : 0.62; R2 : 0.63

3.9 Cat Boost

A gradient boosting framework that uses categorical features as input and applies a novel algorithm called Ordered Boosting to reduce overfitting and improve prediction accuracy.

Best Parameters: {'subsample': 0.8999999999999999, 'n_estimators': 600, 'max_ depth': 9, 'learning_rate': 0.061359072734131756, 'l2_leaf_reg': 54.6227721768 4348, 'colsample_bylevel': 0.7999999999999999}

MAE : 0.46; MSE : 0.38; RMSE : 0.62; R2 : 0.63

3.10 XGBoost

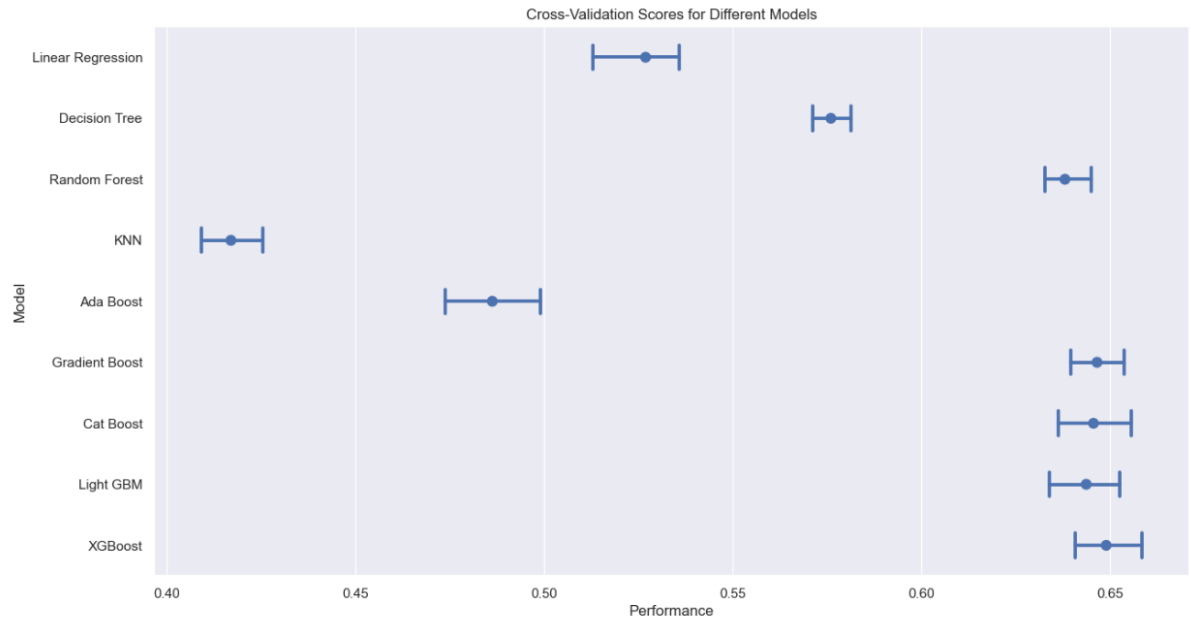
A gradient boosting framework that uses a tree-based learning algorithm and applies several techniques to improve prediction accuracy, such as regularization, parallel processing, and sparsity awareness.

Best Parameters: {'subsample': 0.7999999999999999, 'reg_alpha': 0.016681005372000592, 'n_estimators': 500, 'max_depth': 9, 'learning_rate': 0.018307382802953697, 'gamma': 0.1291549665014884, 'colsample_bytree': 0.6}

MAE : 0.46; MSE : 0.37; RMSE : 0.61; R2 : 0.64

CHAPTER 4 : KEY FINDINGS

XGBoost performed the best among all the models tested, with an R-squared score of 0.64, indicating that 64% of the variance in the target variable can be explained by this model.



The point plot shows that XGBoost had the highest performance, followed by Cat Boost, Light GBM, Gradient Boost and Random Forest while KNN had the lowest performance.

Linear regression, Decision Tree and AdaBoost, performed somewhere in between XGBoost and KNN.

	MAE	MSE	RMSE	R2
Linear Regression	0.549566	0.507667	0.712507	0.517544
Decision Tree	0.517078	0.455647	0.675016	0.566980
Random Forest	0.473580	0.393368	0.627190	0.626167
KNN	0.598584	0.600683	0.775037	0.429146
Ada Boost	0.582384	0.550523	0.741972	0.476816
Gradient Boost	0.465733	0.381991	0.618054	0.636978
Cat Boost	0.466456	0.381832	0.617926	0.637129
Light Gradient Boost	0.465710	0.381490	0.617649	0.637454
Xtreme Gradient Boost	0.463795	0.378487	0.615213	0.640308

The table shows that XGBoost had the lowest mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE), and the highest R-squared score among all the models tested.

CHAPTER 5 : RECOMMENDATIONS AND CONCLUSION

RECOMMENDATIONS:

- Based on the analysis, XGBoost is recommended as the best model for the given dataset and target variable. Further optimization and tuning of XGBoost could potentially improve its performance.

- Feature engineering and selection could be explored to potentially improve the performance of the models.

- Cross-validation techniques such as k-fold or stratified k-fold can be used to validate the model's performance on different subsets of the data and avoid overfitting.

CONCLUSION:

- The results of the analysis indicate that XGBoost is the most suitable model for the given dataset and target variable.

- The study demonstrates the importance of exploring multiple models and evaluating their performance to select the best one for the given problem.

- The findings can be used to make data-driven decisions and improve the performance of the model for similar problems in the future.