WINTER 2020, ALY6015

Intermediate Analytics



Week 6 - Final Project

**Presented by**

Avinash Sunil Tripathi

1. **Introduction:**

The dataset that I took into consideration for our final project is AB\_NYC\_2019. It is taken from the official Airbnb site.

This dataset is about the listing activity and metrics in NYC for 2019. Since 2008, guests and hosts have used Airbnb to expand on traveling possibilities and present more unique, personalized way of experiencing the world.

This data file includes all needed information to find out more about hosts, geographical availability, necessary metrics to make predictions and draw conclusions.

*Tabl.1 description of data fields*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Columns** | **Column Description** | | id | listing ID | | name | name of the listing | | host\_id | host ID | | host\_name | name of the host | | neighbourhood\_group | location | | neighbourhood | area | | latitude | latitude coordinates | | longitude | longitude coordinates | | room\_type | listing space type | | price | price in dollars | | minimum\_nights | amount of nights minimum | | number\_of\_reviews | number of reviews | | last\_review | latest review | | reviews\_per\_month | number of reviews per month | | calculated\_host\_listings\_count | amount of listing per host | | availability\_365 | number of days when listing is available for booking | |

In this dataset, we have around 49000 entries and 16 different features. We will initially see if our dataset is having any null values. For this project, we have decided to perform different steps in model building, like data analysis, data visualization and data modelling.

The main objective of this project is to predict the no. of nights for which a particular room will be available if all other information is given. We will try to see how we can implement different models like linear, lasso, decision tree, neural network, gradient boosting and random forest.

The detailed description of each step is given in the analysis part.

1. **Analysis:**

Let us now see how we are going to analyse this dataset to create different model.

We will go through the below mentioned steps:

1. Importing required lib libraries and dataset into python
2. Data cleansing
3. Data visualization
4. Model analysis
5. **Importing required libraries and dataset into python:**

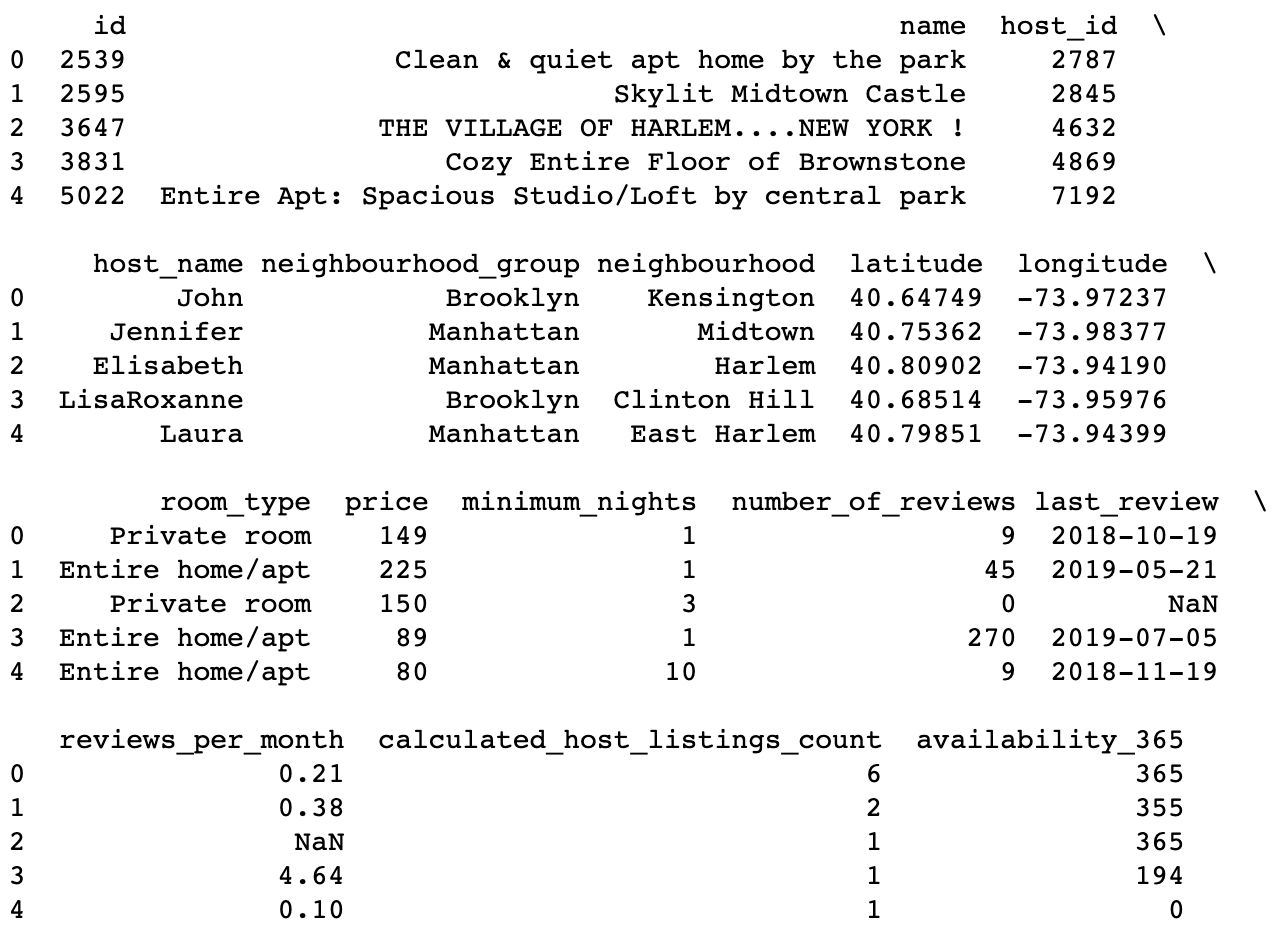
We need to import the following libraries into our system:

* NUMPY
* PANDAS
* MATPLOTLIB.PYPLOT
* SEABORN
* SKLEARN

We will store the excel file into a data frame object. Let’s call it as ds. We can validate the data is right or not by printing first few rows of the dataset.



**Output:**

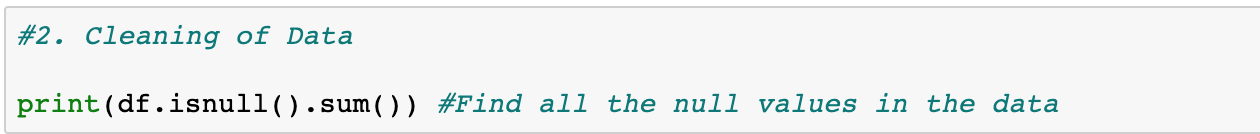


Once we get the data into our system we can move forward and perform the required analysis.

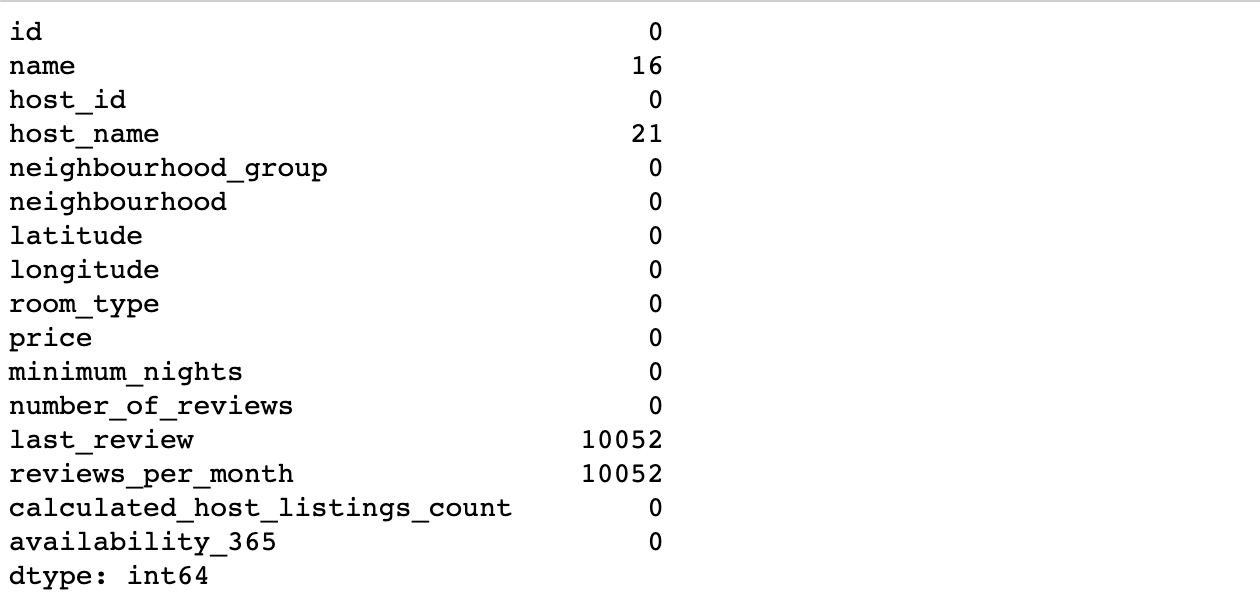
1. **Data cleansing:**

In this part, we will clean our dataset so that we can improve our data quality and in doing so increase the performance.

Let’s first seethe total number of NULL values that we have in our dataset.



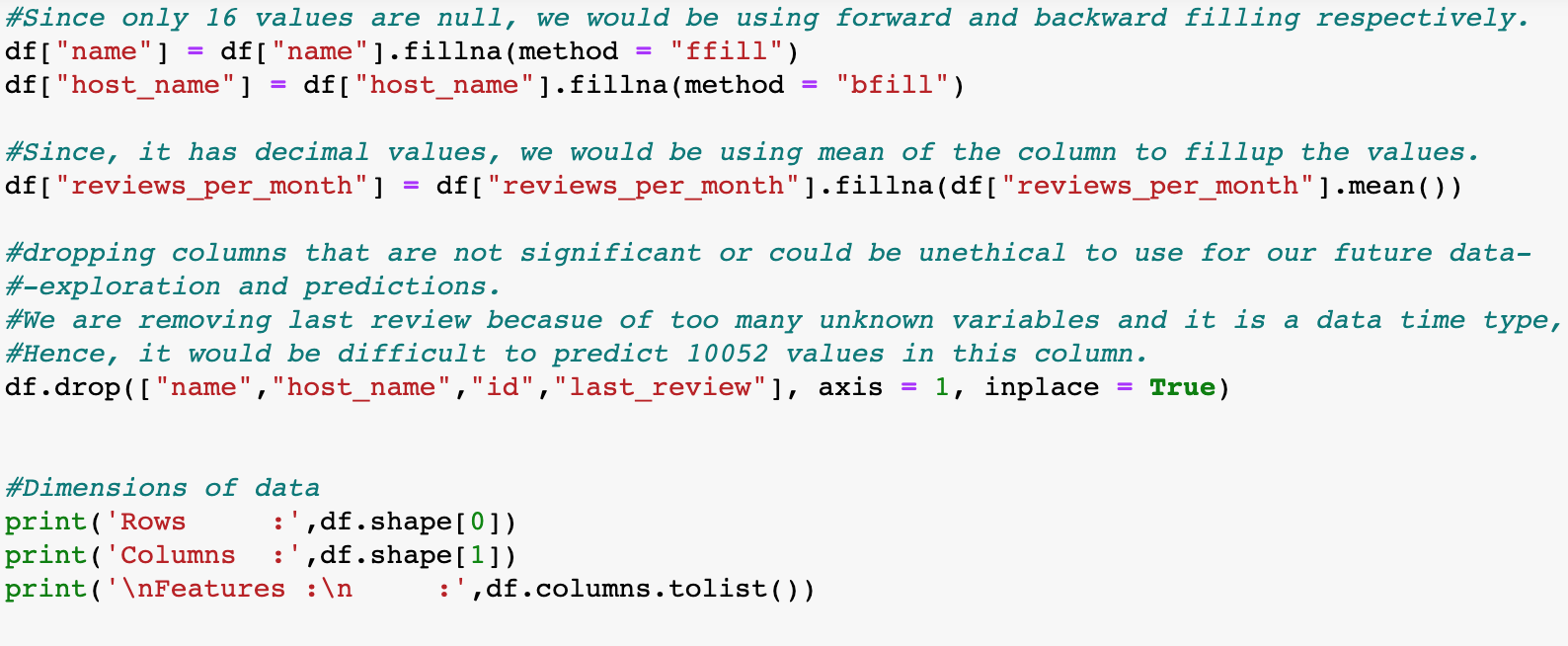
**Output:**



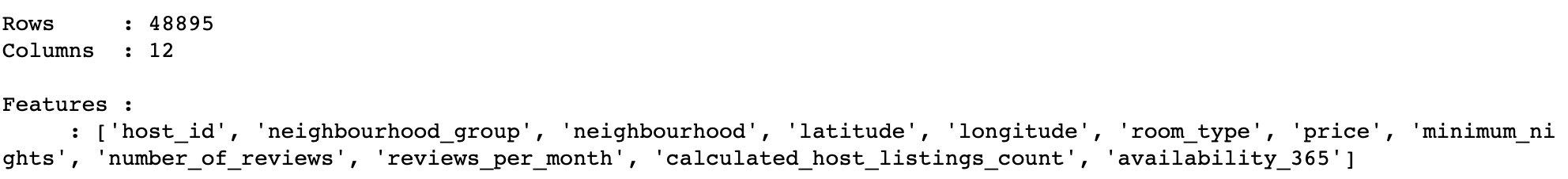
As only 16 values from name and 21 values from host name are there, we will use forward and backward filling on the dataset to fill those.

In case of column reviews per month, we will use mean of the column to fill up the values.

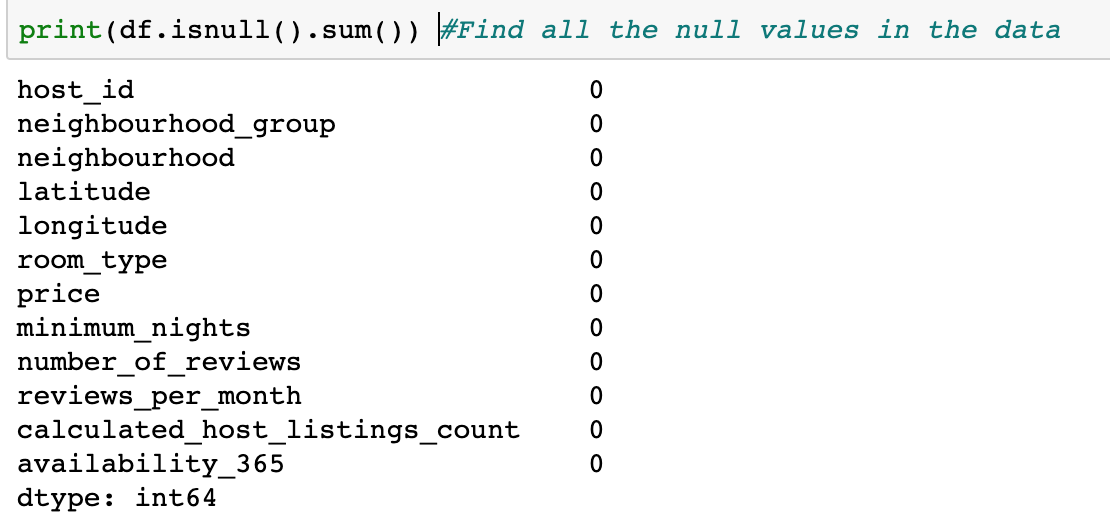
We are removing, last review as there are too many unknowns and it is a date time type. Hence it would be difficult to predict all those 10052 values.



**Output:**



After cleaning our dataset, we can run the ISNULL () method to confirm that there is no NULL value that’s present. And there is no feature with name LAST\_REVIEW as we have removed it.

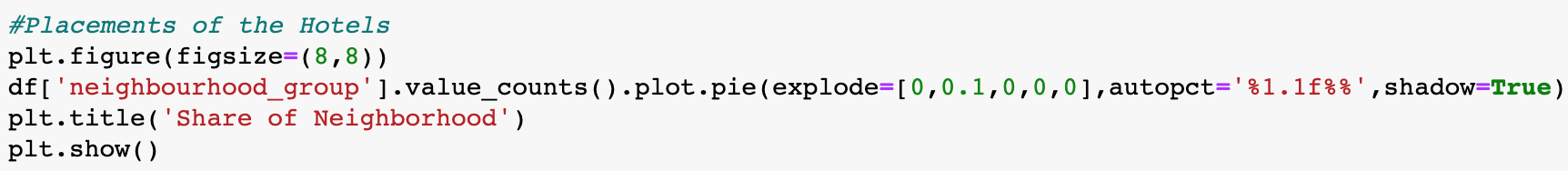


1. **Data visualization:**

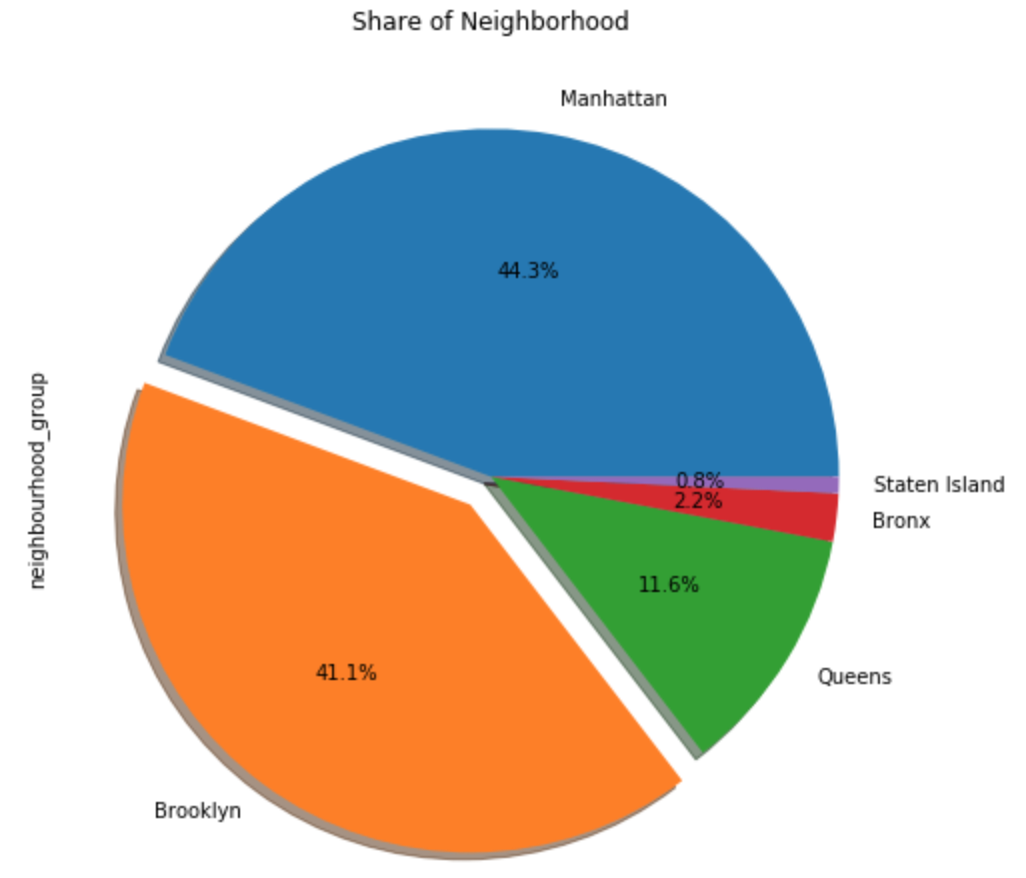
After cleaning our dataset, we will now move onto the data visualization part:

* Placements of the Hotels
* Variations of prices of hotels with area
* Boxplot showing price variation
* Type of Rooms
* Rooms Vs Neighbourhood Group
* Distribution of Room Availability and minimum number of nights

1. **Let’s observe the placement of hotels as per the neighborhood group.**



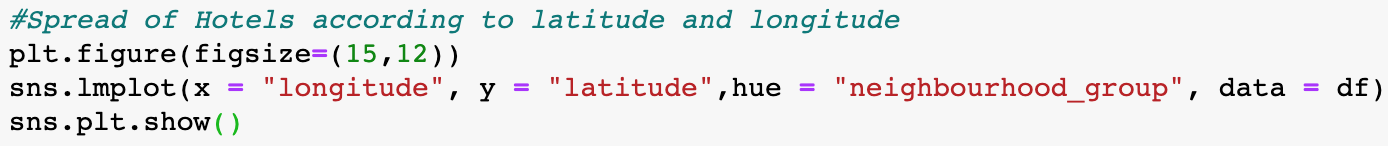
**Output:**



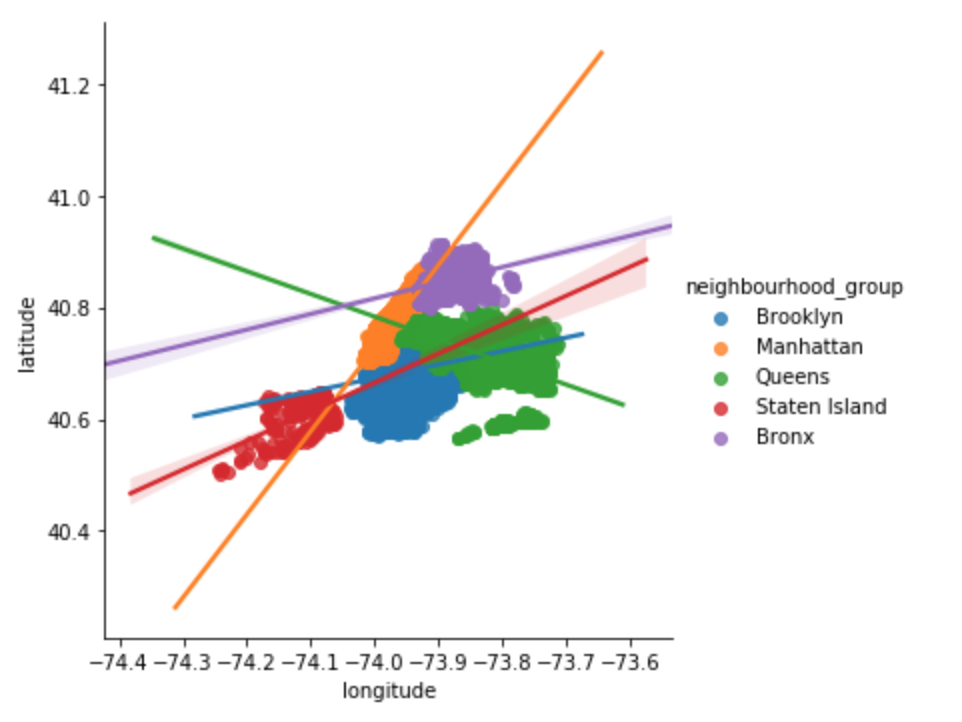
New York city has % county level administrative boroughs namely Manhattan, Brooklyn, queens, Bronx and Staten island. From the above plot, we can conclude that the top boroughs of New York are Manhattan, Brooklyn and Queens. while the least belongs to Bronx and Staten island.

1. **Now let’s observe the spread of hotel according to longitude and latitude:**

We are using simple linear regression to observe the latitude vs longitude graph as per the neighborhood group.



**Output:**



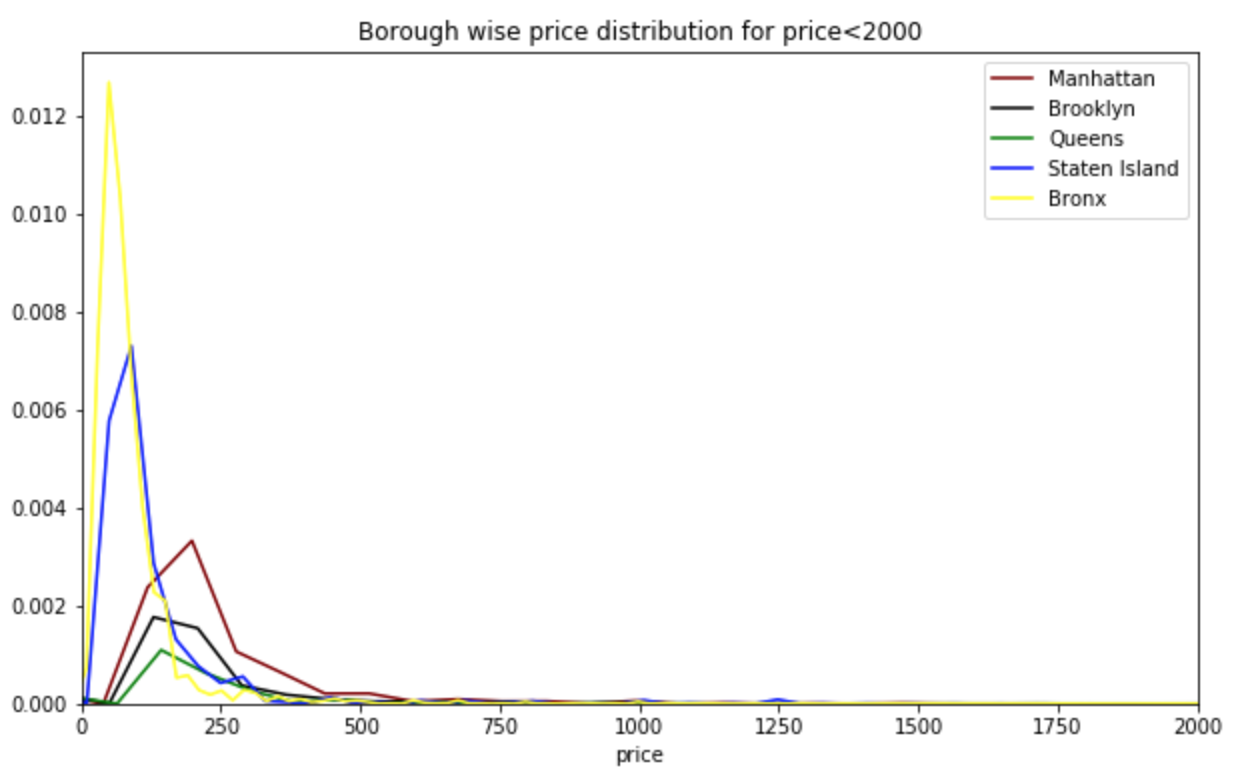
This is an interesting plot as we can see how the data is distributed overall, we can get the sense of how the different boroughs of New York city are placed and the area they occupy within the New York city.

1. **Variations of prices of hotels with area:**

In order to see the price variation as per the location, we can take the column NEIGHBORHOOD.GROUP and see the PRICE column for each group variations.



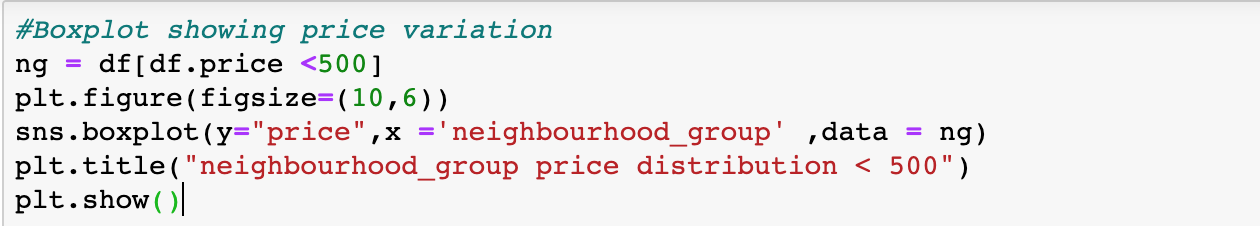
**Output:**



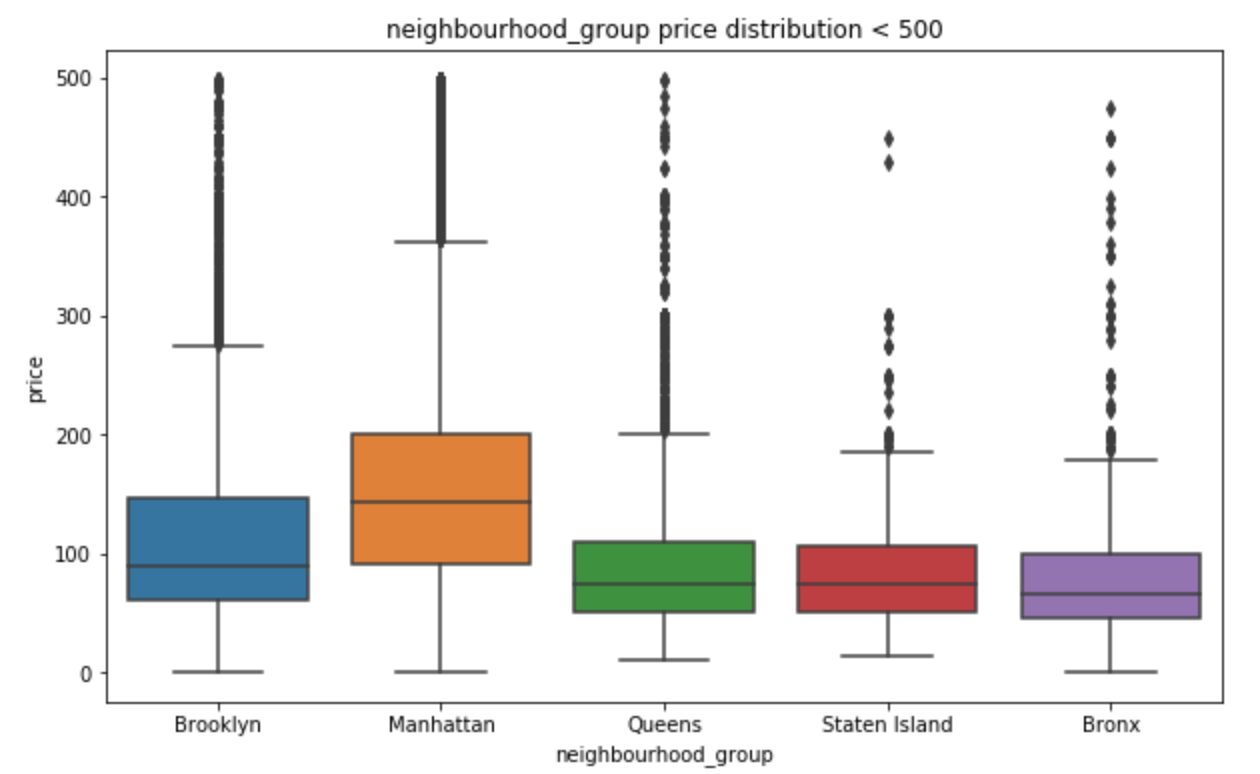
From the plot we can say that Bronx and Staten island are having lot of price variation from 10 to 200 (basically a lot of cheap places to stay). Whereas Manhattan and Brooklyn and Queens are somewhat costlier than the Bronx and Staten Island.

1. **We can also see the boxplot for variation in price:**

When we plot boxplot of the PRICE Vs NEIGHBOURHOOD\_GROUP. The price range that we are taking into consideration for this is from 0 to 500. Because from above graph we can see that most of the prices lie between this price range.



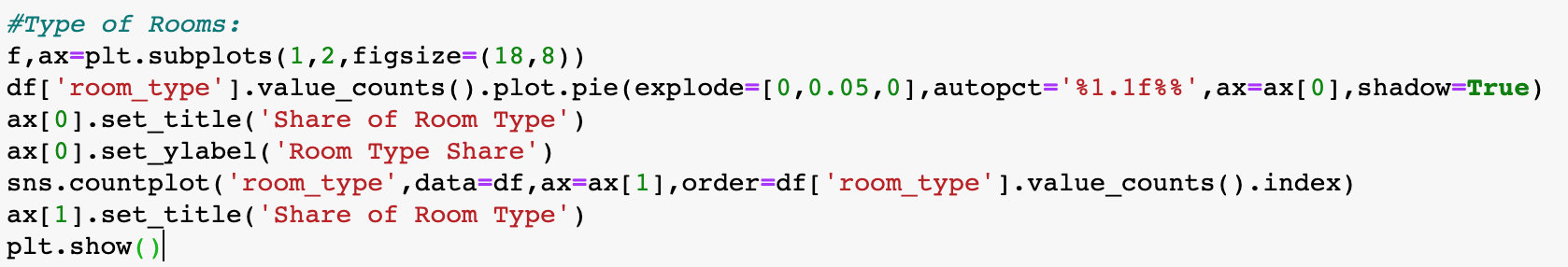
**Output:**



The boxplot tells about that the median, and the ranges of the price. As you can see the median for Queens, Staten island and Bronx are almost the same and the prices for Manhattan and Brooklyn are higher.

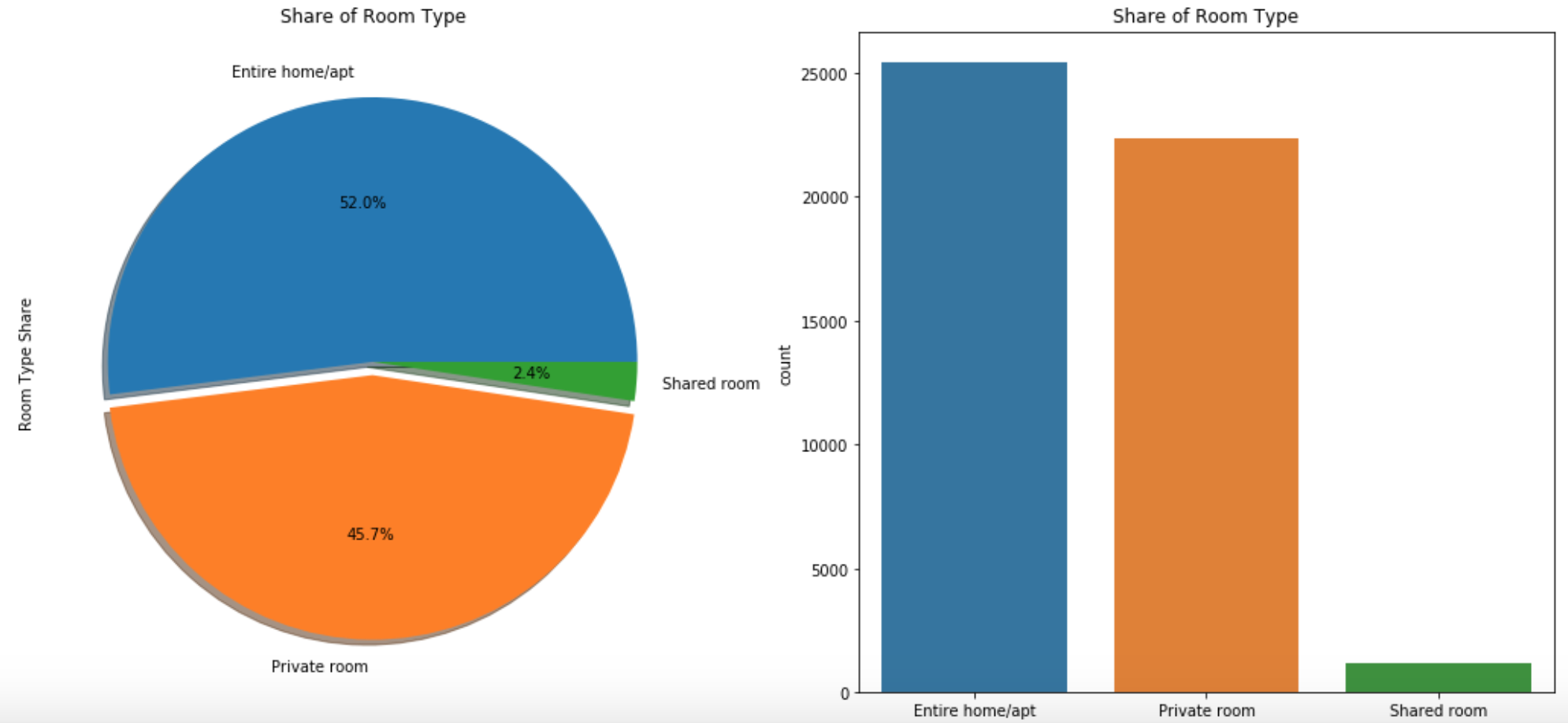
1. **Type of rooms:**

In the ROOM\_TYPE column, there are three different type of rooms – entire home/apartment, private room and shared room. We can check and compare the number for each type with the help of histogram or pie chart as below:



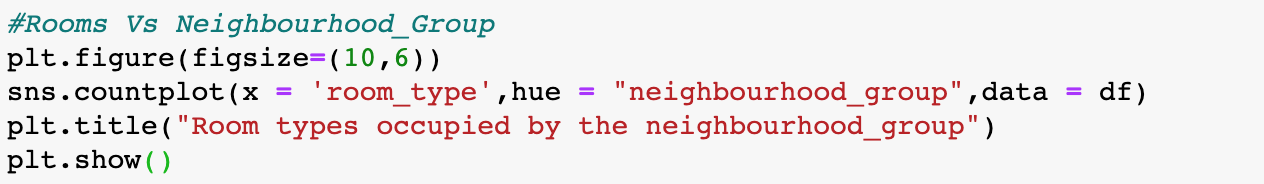
**Output:**

In the output, you can see that more than half of our listings belong to Entire home or apartment category, for which the count is around 25000. Then we have private room which is also around 45% (i.e. around 22,500). Rest of the rooms belong to shared rooms.

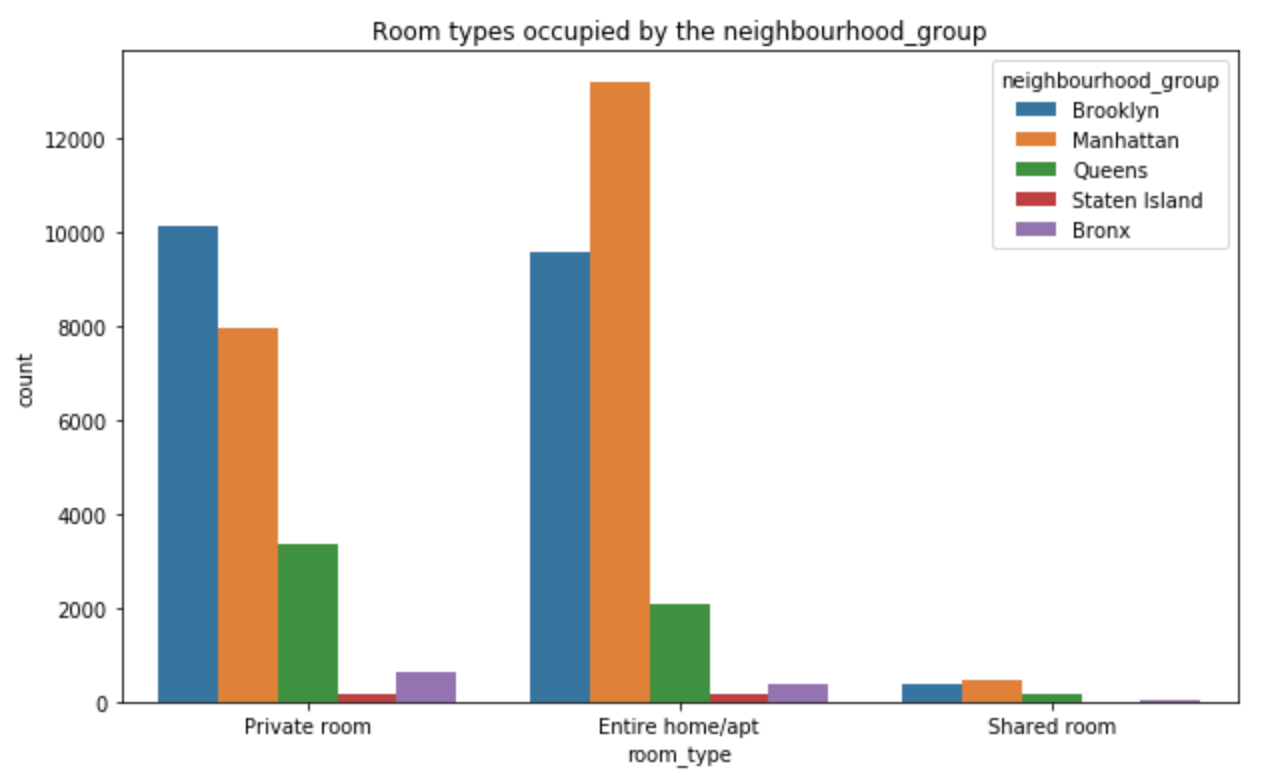


1. **ROOMS vs NEIGHBORHOOD\_GROUP:**

We can also see the distribution of different types of rooms with respect to each borough. Through this graph, we can get the idea of what to expect where. For ex. We can expect high number of private rooms in Brooklyn area. Whereas Manhattan has large number of entire home or the apartments.

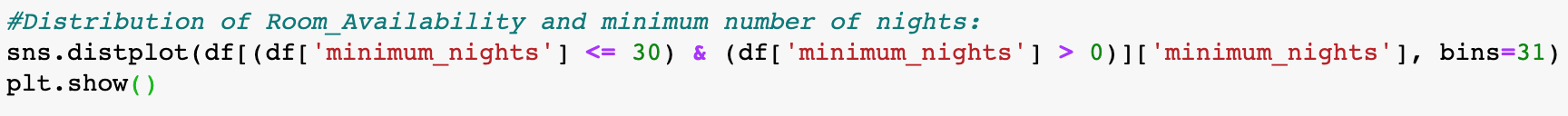


**Output:**



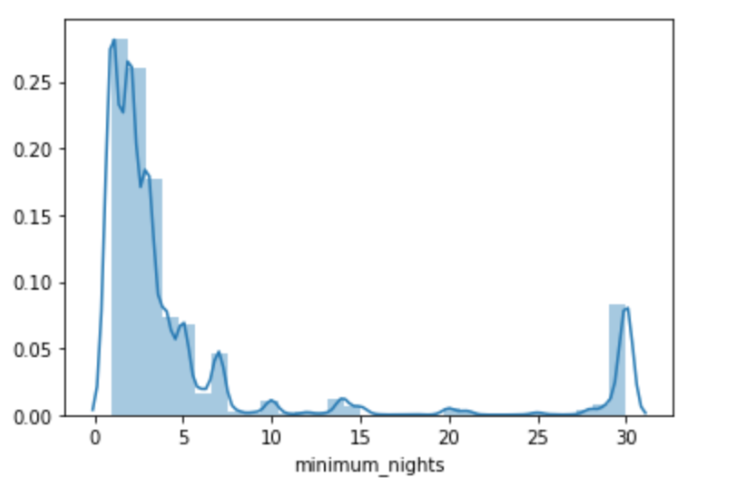
1. **Distribution of ROOM\_AVAILIBILITY and minimum number if nights:**

We can also have a plot that can help us understand what’s the minimum number of durations that one can get. We can plot the MINIMUM\_NIGHTS using DISTPLOT (). This function plots the histogram with a line over it.



**Output:**

Below graph tells us that most of the listings offer between 1 to 5 days of minimum booking. There are also some of the listings which offer minimum of 30 days of stay which also help us understand that some of the listings are looking for people to stay on a monthly basis. Here we have considered a limit as 30, just to showcase the trend.



1. **Model implementation and testing:**

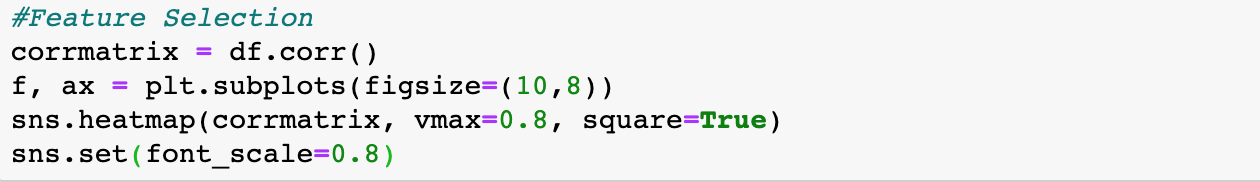
In this section we will see how different modelling techniques provide different results with the predictions.

Before starting with the model selection, we should first decide which features to select and which to drop. As there are lot of input parameters in our datasets. Hence, we need to perform feature selection to reduce the number of input parameters. this will not only reduce the computational time and model building cost but also it will help in improving our predictive model.

In order to understand what features to select, we can take help of correlation plot.

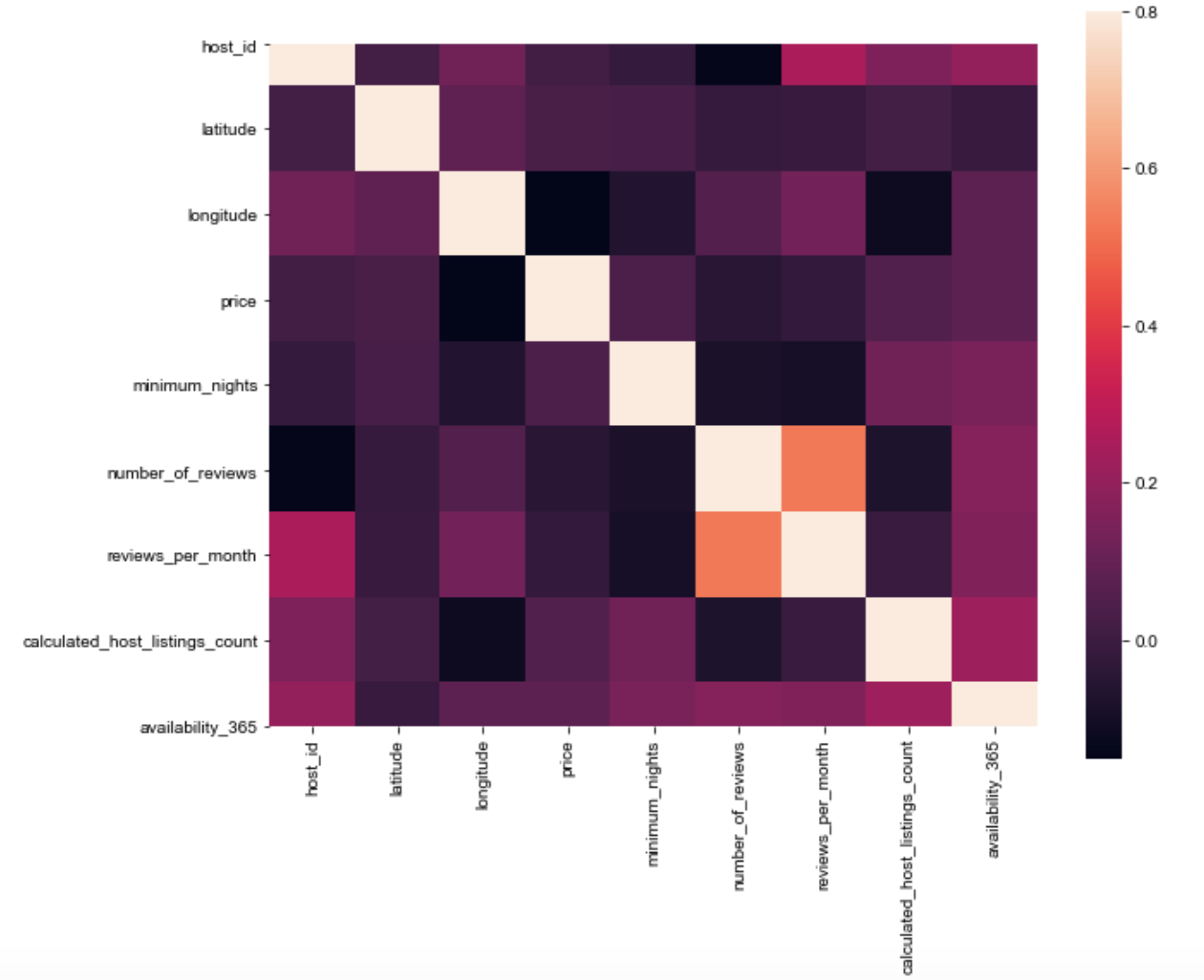
“Features with high correlation are more linearly dependent and hence have almost the same effect on the dependent variable. So, when two features have high correlation, we can drop one of the two features.” (R. Vishal (sep, 2018)

.

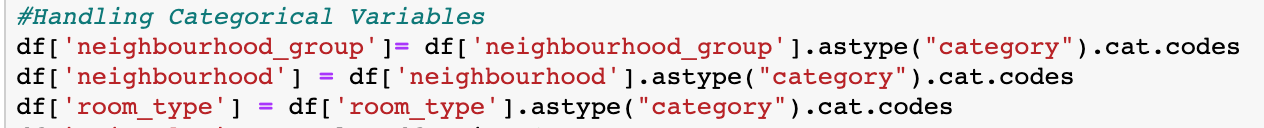


**Output:**

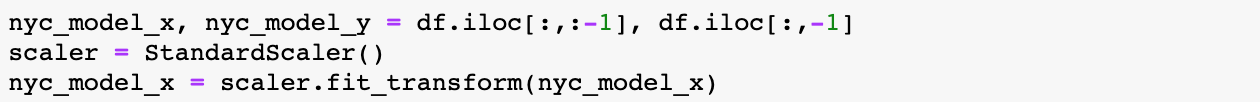
In the correlation plot, we can observe that there is high correlation between number of reviews and reviews per month.

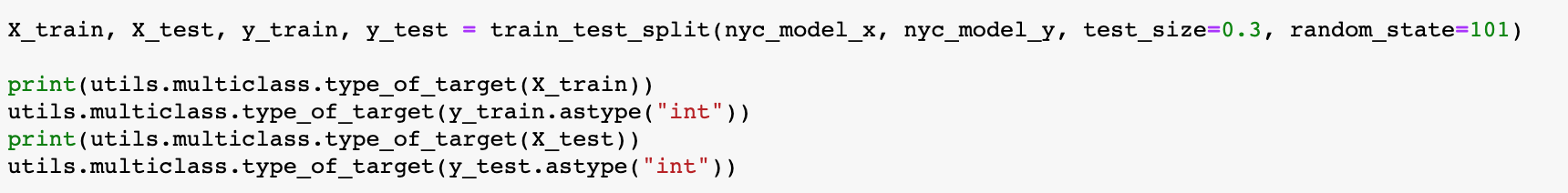


To handle the categorical variable, we need to firstly, categorize them by converting them into numbers and to do this we can use ASTYPE(‘CATEGORY’) CAT.CODES.





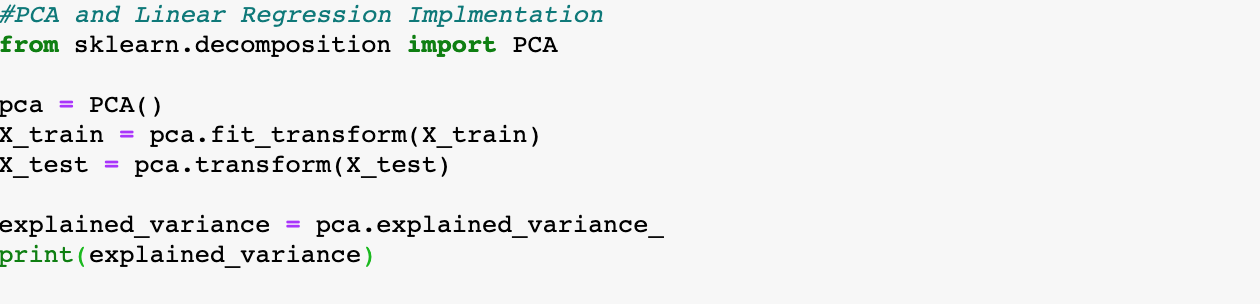




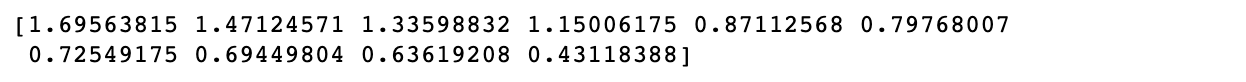
1. **PCA and linear regression implementation:**

PCA is done in order to reduce the present dimensionality in the data. It can also be used to describe the variation present in the data. To use PCA, we need to first import it from SKLEARN.DECOMPOSITION library.

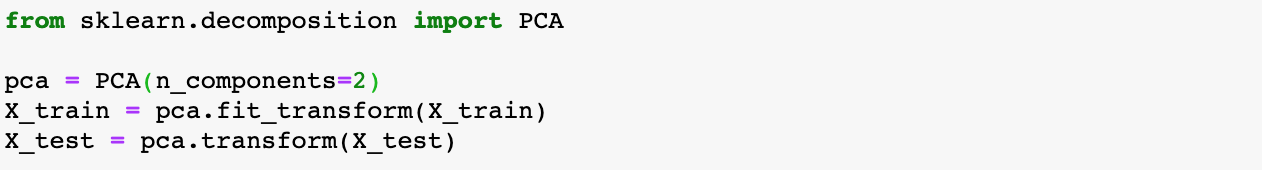
To use PCA, we need to standardize our data. We will normalize our data within a particular range with the help of transform and FIT\_TRANSFORM function. We will do this on our training data first and then transform or standardize our testing data. We can check the discrepancy between the model and our test data with the help of EXPLAINED\_VARIANCE method.



**Output:**

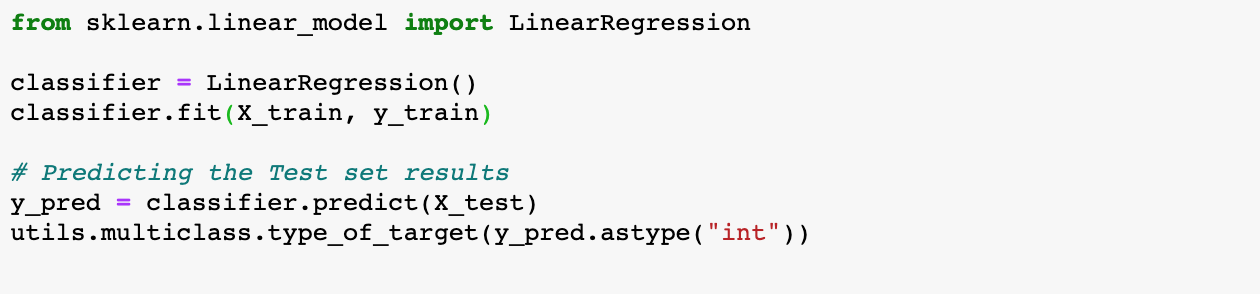


We got this as the output. Thus, we can say that there are two principal components which explains the entire variance of the model. Then we fit and transform our data by the instance of PCA which was created for two principal components.

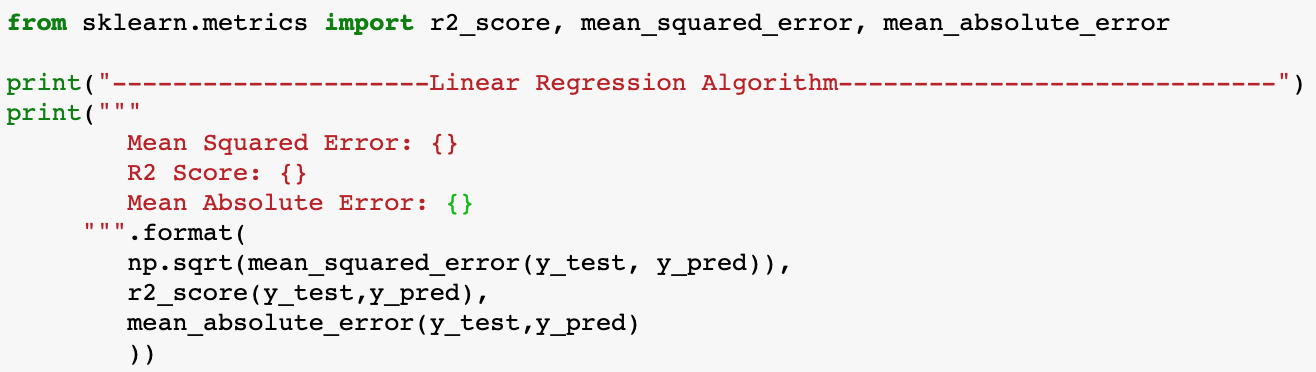


In order to perform linear regression on our dataset, we need to first create a model based on our train data. Then we will test the model with the help of test data. We are creating test and train groups by using TRAIN\_TEST\_SPLIT () function. This function will be imported from SKLEARN.MODEL\_SELECTION library.

Now we create the instance of linear regression and we will try to see how the designed model handles the test data. We want to fit our Y\_TRAIN to our X\_TRAIN.

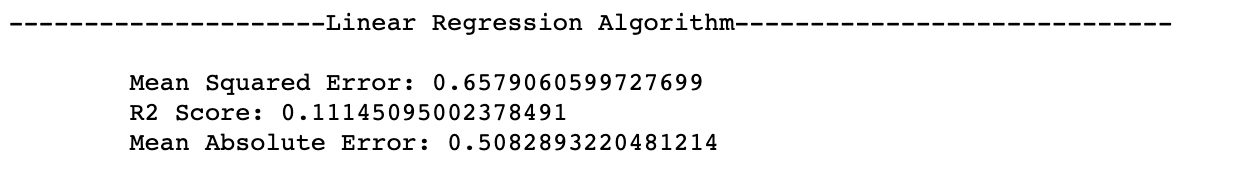


Now we will check the mean square error between our predicted values and actual test value. In order t do so we have written the following code:



**Output:**

As you can see the R2 value is 0.111 and mean squared error is 0.65 this means the difference between the actual and the predicted value is 0.65 which is quite high. Let’s see how the ridge will perform for the same.

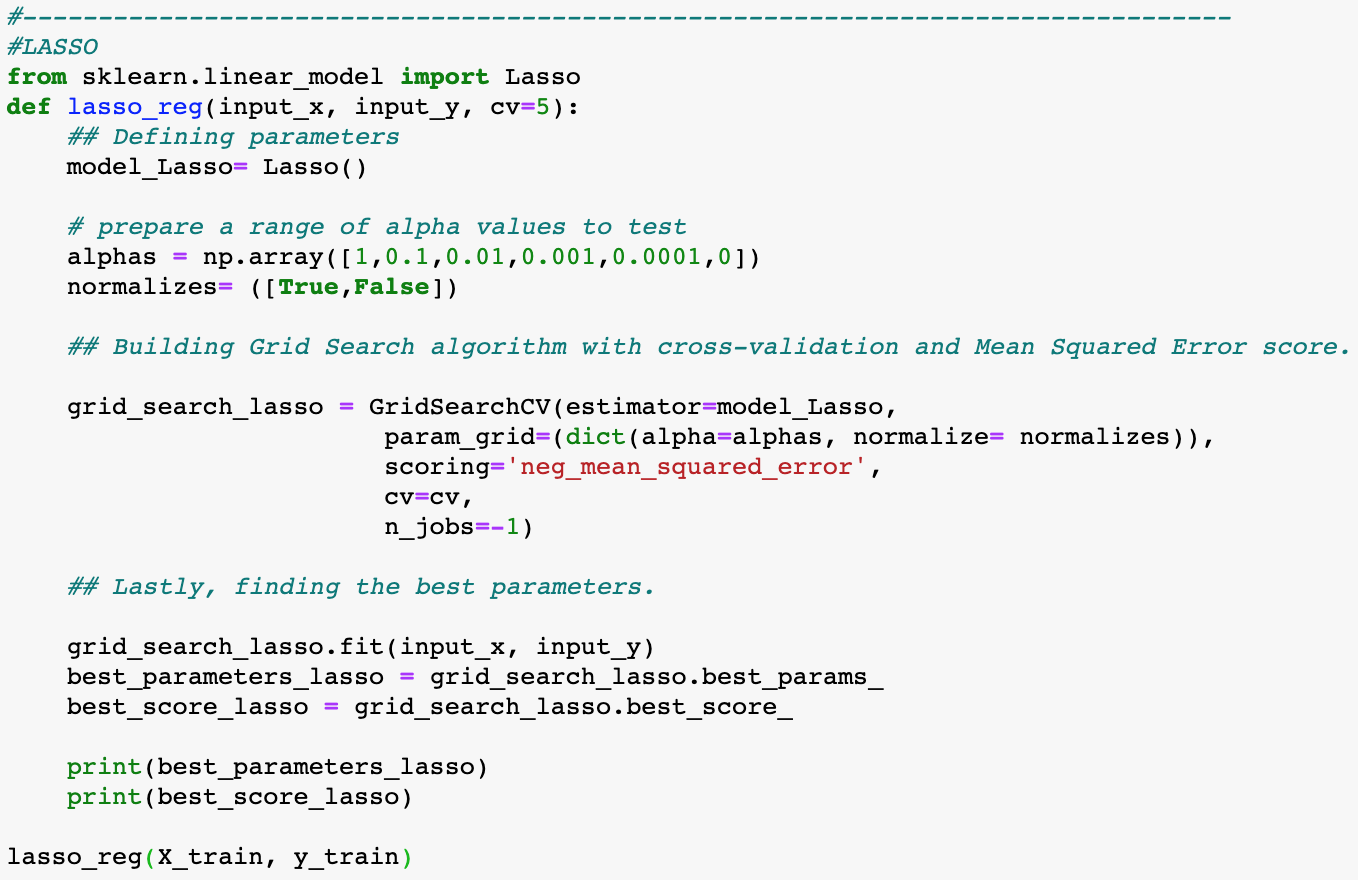


1. **Lasso Regression:**

Lasso regression as we all know is a technique through which we provide bias to the regression line so that we can get better future prediction.

As we know, every dataset has noisy samples. For example, the house size wasn’t measured accurately or the price is not up to date. “The inaccuracies can lead to a low-quality model if not trained carefully. The model might end up memorizing the noise instead of learning the trend of the data.”Chakon, O. (2017)

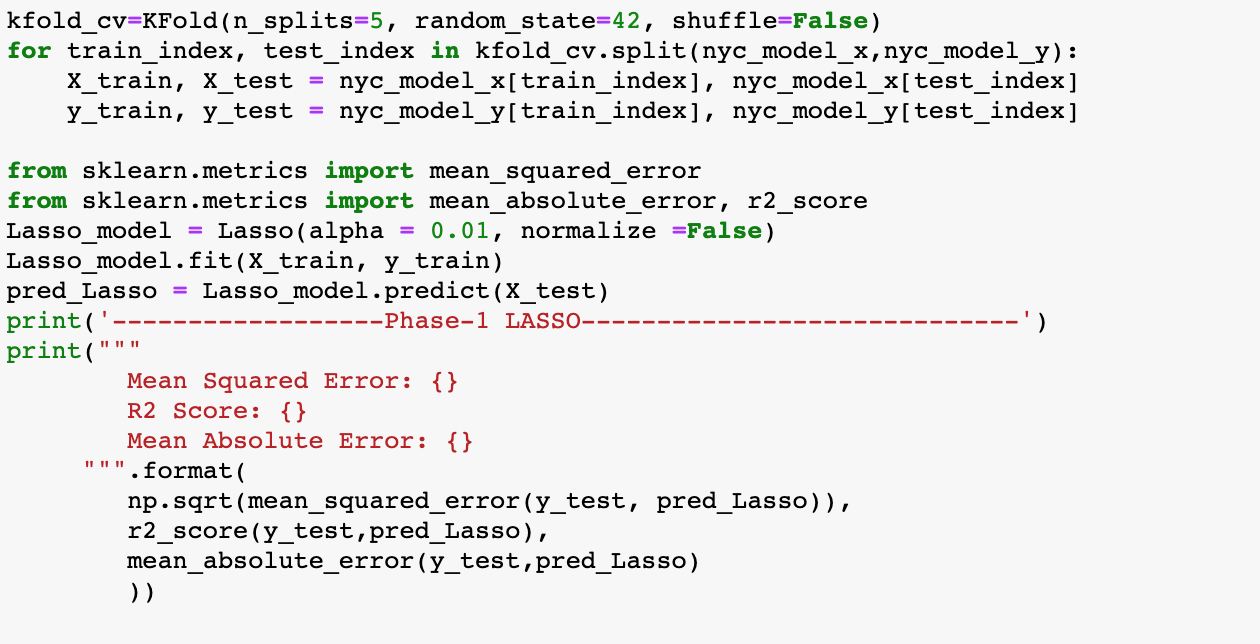
In this we firstly create an instance of the lasso model () then we create an array of different values of alpha to test. We will perform the grid search to find that value of alpha for which we are getting the best parameters.



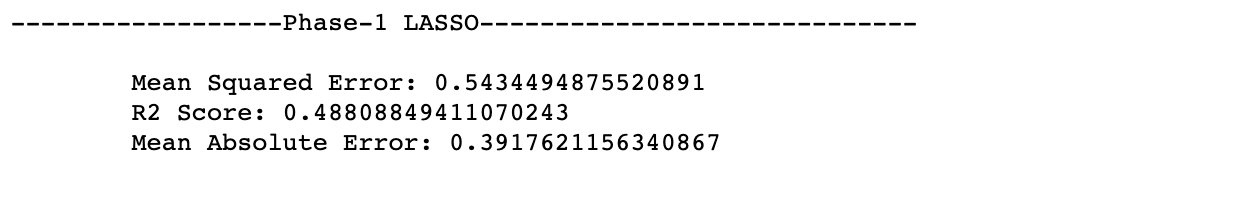
We will test this function with our X\_TRAIN and Y\_TRAIN to see for what value of alpha we get the best model. After running our function, we see that the for alpha 0.01 we get the best parameter value.



K- fold cross validation, to obtain the best model we will perform a resampling method, in this we will divide our datasets in different sets which are called as folds. We will keep changing our test dataset to see which model helps in getting the least possible mean square value.



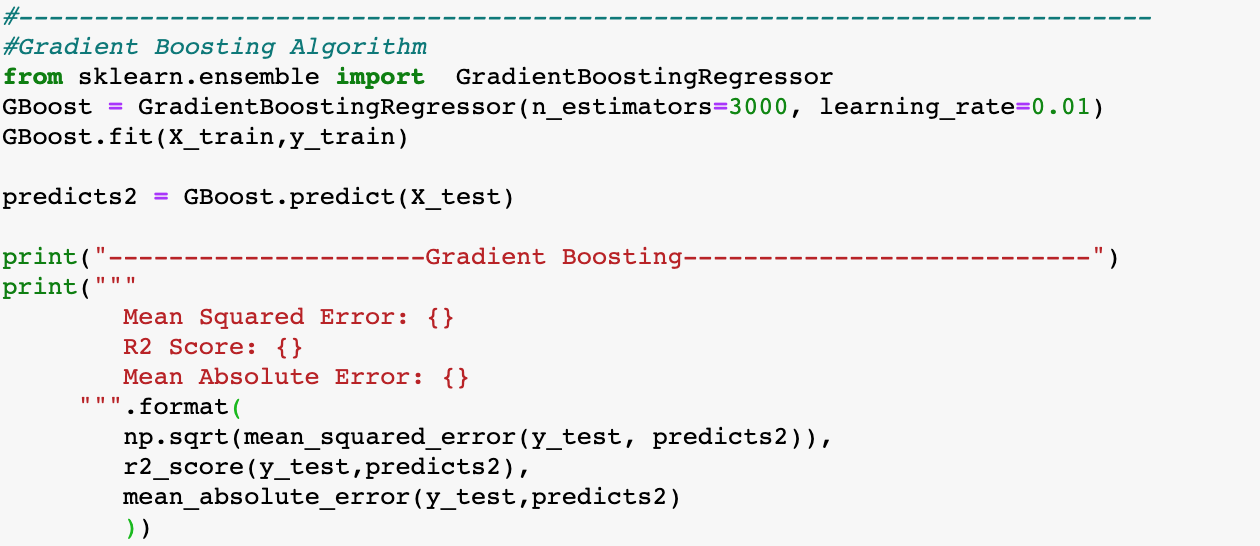
Since we have obtained best alpha value as 0.01 hence, we will take that as our alpha value for the lasso K -fold model. if you go above you can see that the R2 value for the linear model is very low compared to what we are getting for the lasso model R2 value, we can say this model is definitely better than the linear model.



1. **Gradient boosting algorithm:**

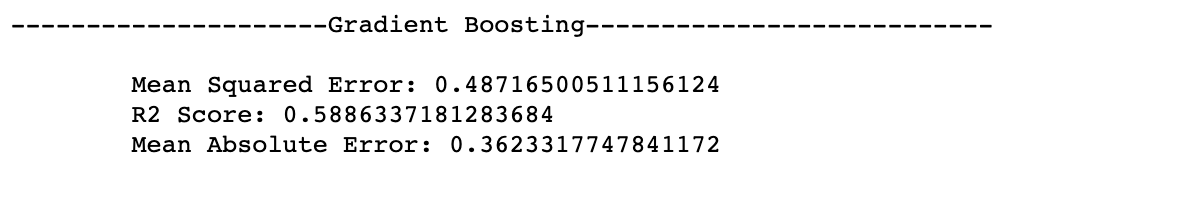
Gradient boosting basically helps the weak learners in the model to learn better. For example, let’s consider Tree 1 of the Decision Tree, it classifies some of the observation easily, this becomes our strong learners, our Tree 1 classifies some of observation incorrectly. Our task is to help these weak learners. Gradient Boosting algorithm computes the loss function, further gradient, and attempts to optimize only certain coefficients of cost function, hence the model is trained better.

We have used GRADIENTBOOSTINGREGRESSOR () method to implement regression using gradient boosting. This function has been imported through SKLEARN.ENSEMBLE. we will fit our model and then see the prediction that our model provides and compare it with the above methods.



**Output:**

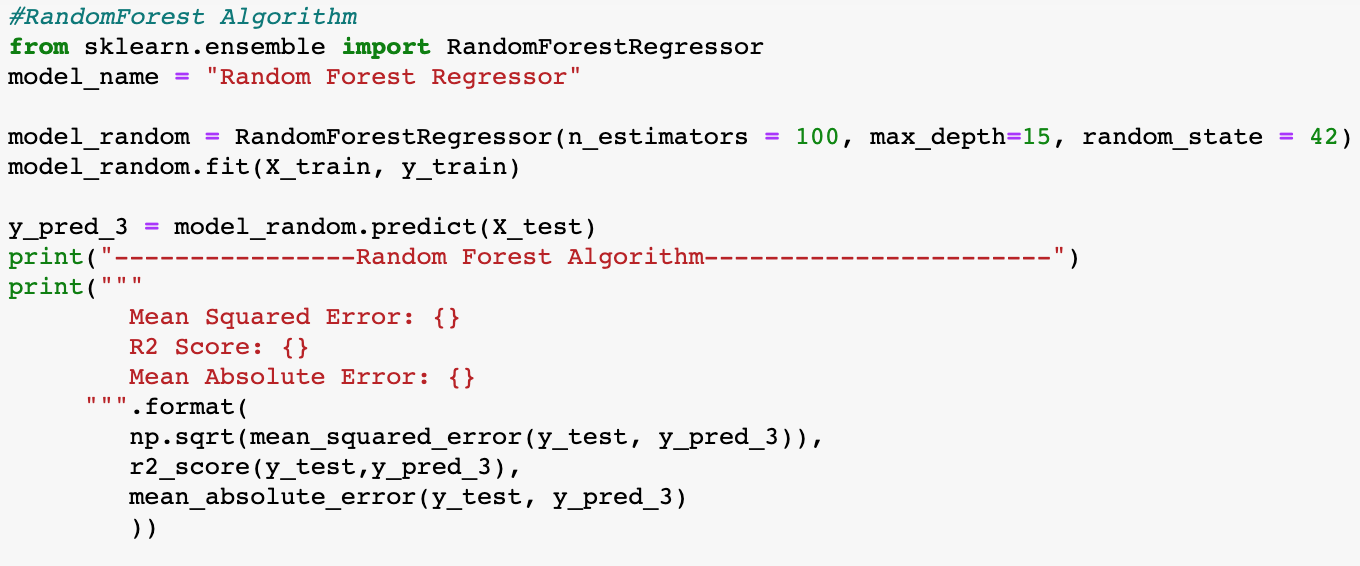
As you can see that we have r2 nearly 60%. And mean square error is coming as 0.49. this help us understand that the model is able to predict in a better way than the lasso and linear regression.



1. **Random forest algorithm:**

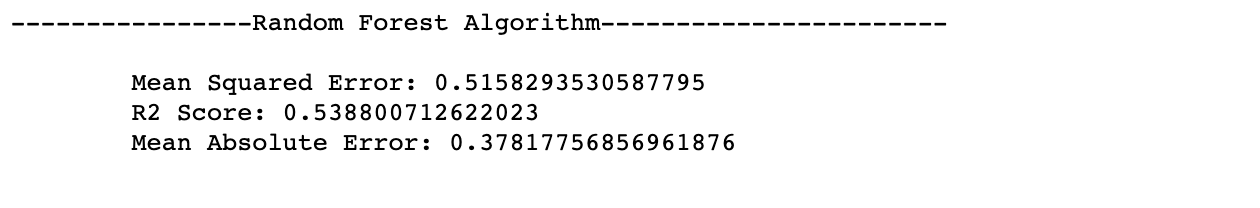
Random forest is a collective learning method for classification, regression and other tasks that operate by making a multitude of decision trees at training time and then we get outputting the classes that is classification or regression of each tree. This technique helps in correcting decision tree habit of overfitting to the available training data set.

To perform this method, we will import randomforestregressor() from the sklearn.ensemble. we will fit our training dataset and use the predict function to see how well our data predict compared to the actual values.



**Output:**

R2 square value that we get through this is 0.53. That means our developed model has 54% correlation and the mean square value is coming as 0.52.



1. **Neural network implementation:**

Neural network is a concept of deep learning. Deep leaning is one of the parts of machine learning. Through neural network, we try to recognize pattern and this is done by creating a web of algorithms which will generate some function and each function will compute some value if the observed value is above the threshold then we select that neuron to stay active and send the information to the next stage. This is how the information propagate.

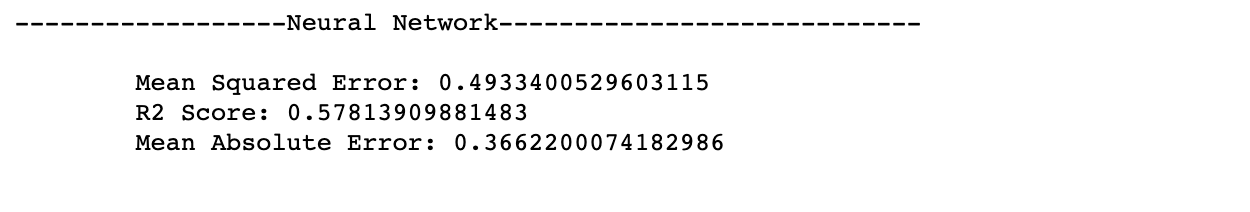
The neuron activation is determined by activation function, activation functions are threshold function. If the value is above the threshold then only the next neuron will be activated. We should also know that the difference between the prediction and the actual values are done at each layer. And this process is run in both forward and backward direction. Thus, we can say that most of the data processing is done in the hidden layers.

We need to use function MLRegressor to perform Neural network regression. We will implement this on our training dataset. Once we get our model, we will predict the value for Y based on our y-train.



**Output:**

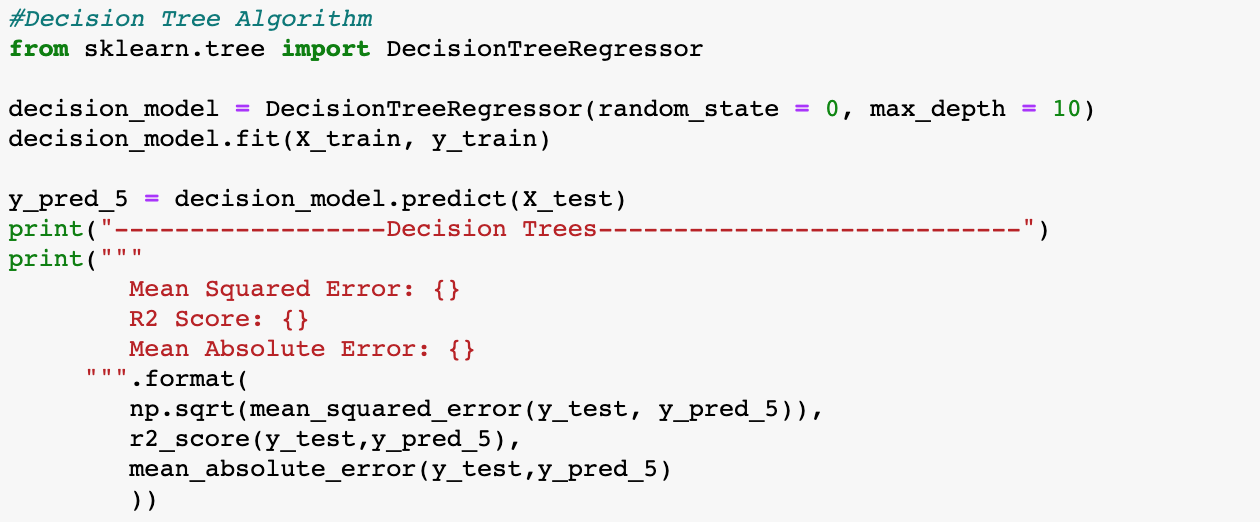
As you can see that we are getting the value for neural network implementation as 0.57 and mean squared error is showing us how far we are from the actual values. We have mean square value as 0.49. if we take MSQ into consideration then this model seems better in predicting when compared with random forest.



1. **Decision tree implementation:**

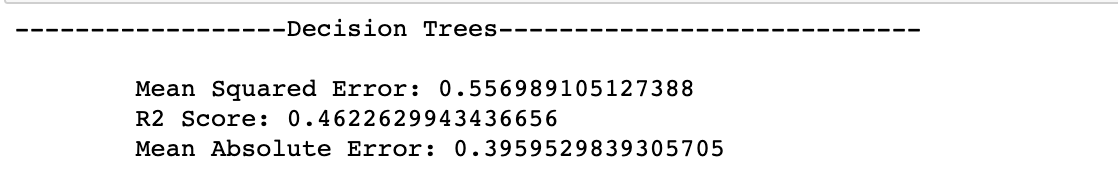
In simple terms, decision tree is nothing but the grouping of data. To understand how the decision tree works, we can look at a simple example of amazon. In amazon once a product comes, it was assigned to particular group and then there are several groups under that group and the several under that. This classification of data makes that object to be placed in a very precise section. So, in case if the user is looking for let’s say laptop accessories then all the object who belong to that subgroup will be shown to user.

To implement decision trees in python, we have to first import decisiontreeregressor function form the sklearn.tree. then we try to fit our dataset into this decision tree model. and check how our model is predicting for the testing data.



**Output:**

We will check how the model is performing by calculating it’s R2 and MSQ values. R2 value for this model is found to be 0.46. Mean squared value is found to be 0.55. This says that our model is right almost 50% of the times.

****

1. **Summary:**

We have successfully analysed our dataset of Airbnb listings.

To clean our dataset, we preformed forward and backword filling. In order to fill the Null values for the numerical datatype, we considered filling the cells by the mean value, which was calculated based on the available data.

In the data visualization part, we analysed different plots like Placements of the Hotels, Variations of prices of hotels with area, Boxplot showing price variation, Type of Rooms, Rooms Vs Neighbourhood Group, Distribution of Room Availability and minimum number of nights. We also gained insights like how the room distribution is as per different boroughs and what kind of prices can you expect in which area.

After that we moved onto the analysis of different models. We segregated our dataset in training and testing and then checked how well our model is performing with respect to testing dataset. For the comparison, and to see how well our model is at predictions, we took different parameters into consideration like R2, mean square error and mean absolute error.

*Table 2. R2 ,MSE and MAE for different models*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Models** | **Linear** | **Lasso** | **Gradient boosting** | **Random Forest** | **Neural Network** | **Decision Tree** |
| **R2** | 0.111 | 0.488 | 0.588 | 0.538 | 0.538 | 0.578 |
| **MSE** | 0.657 | 0.543 | 0.487 | 0.515 | 0.515 | 0.493 |
| **MAE** | 0.508 | 0.317 | 0.362 | 0.378 | 0.378 | 0.366 |

From the above table we can say that for this particular dataset, Gradient boosting is somewhat better than the other modelling techniques.

We can further implement additional modification for our model like decision tree, neural network. To improve their performance. Like increasing the number of hidden layers or if we can include the dropout layer for the decision tree we may get some other model as the best one. But with the components that we have currently taken into consideration we found out that gradient boosting is the best option.

1. **References:**
2. Vishal, R. (2018). Feature selection — Correlation and P-value. (2019). Retrieved 14 February 2020, from <https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf>
3. New York City Airbnb Open Data. (2020). Retrieved 15 February 2020, from <https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data>

#### Chakon, O. (2017) Practical machine learning: Ridge Regression vs. Lasso. (2017). Retrieved 15 February 2020, from <https://hackernoon.com/practical-machine-learning-ridge-regression-vs-lasso-a00326371ece>

#### Keen, b. (2017). Gradient Boosting in python using scikit-learn – Ben Alex Keen. (2020). Retrieved 15 February 2020, from <http://benalexkeen.com/gradient-boosting-in-python-using-scikit-learn/>

#### Simplilearn (Jun, 2019). Neural Network In 5 Minutes | What Is A Neural Network? | How Neural Networks Work | Simplilearn. Retrieved 15 February 2020, from <https://www.youtube.com/watch?v=bfmFfD2RIcg>