SPRINGBOARD - DSC

CAPSTONE PROJECT # 1

NEW YORK AIRBNB PRICING METRICS USING MACHINE LEARNING

Introduction

Airbnb is a marketing company that acts as a broker to provide short-term homestay experiences that offer several advantages like exposure to everyday life in another location, and the opportunity to experience local culture and traditions. Multiple reports forecast an increase in future Airbnb market capital and revenue with a growing economy.

Goal

Our client is a successful investment and venture capital firm based in New York, owns several successful businesses, and now wants to invest in new york based Airbnbs but wants to understand in-depth knowledge and information about the market before strategic decision-making.

Data Specification

This project will use open-source data from Kaggle:

https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data

The dataset of 48,896 rows and 16 columns contains information about location, room type, prices, reviews, availability, and host information.

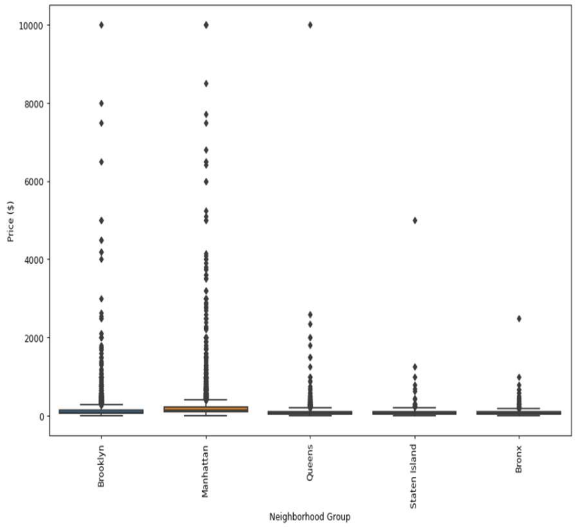
The business scenario will be modeled by building several regression models, which will be analyzed, evaluated, and compared according to appropriate performance metrics selected according to the goals of the client. Similarly, we will look more in detail at different prices based on unit type, followed by data wrangling and estimated data analysis. Lastly, we will build the model. In addition, interpretability analyses will be conducted to characterize how the variation of identified features will affect the probability associated with it.

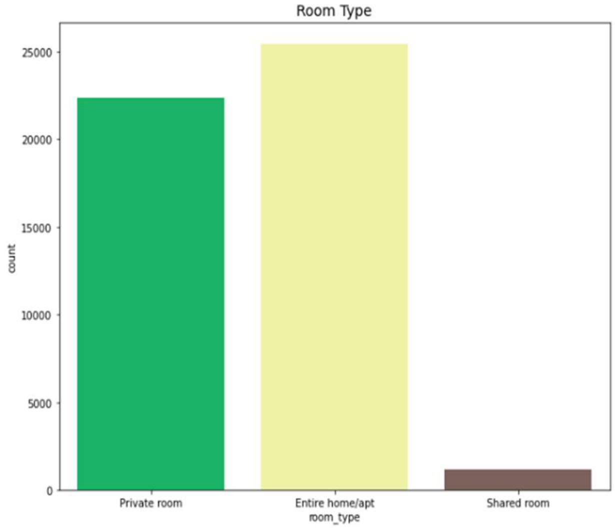
Raw Data

```
id                              int64
name                            object
host_id                         int64
host_name                       object
neighbourhood_group             object
neighbourhood                   object
latitude                        float64
longitude                       float64
room_type                       object
price                           int64
minimum_nights                  int64
number_of_reviews               int64
last_review                     object
reviews_per_month               float64
calculated_host_listings_count  int64
availability_365                int64
```

Data Wrangling

The Kaggle data was clean and organized in rows and columns and not messy at all which speed up the data preparation process.





Here is the overview of the main issue I came across while cleaning the data

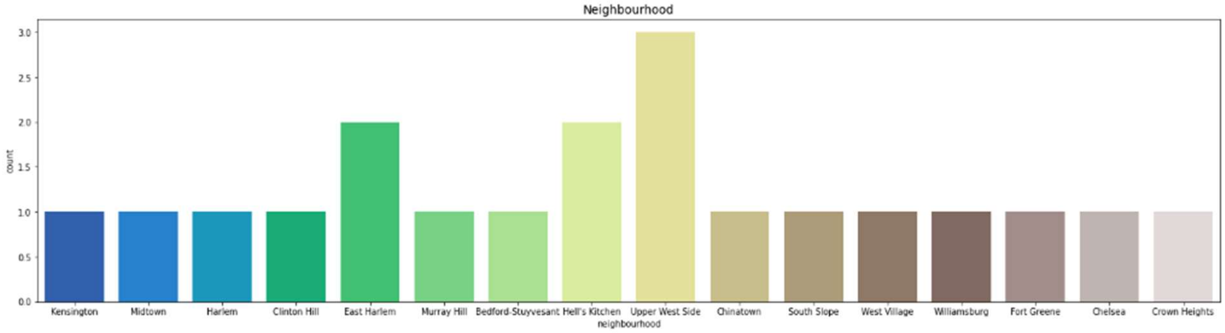
Problem: Lot of null values in multiple columns. Some of the columns have 20% null values.

Solution: Sort and count the data for better understanding. Remove the null values and replace them with zeros, save them in a new file, and prepare it for more details exploratory data analysis.

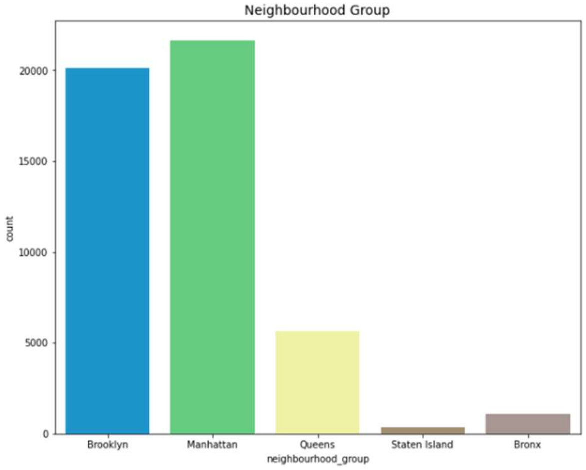
I also had to clean the table by deleting a lot of extraneous information which would not serve this project like *id, name, host_id, and host_name*.

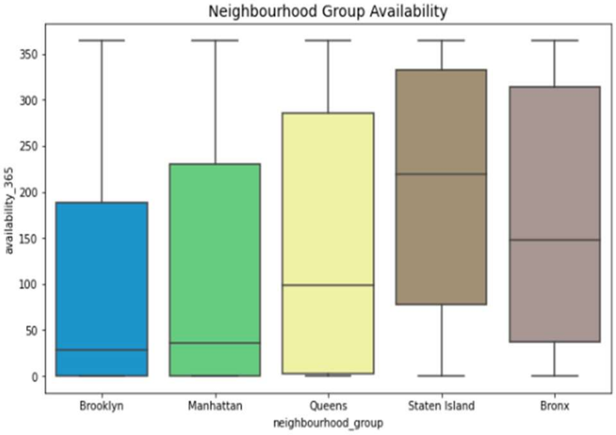
Exploratory Data Analysis - EDA

The findings of the exploratory data analysis identified that the data we use for this project is sufficient. Here is an overview of EDA findings,

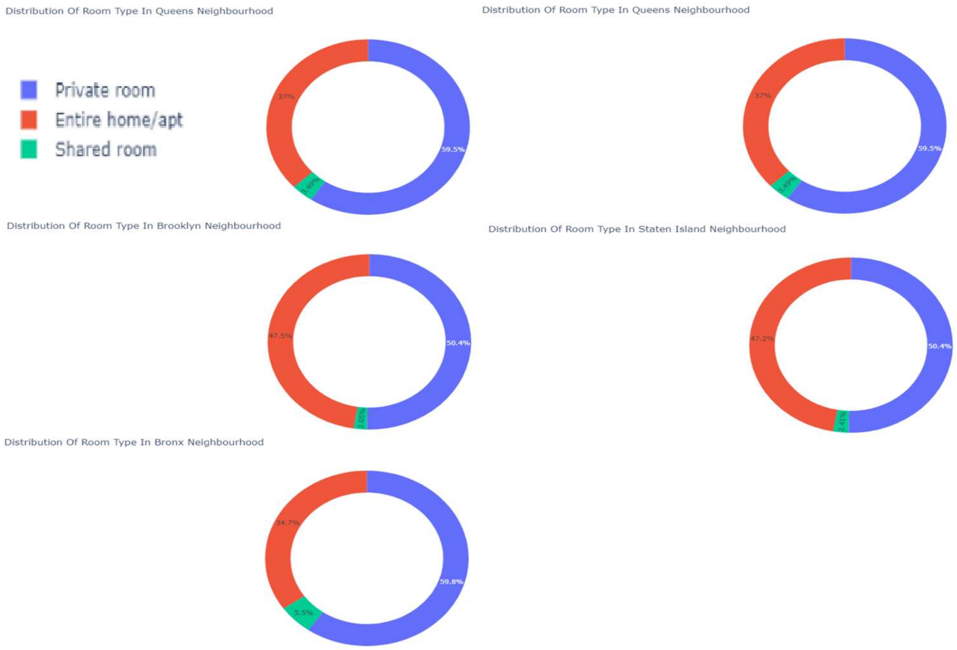


Seaborn Barplot of the Neighbourhood shows the top 20 locations in the New York area. The upper west side has the highest number of Airbnb units followed by Hell's Kitchen and East Harlem.

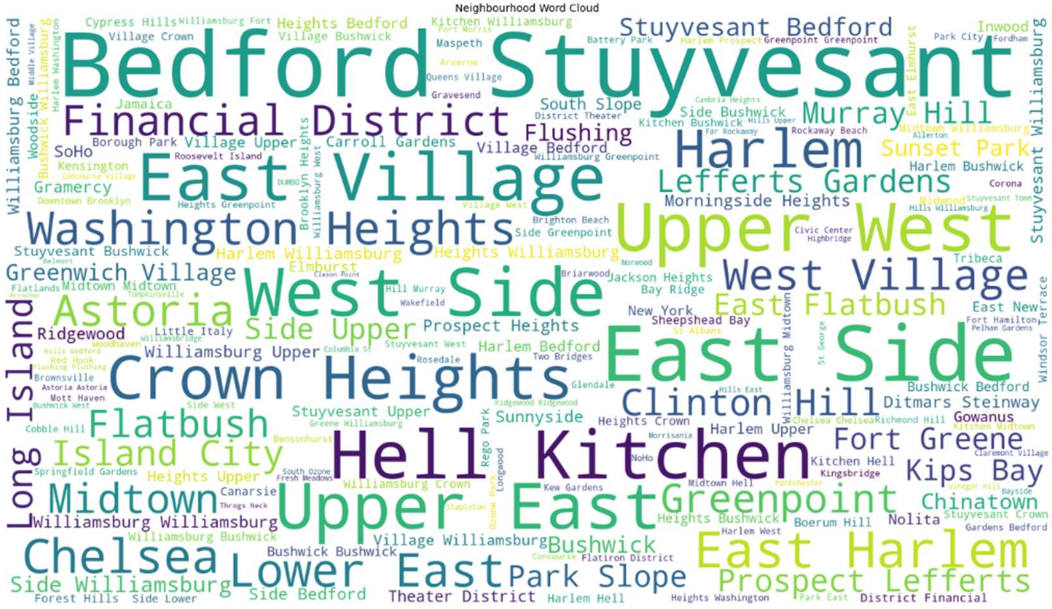




The left plot shows the comparison among different neighborhood groups in the New York area Manhattan has the most listed Airbnb units, and Brooklyn comes close with the total listed units just above 20k. Staten Island is among the top neighbourhood group. We can also interpret this as the most favorite tourist destination with the highest percentile of rooms available throughout the year.



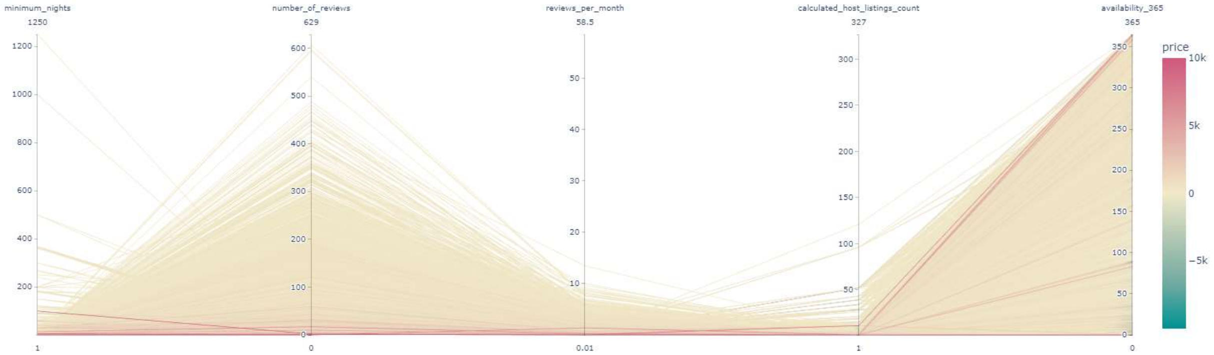
Here, private rooms have the highest percentage in every neighborhood group with over 50% each. Shared rooms have the lowest.



Word cloud is used to visualize text data and the font size of each show the frequency, importance, and prominence. Hell Kitchen, East side, west side, Bedford, and Stuyvesant are the most frequent areas.

Continous Variable Relationship with the Price (target variable)

We also compare *price* (target variable) with the main features *minimum_nights, number_of_reviews reviews_per_month, calculated_host_listings_count,* and *availability_365* in our data by visualizing it through a parallel coordinate plot. Each feature corresponds to a vertical axis value/element and is displayed as a series of connected points.

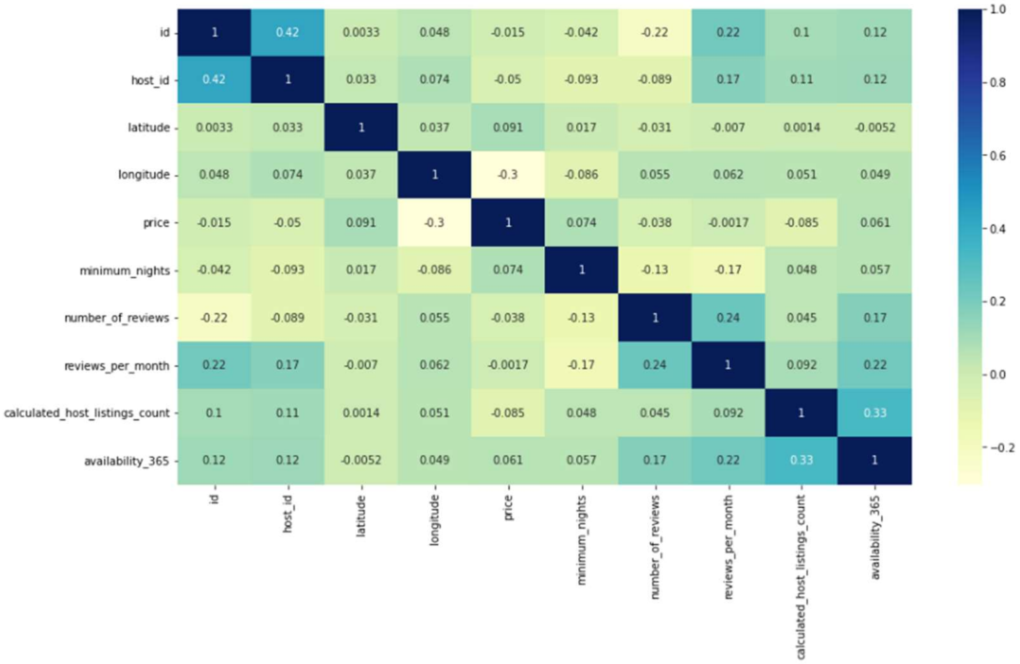


A parallel coordinate plot shows a nice overview of comparison among the different features. Too much overlapping can be done by scaling the axis because the plot is based on interpolation (linear combination).

When most lines between two parallel axes are somewhat parallel, it suggests a positive relationship between these two features. When lines cross like X-shapes, it's a negative relationship. When lines cross randomly or are parallel, it shows there is no particular relationship.

Correlation Matrix

The correlation matrix heatmap plot shows the 2D correlation matrix among different features with a color palette showing the variation. We use the Kendell method here to find the similarities and measures of correspondence.

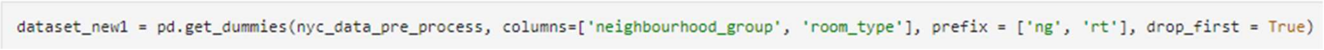


Here, 1 indicates strong agreement, and -1 indicates strong disagreement. We can see here that price is strongly correlated with the number of accommodations, location, number of nights, and reviews. We also see that these features are correlated. These results are quite obvious because the price depends on the location of the place.

Pre-processing and Training Data Development

Dummy Variables

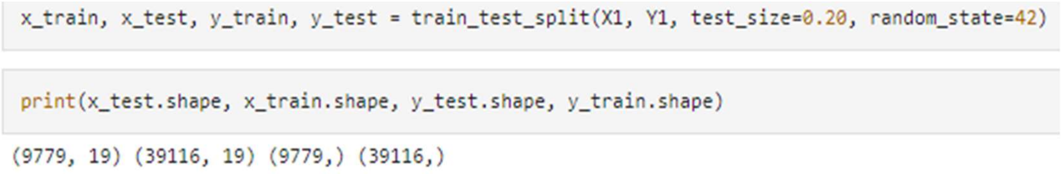
A dummy variable takes the values 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome. Here, we are studying the relationship between neighbourhood_group and room_type (two categorical variables).



```
dataset_new1 = pd.get_dummies(nyc_data_pre_process, columns=['neighbourhood_group', 'room_type'], prefix = ['ng', 'rt'], drop_first = True)
```

Splitting into Test and Train

Training and testing data are two sets of data used in the development of a machine learning model. The training data is used to create the model, while the testing data is used to evaluate the model's performance and ensure it produces accurate and reliable results.



```
x_train, x_test, y_train, y_test = train_test_split(X1, Y1, test_size=0.20, random_state=42)
```

```
print(x_test.shape, x_train.shape, y_test.shape, y_train.shape)
```

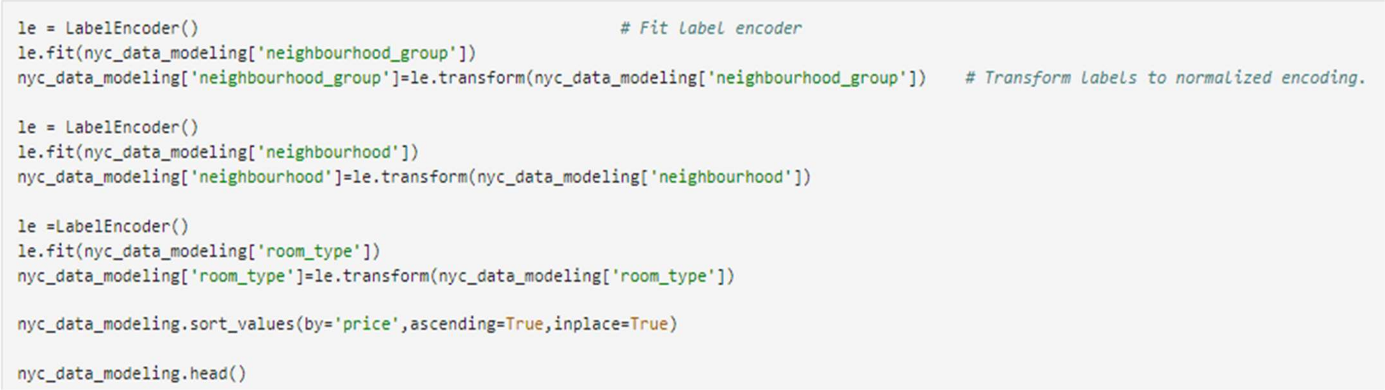
```
(9779, 19) (39116, 19) (9779,) (39116,)
```

Data Modeling

In this section, we will examine various machine learning models, evaluate their performances and recommend the best model for the project.

Feature selection and Normalization Labels

Models will only use listings which has a price set up with *price* > *0* and *availability_365* > *0*. There are multiple listings with no prices. Normalization involves scaling the values of each feature between 0 and 1 while transforming columns refers to replacing a feature's values with a new set of values that are scaled between 0 and 1. This enables the machine learning model to accurately identify the important features in the dataset and ensure that all the features are on the same scale.



```
le = LabelEncoder()                                      # Fit label encoder
le.fit(nyc_data_modeling['neighbourhood_group'])
nyc_data_modeling['neighbourhood_group']=le.transform(nyc_data_modeling['neighbourhood_group'])    # Transform labels to normalized encoding.

le = LabelEncoder()
le.fit(nyc_data_modeling['neighbourhood'])
nyc_data_modeling['neighbourhood']=le.transform(nyc_data_modeling['neighbourhood'])

le =LabelEncoder()
le.fit(nyc_data_modeling['room_type'])
nyc_data_modeling['room_type']=le.transform(nyc_data_modeling['room_type'])

nyc_data_modeling.sort_values(by='price',ascending=True,inplace=True)

nyc_data_modeling.head()
```

I also use the log base 10 functions here for normalization which helps to reduce the outliers and extreme values in the data.

Machine Learning Model

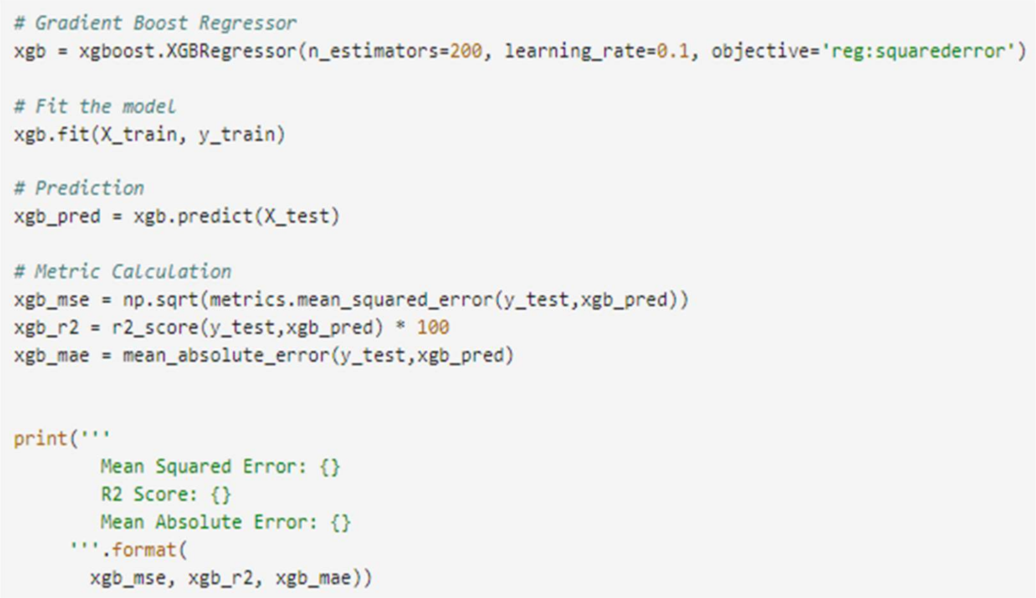
Model Selection

There are a variety of ways to select a model, such as using cross-validation, grid search, or an information criterion. For this project, we choose cross-validation which is a technique used to estimate the accuracy of a model by training it on a subset of the data and testing it on the remaining data.

Let's build various models to evaluate the performance of the data. In this notebook. I will compare six different supervised machine-learning models using the great Scikit-Learn library:

1. Linear Regression
2. Decision Tree
3. Bayesian Regression
4. Ridge Regression (linear model)
5. Lasso Regression (linear model)
6. Gradient Boosting Regression

One of the best parts about scikit-learn is that all models are implemented identically. The implementation of entire training and testing procedures for all those listed models can be done in just a few lines of code.



```
# Gradient Boost Regressor
xgb = xgboost.XGBRegressor(n_estimators=200, learning_rate=0.1, objective='reg:squarederror')

# Fit the model
xgb.fit(X_train, y_train)

# Prediction
xgb_pred = xgb.predict(X_test)

# Metric Calculation
xgb_mse = np.sqrt(metrics.mean_squared_error(y_test,xgb_pred))
xgb_r2 = r2_score(y_test,xgb_pred) * 100
xgb_mae = mean_absolute_error(y_test,xgb_pred)


print('''
        Mean Squared Error: {}
        R2 Score: {}
        Mean Absolute Error: {}
     '''.format(
       xgb_mse, xgb_r2, xgb_mae))
```

Model Overview

- Linear regression models are used to predict a continuous dependent variable, based on one or more independent variables. They are simple to use and understand, and they can provide good predictive accuracy when the data is linear. They are also relatively fast to train, which makes them suitable for large datasets. Additionally, linear regression models are often the first model used when trying to solve a supervised learning problem.
- Decision tree models are a type of supervised machine learning algorithm that is used to predict a target variable based on one or more input features. Decision trees are popular because they are easy to interpret, can handle both numerical and categorical data, and are relatively fast to train. They are also able to capture non-linear relationships between the target and input variables, which can make them more accurate than linear models. Additionally, decision trees can be used to model complex decision-making processes, which makes them well-suited for tasks such as classification and regression.

- Bayesian regression is a type of supervised machine learning algorithm that is used to predict a continuous target variable based on one or more input features. It is a probabilistic method, which means it considers the uncertainty associated with each prediction and can update its predictions as new data is encountered. Bayesian regression is often used when there is a limited amount of data available, as it can use prior information to make more reliable predictions. Additionally, Bayesian regression can be used to incorporate domain knowledge into the model, which can help improve the accuracy of the predictions.
- Ridge regression is a type of regularized linear regression that is used to reduce the variance of a linear regression model by adding a penalty term to the cost function.
- Lasso regression is a type of regularized linear regression that is used to reduce the complexity of a model by adding a penalty term to the cost function. The main difference between lasso and ridge regression is that lasso performs feature selection while ridge regression does not.
- Gradient boosting regression is a type of supervised machine learning algorithm that is used to predict a continuous target variable based on one or more input features. It is a sequential ensemble method, which means it builds a model by sequentially adding weak learners to the ensemble. Gradient boosting is a powerful technique that can produce accurate and robust models, and it is often used when dealing with complex datasets with a large number of features. Additionally, gradient boosting can handle non-linear relationships between the target and input variables, which can make it more accurate than linear models.

Model Metric Comparision

Here I used three metrics to compute the performance of every model, mean square error, r2 score, and mean absolute error. Mean Squared Error (MSE) is a risk function that measures the square of errors.

MSE = (1/n) * Σ(actual – forecast)^2

Where Σ – a symbol that means “sum”, n – sample size, actual = the actual data value, forecast = the predicted data value

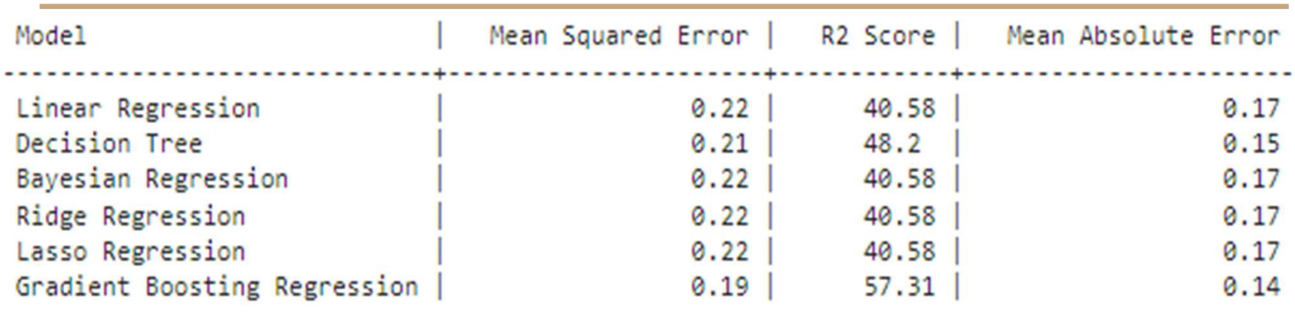
The coefficient of determination (R2) is used to explain how much variability of one factor can be caused by its relationship to another factor, and is sometimes referred to as the "goodness of fit".

R-squared = Explained variation / Total variation

R-squared is always between 0 and 100% (0 & 1). 0% indicates that the model explains none of the variability of the response data around its mean and 100% indicates that the model explains all the variability of the response data around its mean. In general, the higher the R-squared, the better the model fits your data.

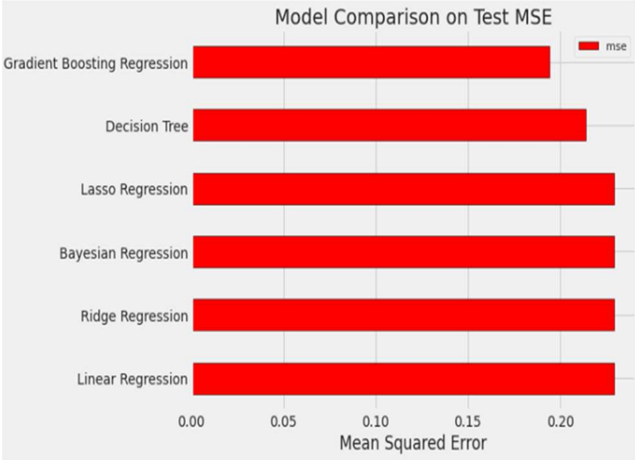
Mean Absolute Error (MAE) is a model evaluation metric that gives the mean of the absolute difference between model prediction and a target value, used with regression models. It is calculated by adding up all the absolute errors and dividing them by the number of errors.

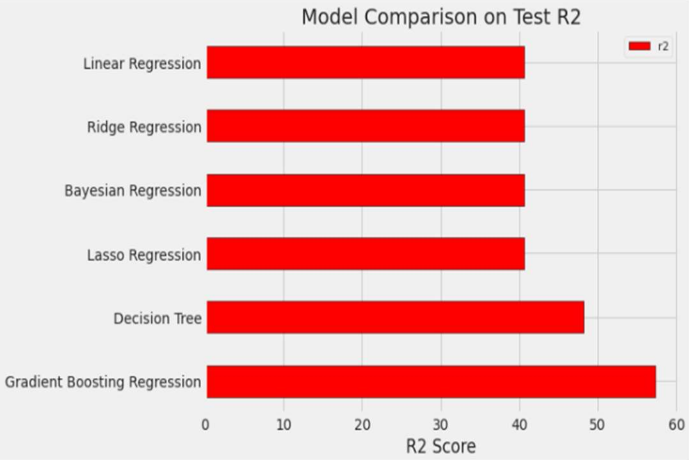
In MAE, different errors are not weighted more or less, but the scores increase linearly with the increase in errors. The difference between an expected value and a predicted value can be positive or negative and will necessarily be positive when calculating the MAE. The closer MAE is to 0, the more accurate the model is. Lower values are better. The table below shows the comparison of all six models in a table format for comparison.

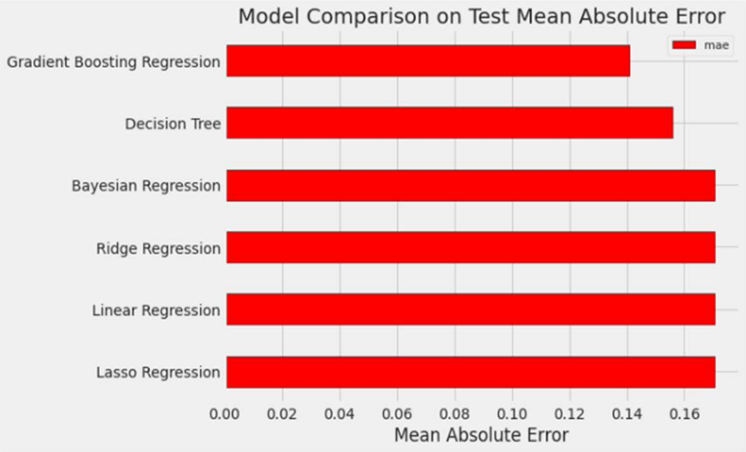


```
Model                        |   Mean Squared Error |   R2 Score |   Mean Absolute Error
-----------------------------+----------------------+------------+----------------------
Linear Regression            |                 0.22 |      40.58 |                  0.17
Decision Tree                |                 0.21 |      48.2  |                  0.15
Bayesian Regression          |                 0.22 |      40.58 |                  0.17
Ridge Regression             |                 0.22 |      40.58 |                  0.17
Lasso Regression             |                 0.22 |      40.58 |                  0.17
Gradient Boosting Regression |                 0.19 |      57.31 |                  0.14
```

The model comparison result shows that all models have a similar prediction. We can see that the Gradient Boosting Regression model has the highest R2 score value and the least mean absolute error (0 indicates no error). Hence it is the best model for our Airbnb project. Plots of the comparison are shown below to better understand the performance of different models.







Summary and Findings

In this project, we performed all the data science steps to create a data science pipeline (DSP) including data wrangling, exploratory data analysis (EDA), pre-processing training, and data modeling.

We also modeled and compared six machine-learning models using different metrics and plots including mean squared error, r2 score, and mean absolute error to compute the performance of every model.

We can see that the Gradient Boosting Regression model has the highest R2 score value and the least error. Hence it is the best model for our Airbnb project. The advanced models, which include bayesian regression, and gradient boosting all show similar metrics. Hence, the XGB Gradient Boosting Regression model has the best performance model.

Recommendations and Future Work

There are a variety of ways to select a model, such as using cross-validation, grid search, or an information criterion. Cross-validation is the only method used in this project. Grid search is a method of tuning hyperparameters by training a model on a grid of different parameter combinations. An information criterion is a measure of how well a model fits the data and can also be used to select the best model.