Prediction Analysis on Price of Boston, New York and Seattle Airbnb Listings

Manideep Nune
Maters in Computer Science
University of Ottawa, Ottawa,
Canada
mnune055@uottawa.ca

***Abstract*— The Airbnb Price Prediction is a supervised regression problem addressing the prediction of Airbnb price using cartographic features. This project analyzes Airbnb listings in the cities of Boston, New York and Seattle to better understand how different attributes such as bedrooms, location, house type amongst others Can be used to accurately predict the price of a new listing, which is the best for the landlord's profitability but affordable for their guests. This model is designed to help Airbnb provide landlords with internal pricing tools. There are many ways to accomplish his goal, but more specifically, I will try and predict the best price for any Airbnb given standard measures such as location of the listings, the date of pricing, and the features that Airbnb offers etc. This can take several different forms of projects but my most promising idea is to try to recommend a price for an individual looking to put a listing on. The analysis begins with exploring and examining the data to make necessary data preprocessing, feature selection that can be conducive for a better understanding of the problem at large. Moving further, machine learning models are built that are intuitive to use to validate the hypotheses on pricing and availability and run experiments in that context to arrive at. This project then concludes with a discussion of business impact, related risks, and future scope.**

Keywords— Regression model, machine learning, dataset, features

I. INTRODUCTION

Data plays a pivotal role in world today. Buzzwords we often hear technical discussions have a lot to do with data. Millions of companies can make decisions based on forecasts derived from analysis of quality data. Various other key areas of governance, armed forces, telecommunications, entertainment, education, e-commerce, etc. also rely on the results of data analysis as a guide to performing various operations including key decisions. So, it's no surprise if someone infers that data science is one of the most "on-the-fly" fields in the world today.

As we all know, machine learning plays a vital role in data science because it is used by data scientists to build components (machine learning models) to help complete various tasks. Since creating these models requires a lot of data for learning, it is very important that we understand every detail of the data used and how they affect the machine learning model we choose for the data.

Airbnb is an American company staying in the United States Platform that enables users to rent or share their property in a reliable, comment-based, active community. Airbnb provides its homeowners with a completely independent home pricing function, providing only a minimum pointer so that landlords can compare similar lists nearby to get a competitive price. Could be the host for any additional amenities they may find necessary. As the number of landlords using Airbnb grows, it is imperative to offer the right price to keep the landlord competitive. On the flip side, guests using Airbnb are often plagued with non-availability of accommodation due to a variety of reasons.[8]

This is an interesting project because it involves many elements of behavioral economics, game theory, and the disruption of the technology industry. As the Airbnb concept is increasingly becoming a reality in the hospitality industry, listings and competition across Airbnb are increasing. Therefore, from the moderator's point of view, it is difficult to know whether everyone is maximizing their income potential, or whether they are eating away at their performance due to over-pricing. Similarly, it is unclear whether the landlord can invest in other conveniences in substantial benefits, such as having more bedrooms available or simple facilities such as providing toiletries. For Airbnb, this is very important for the company, as their success depends heavily on creating a market that is easy to navigate for current and new landlords.

The job of predicating the price using the categorical data can be done by developing a machine learning model that uses an algorithm that learns from a set of data (supervised learning) and performs multi-class classification. The challenge is to identify the machine learning algorithm and training method that provides the optimum result. (which is, in my case, performance measures like mean square error, Absolute error, root mean square error and variance of the model). I will be using various data analysis, data pre-processing techniques.

II. PROBLEM DOMAIN DESCRIPTION

Previously mentioned, I have taken the three data sets from Kaggle repository which includes Airbnb listings of three famous cities Boston, New York and Seattle.

The main problem domain is hosts on Airbnb experiment and charge an optimal price. So, can we analyze similar lists in the past to recommend the best way for hosts to charge for a new listing? There is no way to determine when the list is hosted or when the list is unavailable. Host / Guest Change Plan. So, with past information about vacancies, can we recommend guests who have a list?

III. DATA ANALYSIS

A. Data Source

The dataset we are using is provided by “Kaggle”, an open source data repository, whose pure purpose is to provide clean data on all the Airbnb listings/reviews from each of the main metropolitan cities in which Airbnb operates in. The neat thing about this dataset is that we get all the public information as provided on the official Airbnb website, regarding each and every single listing/review, all properly

stored in CSV format. This is a huge benefit as it significantly reduced the time needed for us to crawl and clean the data ourselves.

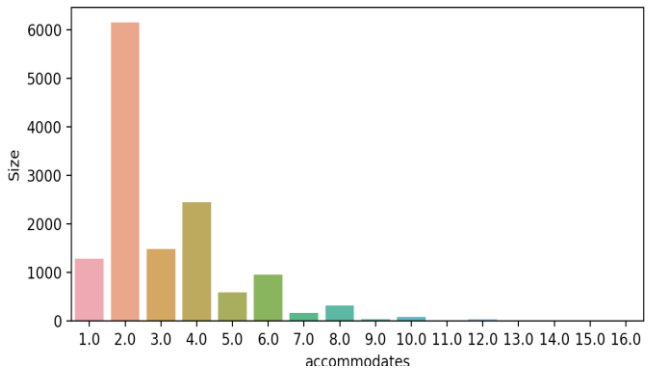
Out of the many metropolitan cities provided by “Kaggle”, I chose to do my research on the city of Boston, New York and Seattle. With New York being such a popular tourist destination, it became the city with the most Airbnb listing in the world. This was crucial as the more data we have, the easier it is for us to train our model without overfitting on a small subgroup of information. As a result, we ended up with a dataset of 51,721 unique Airbnb listings located in Boston, New York and Seattle.

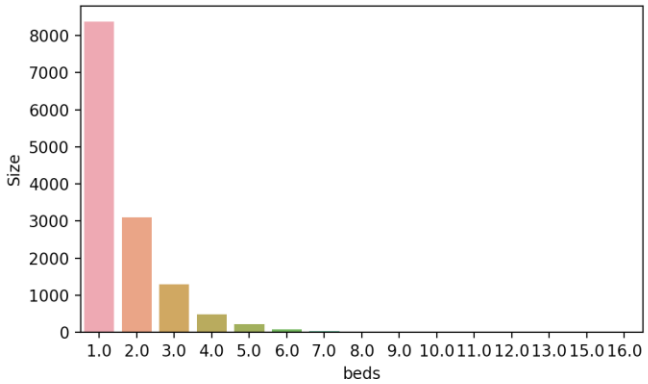
Basic Features:

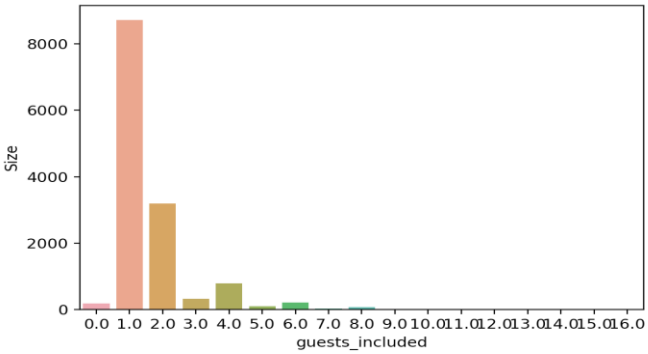
1. host_is_superhost (categorical -YES/NO) whether the host is an Airbnb “Superhost"
2. host_identity_verified (categorical - YES/NO) whether Airbnb has verified the identity of the “host”
3. property_type (categorical - 17 levels) the type of property
4. room_type (categorical - Entire home/Private room/Shared room) the type of room
5. accommodates (continuous) the number of people the property can hold
6. bathrooms (continuous) the number of bathrooms the property has
7. bedrooms (continuous) the number of bedrooms the property has
8. beds (continuous) the number of beds the property has
9. bed type (categorical - Airbed/Couch/Futon/Pull-out Sofa/Real Bed) the type of bed
10. guests included (continuous) the number of guests allowed
11. minimum nights (continuous) the minimum number of nights for a reservation
12. *price* (continuous) the feature we are regressing on the daily price of the property in dollars

B. *Exploratory Data Analysis*

In the beginning of our analysis, it is essential to visualize the data and especially the variables we are interested in. For the response variable, price, the mean, median, standard deviation provides us with information regarding its distributional properties. Based on these values, it is clear that the price distribution is not normal, but is skewed to the right, resulting in a right heavy tail. Shows statistics such as Rotting Mean Square Error, Mean Absolute Error, Variance, and Mean Square Error. These statistics can be used to gain insights from the data in addition to the graphs. Below graphs visualize the data of different features.







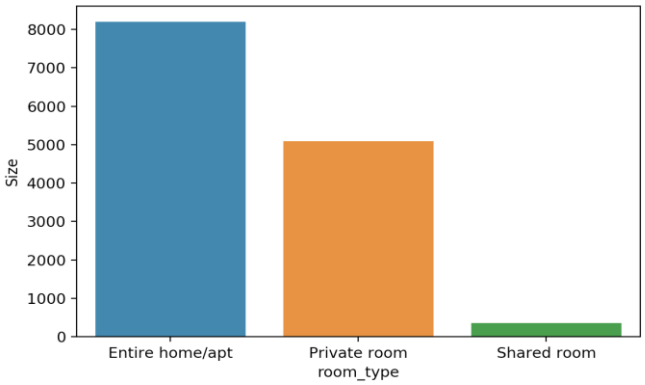


Figure 1: Visualizes the data of different features

C. *Evaluation metrics*

1. RMSE (Root Mean Square Error)

It is the simple standard deviation of the differences between the values that are predicted and observed [1]. calculated using this formula:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

2. MAE (Mean Absolute Error)

It is the average of the absolute difference between the predicted values and observed value. The MAE is a linear score which means that all the individual differences are weighted equally in the average. For example, the difference between 10 and 0 will be twice the difference between 5 and 0 [1]. Mathematically, it is calculated using this formula:

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

3. MSE (Mean Square Error)

MSE is the sum of squared distances between our target variable and predicted values. [1]

$$MSE = \frac{\sum_{i=1}^{n}(y_i - y_i^p)^2}{n}$$

4. Variance

This is the amount that the estimate of the target function will change if different training data was used. The predicted value is estimated from the training data from different machine learning algorithms and we get some variance for the output predicted. [1]

D. Scan for missing values

Real-world data is rarely clean and uniform. They are often incomplete, noisy, and inconsistent, so camouflaging data by filling in missing values is an important task for data scientists. It is important to deal with them, because for any given model, they can lead to incorrect predictions or classifications. [10]

Missing information can also appear in multiple ways, it can be an empty string, it can be NA, N / A, None, -1 or 999. One special way to prepare for missing values is to understand your records: Know how missing values are represented, how to collect data, and no longer assume missing values and where they are used, especially not indicating missing data. Understanding of domain knowledge and statistical knowledge is the most necessary factor to properly handle missing values. In my data set, I can see that after selecting the features to consider, 14 rows contain missing values. This is small compared to the 4870 total rows in the dataset, so we can delete them without having to worry about impacting the analysis.

E. Data Dimensions and Class distribution

After analyzing the data and visualizing the data distribution in a graphical representation, I found some imbalanced classification data, and the number of observations of one classification was significantly less than the number of observations of the other classifications. To solve this problem, I used different normalization techniques. These methods are explained later in the "Normalization" section.

F. Finding Skew and Kurtosis

Skew:

According to Wikipedia, in probability theory and statistics, skewness is a measure of the asymmetry of the probability distribution of a mean-valued real-valued random variable. Previously we checked for imbalances in the classes, and now I checked skewness because it affects the performance of machine learning models. If skewness value lies above +1 or below -1, data is highly skewed. If it lies between +0.5 to -0.5, it is moderately skewed. If the value is 0, then the data is symmetric. Once, we know the skewness level, we should know whether it is positively skewed or negatively skewed. [12]

Skewness	
host_total_listings_count	15.705754
accommodates	1.974423
bathrooms	3.328674
bedrooms	1.799334
beds	2.844079
guests_included	3.239098
minimum_nights	60.219923
number_of_reviews	3.795645
review_scores_rating	-3.068099
reviews_per_month	1.702585

Figure 2: Skewness of different features

Kurtosis:

Kurtosis is definitely the tail of the distribution, not kurtosis or flatness. It is used to describe the extreme value of one tail relative to another. This is actually a measure of outliers in distribution. Most kurtosis in the data set indicates a large tail or outlier in the data. If kurtosis is high, we should investigate why there are so many outliers. It shows a lot of things, could be wrong data entry or anything else. Low kurtosis in the data set indicates that the tail of the data is shallow or that there are no outliers. If our kurtosis is low (incredibly good), then we should also research and trim a bad result dataset. [11]

kurtosis	
host_total_listings_count	381.154698
accommodates	6.068893
bathrooms	17.683210
bedrooms	6.069395
beds	14.351915
guests_included	16.259077
minimum_nights	4278.740702
number_of_reviews	22.506509
review_scores_rating	16.980526
reviews_per_month	3.892882

Figure 3: Kurtosis of different features

G. Correlation Analysis

Correlation analysis is a statistical method used to assess the strength of the relationship between two quantitative variables. High correlation means that there is a strong correlation between two or more variables, while weak correlation means that these variables are hardly related. In other words, this is the process of studying the strength of this relationship using available statistics. This technique is strictly related to linear regression analysis, which is a statistical method used to model the association between a dependent variable (called a response) and one or more explanatory or independent variables. [13]

The distribution of listing's price is skewed towards the lower price of USD 100. Although most of the listings fall in this ballpark, roughly 5% of the listings have prices as high as USD 1000.

These outliers may influence the regression lines if a linear regression is fit on this data. We would perform an exercise to remove these outliers.

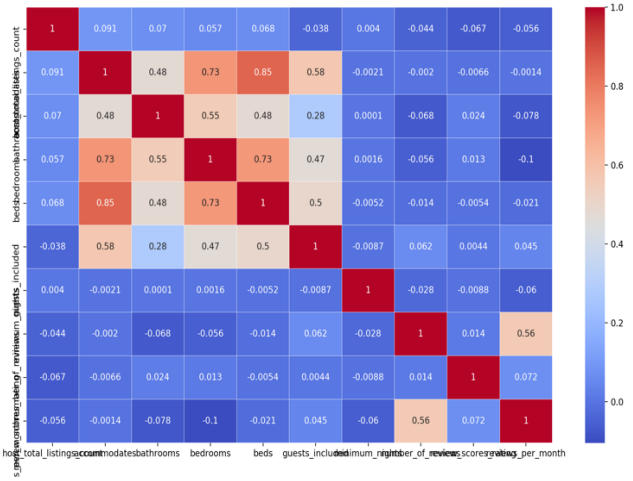


Figure 4: Heat map showing the correlation between the features

Another interesting property of this list is the number of bedrooms, which range from 0 to 15 (Figure 4). Although more than 97% of listings have less than 4 bedrooms, these disordered listings also affect models trained with this feature. Such checklists are rare, and we do not have a large number of data points to determine the regularity of these checklists. There may be a correlation between the number of bedrooms and the higher prices, or they may be independent. But listing at a higher price makes the regression fit, which is why we try to eliminate these outliers.

IV. DATA PRE-PROCESSING

Data preprocessing can have a significant impact on the generalization performance of commonly monitored ML algorithms. Noise removal is a difficult problem in inductive machine learning. In general, there are a lot of instances where most empty eigenvalues are removed. These high deviation characteristics are also known as outliers. In addition, a common approach to addressing what cannot be learned from very large data sets is to select a single sample from a large data set. Lack of data processing is another problem that is often addressed in the data preparation phase. [2]

The data set does lack some fields. However, most of these missing fields are related to quality aspects, such as missing comments, the host of information, and tips provided by the owner. Some quantitative information that were missing were host average response time, and if they provided weekly/monthly rent prices. To ensure the consistency of our data, missing text were left as empty strings, and missing quantitative information were filled in with zeros. Furthermore, we tried to avoid using features that were not present in every single listing data, as these data fields were often optional and could be described by other features. Therefore, it is extremely important that we pre-process our data before feeding it into our model. [2]

A. One hot encoding

To model classification variables, we use one-hot coding. Since I have a column called "amenities", it creates 15 columns with columns set to 0 or 1. Each virtual column is assigned one of 15 categories and the rows of that category contain the values "1" and "0" respectively. This type of binary representation of a classification variable is called a one-hot because each row has a value of 1 and the other row has a value of 0. When I say this is a binary representation of classification variables, it is a classification because the variables we are encoding are classification and each function is a binary, which means they take one of two values (i.e. 0 or 1). In other cases, we have other ways of representing classification variables such as vector representations, but in this way the one-hot representation is useful for our application.

heating	table corner	baby bath	air condition	dog(s)	children, books and toys	r
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	0	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	1	0	
1	0	0	1	0	0	
0	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	
1	0	0	1	0	0	

Figure 5: One-Hot encoding

B. Normalization

Normalization is a "scaling down" transformation of functionality. Within a feature, there is usually a large difference between the maximum and minimum values, such as 0.01 and 1000. When normalization is performed, the size of the value is scaled to a fairly small value [2]. This is important for many neural networks and k nearest neighbor algorithms. The two most common methods for this range are:

Z- score normalization: The result of standardization (or Z-score normalization) is that the features will be rescaled so that they'll have the properties of a standard normal distribution with

$$\mu=0 \text{ and } \sigma=1$$

where μ is the mean (average) and σ is the standard deviation from the mean; standard scores (also called ***z*** scores) of the samples are calculated as follows [3]:

$$z = \frac{x - \mu}{\sigma}$$

Min-Max normalization: In this approach, the data is scaled to a fixed range - usually 0 to 1. The cost of having this bounded range - in contrast to standardization - is that we will end up with smaller standard deviations, which can suppress the effect of outliers. A Min-Max scaling is typically done via the following equation [3]:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

For example, consider a data set containing two features, host_response_rate(x1), and review_score_rating(x2). Where host_response_rate ranges from 0–10, while review_score_rating ranges from 0–100 and higher. So, these two features are in very different ranges. The main aim of normalization is to bring the features of the data set into a common scale so that it will be easier and better for the machine learning algorithm to learn from the data. For this we use techniques Z-score calculation and min-max normalization.

But for normalizing our data set, I have used MinMaxScaler() function of Scikit learn. This automatically takes the min and max valued for the values of the feature in the data set and preforms the normalisation.

property_typ	accommodat	bathrooms	bedrooms	beds	guests_inclu
0.56	0.2	0.12	0.12	0.07	0.12
0	0.27	0.12	0.25	0.07	0.25
0.84	0.07	0.12	0	0	0.12
0	0.13	0.12	0.25	0.07	0.06
0	0.07	0.12	0.12	0	0.06
0	0.13	0.12	0	0.07	0.06
0	0.07	0.12	0.12	0	0.06
0.56	0.13	0.12	0.12	0.07	0.06
0	0.07	0.12	0.12	0	0.06
0	0.2	0.12	0.12	0.07	0.12
0	0.07	0.12	0.12	0.07	0.12
0	0.07	0.12	0.12	0	0.12
0	0.07	0.12	0.12	0	0.06
0	0.2	0.12	0	0.07	0.06
0	0.07	0.12	0	0	0.12
0	0.07	0.12	0.12	0	0.06
0	0.07	0.12	0.12	0	0.06
0	0.2	0.12	0.25	0.07	0.12
0.64	0.07	0.12	0.12	0	0.12
0	0.07	0.12	0.12	0	0.06
0	0	0.12	0.25	0	0.06

Figure 6: Normalization of features

C. *Dimensionality reduction*

Dimension reduction is very simple, it is the process of reducing the size of the feature set. Your feature set can be a data set (a feature) with a hundred columns, or an array of points that make up a large sphere in three-dimensional space. Dimension reduction reduces the number of columns to 20 or transforms a sphere into a circle in 2D space. [4]

The curse of dimensionality refers to all the problems that occur when using data of a higher dimension and does not exist in a lower dimension. As the number of features increases, so does the number of samples. The more features we have, and for all combinations of feature values to be well represented in our sample, the more samples we need. [5]

In my dataset, I have performed dimensionality reduction on few columns like for two features average_rating and rating_score, there is very high correlation between those two, so we can remove them. I have also removed one categorical feature called “reviews”, because it does not affect prediction, so we can remove for reducing the dimensionality.

V. Model Construction

In this paper, As I am trying to predict prices for certain Airbnb listings, the type of model we are looking for is a regression type of model. The most basic type of regression model we can choose from is linear regression. We will consider this type of model, but it might not perform ideally as the feature set, we have chosen to have different variables that correlate to one another.

To address this highly correlated independent variable problem, we also use distance-based regression models to help us estimate Airbnb's listing price. I would like to point out that we have more than 100 feature sets for each list. Due to the nature of the one hot coding, most of these features are 0, so eventually a very sparse feature set is obtained. To solve the second problem while retaining the advantages provided by distance-based regression, we incorporate rule-based regression and decision tree regression models. Finally, we also attempt to combine different models into a random forest regression model using integration technology. This method generally reduces the variance and bias of our data, thereby improving the estimation or prediction accuracy.

After the preprocessing of data and dividing it into test and train splits we have built different models using different kinds of algorithms. All the regression algorithms that are linear based, tree based, rule based, probability based, and ensemble models are evaluated, and results are discussed.

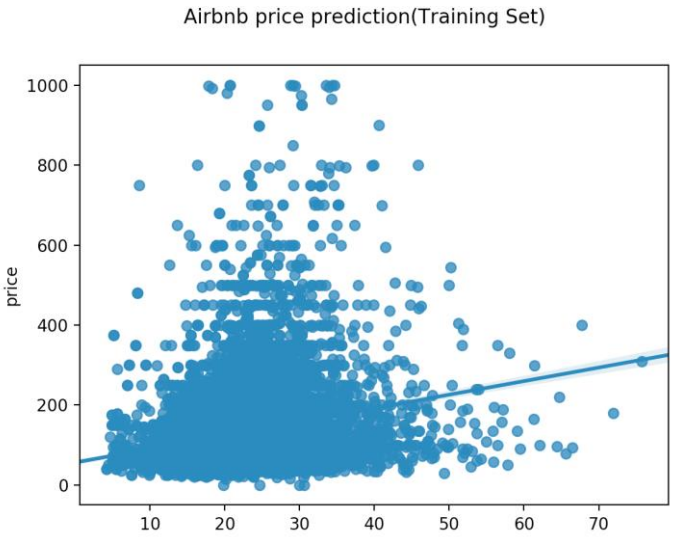


Figure 7: Training Data after pre-processing

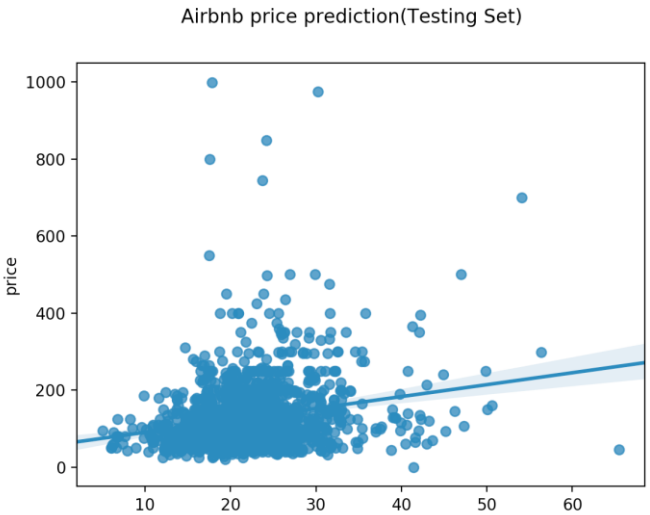


Figure 8: Testing data after pre-processing

A. *Types of model*

1. *Linear Regression Model*

It is an algorithm based on supervised learning which performs a regression task. It is commonly used for finding out the relationship between variables and predicting the target variable. It performs the task to predict a variable value that is depending(y) based on the given independent variable (x). So, we find a linear relation between x and y i.e. the input and output using regression technique. [14]

$$y = \theta_1 + \theta_2.x$$

Here y is the label for output and x for input. First theta is slope and second is the coefficient of X.
After applying Linear regression to the model, the following results are obtained. For this I have used BayesianRidge() function of Scikit learn. This will automatically take train data and test data as input and predicts output based on test data by comparing the predicted value and actual value, I calculate All the four metrics discussed previously are shown in below table 1.
Figure 9 shows the graph plotted between the sum of features and price by using Bayesian ridge regression model.

Mean Absolute Error	40.04641012942539
Mean Squared Error	4314.062100508366
Root Mean Square Error	65.68152023597172
Variance	0.46978650288747703

Table 1: Resultant metrics of Bayesian Ridge

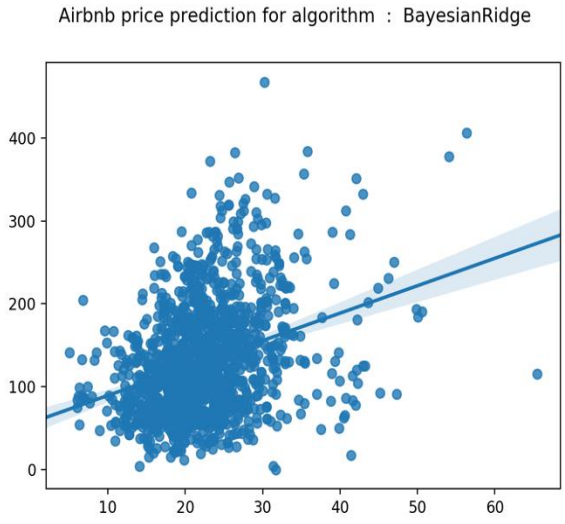


Figure 9: Data after applying Bayesian Ridge regression model

2. *Distance-based Regression Model*

K-nearest neighbors is a non-parametric method used for distance-based regression. It is one of the easiest ML technique used. It is a lazy learning model, with local approximation. In KNN regression, mean of k nearest datapoints is calculated as the output. As a rule of thumb, we select odd numbers as k. KNN is a lazy learning model where the computations happen only runtime. [15]

K nearest neighbors stores all training data and predict the target based on the distance measurement functions. There are different distance calculation methods like Euclidean, Manhattan, Minkowski. We can use these three distance measures for continuous variables. After calculating the distance, best K value is selected by first inspecting the data. Another way to determine a best value for K is by using Cross validation to an independent data set to validate the K value. Generally, K value for most datasets is greater than 10, which produces much better results when one near neighbor is taken [16].

After applying to the KNN model for data, the following results are obtained. For this I have used KNeighbourRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 2.

Figure 10 shows the graph plotted between the sum of features and price by using KNN regression model.

Mean Absolute Error	43.90425531914894
Mean Squared Error	6199.803191489362
Root Mean Square Error	78.73882899490798
Variance	0.23979494283627567

Table 2: Resultant metrics of K nearest neighbors

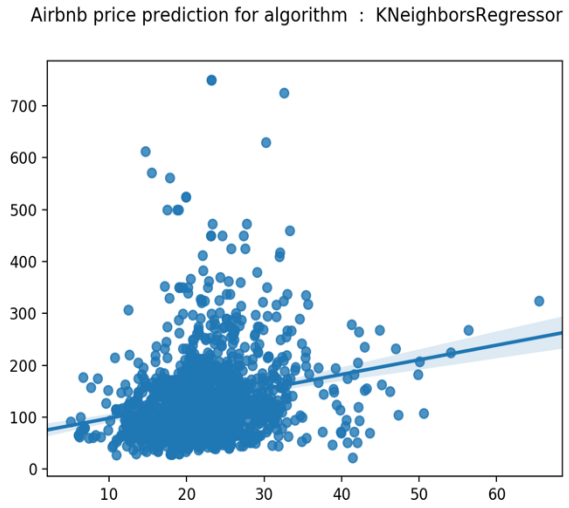


Figure 10: Data after applying KNN regression model

3. *Rule-based Regression Model*

Rule-based regression analysis methods that can identify subgroups with heterogeneous risk profiles in a population without imposing assumptions on the subgroups or method. The rules define the risk pattern of subsets of individuals by not only considering the interactions between the risk factors but also their ranges. Rule-based analysis excels at detecting multiple interactions between risk factors that characterize a subgroup [17].
After applying to the dummy regression model for data, the following results are obtained. For this I have used DummyRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 3.

Figure 11 shows the graph plotted between the sum of features and price by using Bayesian ridge regression model.

Mean Absolute Error	61.56682680137789
Mean Squared Error	8144.491208074536
Root Mean Square Error	90.24683489228049
Variance	2.220446049250313e-16

Table 3: Resultant metrics of Dummy regressor

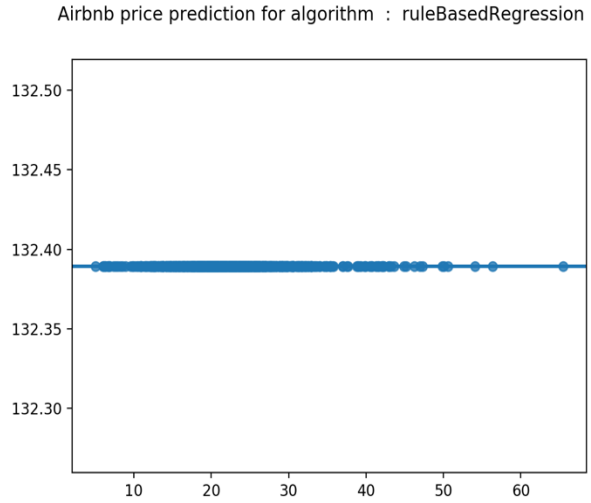


Figure 11: Data after applying Dummy regression model

4. *Decision Tree Regression Model*

The decision tree establishes a regression or classification model in the form of a tree structure, decomposing the data set into smaller and smaller subsets, and gradually develops related decision trees. The end result is a tree with decision nodes and leaf nodes [18].

The core algorithm for building decision trees called ID3 which employs a top-down, greedy search through the space of possible branches with no backtracking. By replacing information gain with standard deviation reduction, the ID3 algorithm can be used to build decision trees for regression [19].

$$SDR(T,X) = S(T) - S(T,X)$$

By using decision tree, a regression model can be built in the form of a tree structure. Many small subsets are created and at the same time a complete decision tree is developed towards the end. Finally, a tree is built with all the decision and leaf nodes. A decision node has two or more branches, each branch represents the values for the attribute that is tested. The node at the end i.e. leaf node represents a decision on the numerical target. By using this we can handle both the numerical categorical data.
After applying to the decision tree model the following results are obtained. For this I have used DecisionTreeRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 4.
Figure 12 shows the graph plotted between the sum of features and price by using Bayesian ridge regression model.

Mean Absolute Error	29.492016909478394
Mean Squared Error	4556.311813842681
Root Mean Square Error	0.4396386839651407
Variance	0.44068818388076525

Table 4: Resultant metrics of Decision Tree

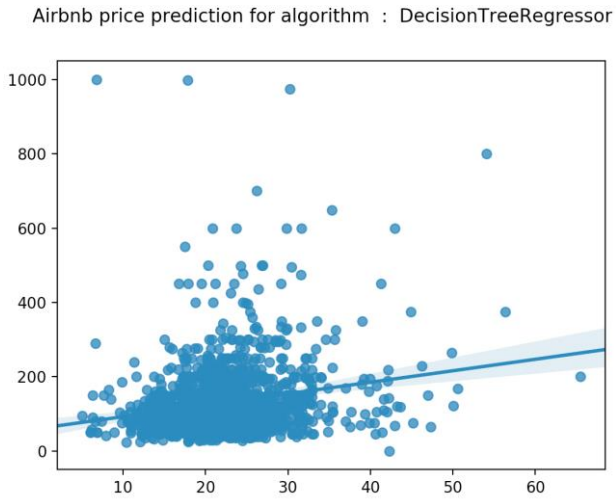


Figure 12: Data after applying Decision tress regression model

5. *Random Forest Regression Model*

Random Forest, from the name we can say that creates a forest and makes it random. The forest that is built, is a combination of Decision Trees. It is trained using the bagging method which is the combination of different models so that the overall result is increased. It has almost the same parameters as a decision tree. While growing the trees It adds some randomness to the model. It searches for the best feature among a random subset of features, instead of searching for the important feature while splitting a node, this way we can get different range of results and a better model [20].

After applying to the Random Forest model, the following results are obtained. For this I have used RandomForestRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 5.
Figure 13 shows the graph plotted between the sum of features and price by using Bayesian ridge regression model.

Mean Absolute Error	45.39736310746803
Mean Squared Error	5881.1418573140545
Root Mean Square Error	53.89975649100991
Variance	0.28228149780806155

Table 5: Resultant metrics of Random Forest

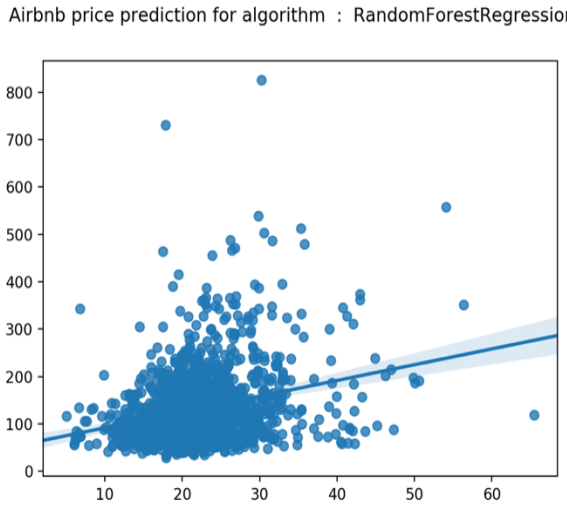


Figure 13: Data after applying random forest regression model

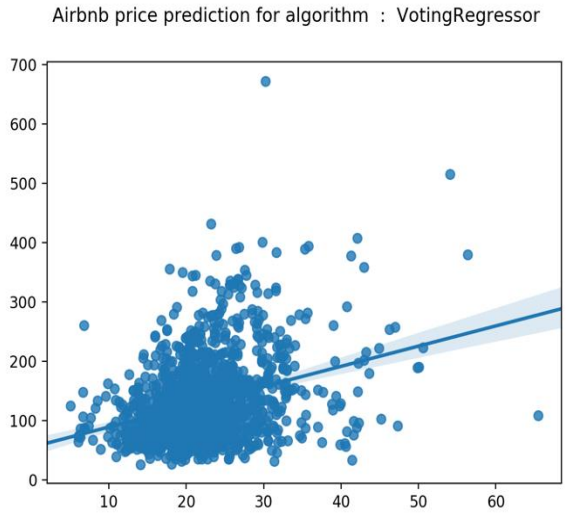


Figure 14:Data after applying ensemble regression model

6. *Ensembles*

Ensemble methods combine several trees base algorithms to construct better predictive performance than a single tree base algorithm. The main principle of the integrated model is that a group of weak learners come together to form a strong learner, thereby improving the accuracy of the model [6].
When we try to use any machine learning technique to predict the target variable, the main reasons for the difference between the actual and predicted values are noise, variance, and bias. Integration helps reduce these factors (noise is an irreducible error, but it is not the exception).
Here, we can see that if we will get the variance, noise and bias in the raw data, image or any other format of the data. Therefore, our model is either underfitting or overfitting. This reason is creating big impact on your model directly here the ensemble learning come in the picture.
Voting is one of the easiest ways to combine predictions from multiple machine learning algorithms. It first works by creating two or more independent models from the training data set. A Voting classifier can then be used to wrap your models and average the predications of the sub-models when asked to make predictions for new data.
After applying to the Ensembles model, the following results are obtained. For this I have used VotingRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 6.

Figure 14 shows the graph plotted between the sum of features and price by using Bayesian ridge regression model.

Mean Absolute Error	44.312671619957165
Mean Squared Error	5147.733090075039
Root Mean Square Error	56.693536726365146
Variance	0.3710436282440456

Table 6: Resultant metrics of ensemble

B. *Feature Selection*

Feature subset selection is the process of identifying and eliminating as many unrelated and redundant features as possible. This reduces the dimensionality of the data and allows algorithms to run faster and more efficiently.
Providing too many feature sets to the model can lead to large error variations. Therefore, many feature selection methods can be used to find the feature with the highest predictive value to reduce the model variance and reduce the computation time. [21]

From Fig. 15 and 16 training and testing data, we can see that selecting important features gives us the best accuracy for prices, and that selecting features. This project build models, run feature selection algorithm (Variance Threshold) using the best number of features, and plot the learning curves again to determine whether the overfitting problems for sum of features and prices have been reduced. For our price prediction model, when comparing the two learning curves in Fig. 14 and 15, from figures show that the train accuracy has decreased, and approaches the dev accuracy curve. In addition, we see that the dev accuracy after feature selection remains very close to 81%. Putting these together, we conclude that feature selection slightly alleviates overfitting of our price prediction model.

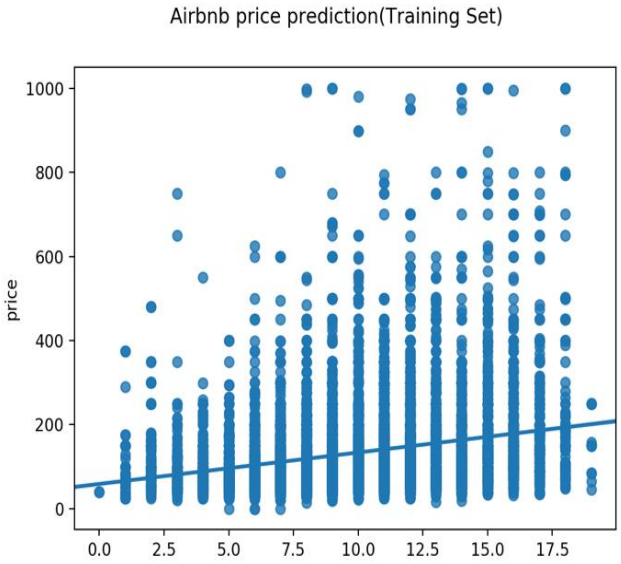


Figure 15: Training Data after applying Feature Selection

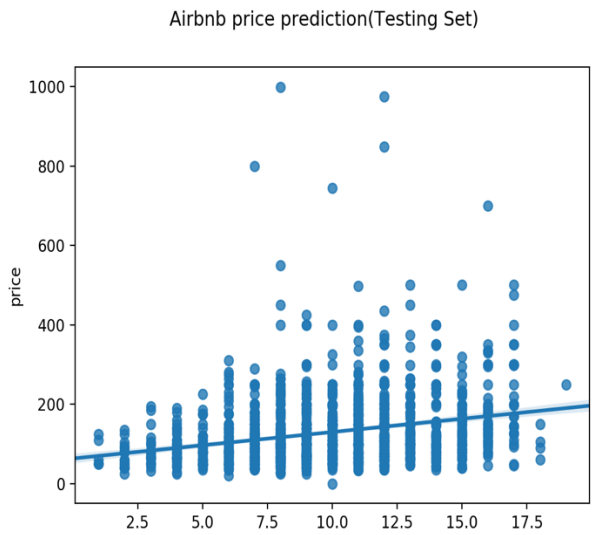


Figure 16: Testing Data after applying Feature Selection

Bayesian Ridge:

After applying feature selection for Linear regression to the model, the following results are obtained. For this I have used BayesianRidge() function of Scikit learn. This will automatically take train data and test data as input and predicts output based on test data by comparing the predicted value and actual value, I calculate All the four metrics discussed previously are shown in below table 7. Figure 17 shows the graph plotted between the sum of features and price by using Bayesian ridge regression model.

Mean Absolute Error	48.38154457738724
Mean Squared Error	5857.571178922096
Root Mean Square Error	76.5347710450753
Variance	0.2826595897657912

Table 7: Resultant metrics of Bayesian Ridge

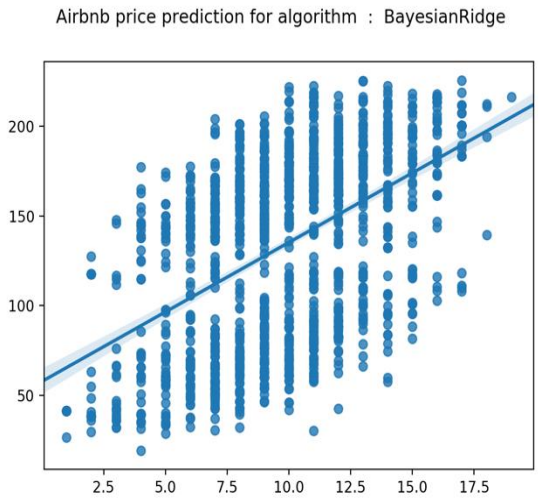


Figure 17: Data after applying Feature Selection to Bayesian Ridge regression model

KNN:

After applying feature selection to the KNN model for data, the following results are obtained. For this I have used KNeighbourRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 8. Figure 18 shows the graph plotted between the sum of features and price by using KNN regression model.

Mean Absolute Error	48.29090242112986
Mean Squared Error	6628.248165810712
Root Mean Square Error	81.41405385933507
Variance	0.18565179959965117

Table 8: Resultant metrics of K nearest neighbor

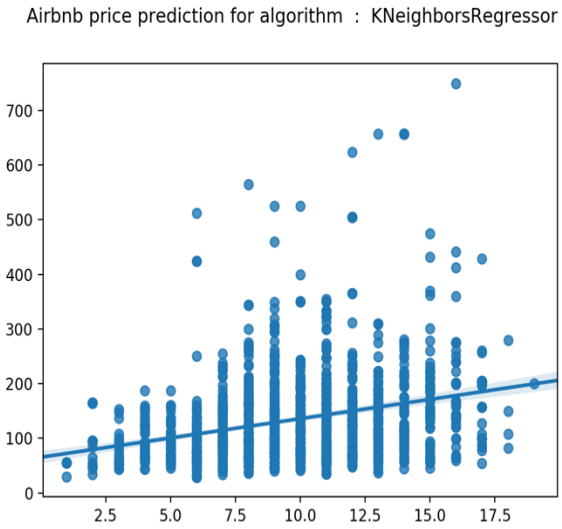


Figure 18: Data after applying Feature Selection to KNN regression model

Dummy:

After applying feature selection to the dummy regression model for data, the following results are obtained. For this I have used DummyRegressor () function of Scikit learn. All the four metrics discussed previously are shown in below table 9. Figure 19 shows the graph plotted between the sum of features and price by using dummy regressor model.

Mean Absolute Error	61.56682680137789
Mean Squared Error	8144.491208074536
Root Mean Square Error	90.24683489228049
Variance	2.220446049250313e-16

Table 9: Resultant metrics of Dummy regressor

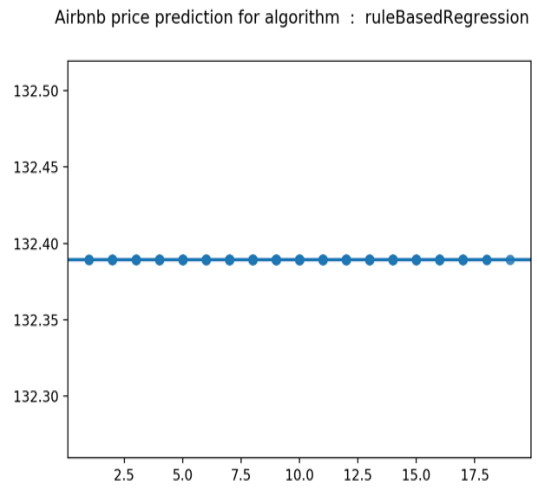


Figure 19 : Data after applying Feature Selection to Dummy regression model

Decision Tree:

After applying feature selection to the decision tree model the following results are obtained. For this I have used DecisionTreeRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 10. Figure 20 shows the graph plotted between the sum of features and price by using decision tree model.

Mean Absolute Error	44.870302297641274
Mean Squared Error	7174.132533234459
Root Mean Square Error	84.70025108129526
Variance	0.11933179959277296

Table 10: Resultant metrics of Decision tree

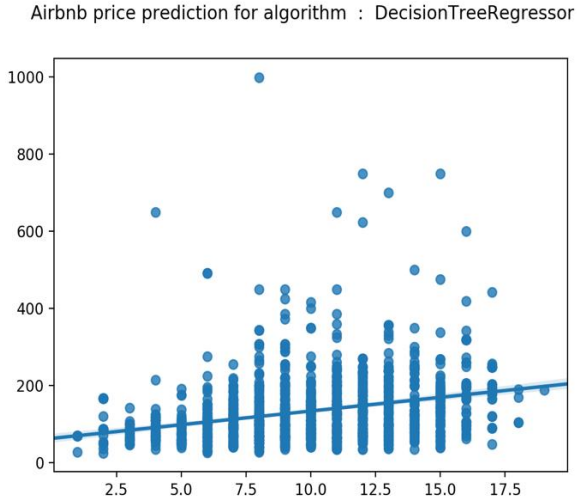


Figure 20: Data after applying Feature Selection to Decision Tree regression model

Random Forest:

After applying feature selection to the Random Forest model, the following results are obtained. For this I have used RandomForestRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 11. Figure 21 shows the graph plotted between the sum of features and price by using random forest regression model.

Mean Absolute Error	44.82383556935384
Mean Squared Error	5660.71785388088
Root Mean Square Error	74.97263297676216
Variance	0.30871522523397066

Table 11: Resultant metrics of Random forest

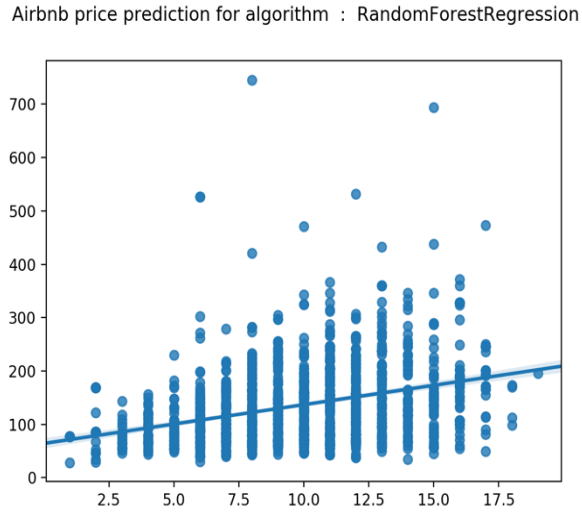


Figure 21 : Data after applying Feature Selection to Random Forest regression model

Voting:

After applying feature selection to the Ensembles model, the following results are obtained. For this I have used VotingRegressor() function of Scikit learn. All the four metrics discussed previously are shown in below table 12. Figure 22 shows the graph plotted between the sum of features and price by using Voting ensembles model.

Mean Absolute Error	44.30424483031248
Mean Squared Error	5147.361531674357
Root Mean Square Error	71.74511503701389
Variance	0.37108342498975844

Table 12: Resultant metrics of Voting ensembles

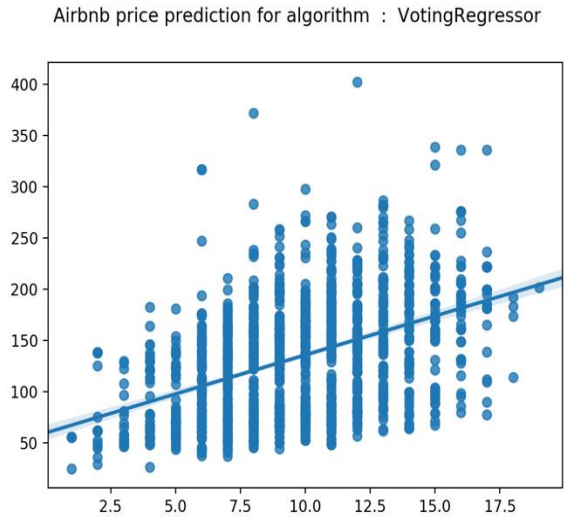


Figure 22: Data after applying Feature Selection to Voting

C. *Experimental Results*

The model is fitted on the train set, and it is evaluated on the test set using metrics such as the Root Mean Squared Error (RMSE), the Mean Absolute Error (MAE), or the Median Absolute Error (MedAE), Mean Square Error (MSE) and Variance. This process is repeated until all model's values are calculated. The average error over the k test sets provides an estimation of the model's prediction error on unseen data. In total, I have considered 6 models containing different combinations of different regression techniques like linear based, tree based, distance based, rule based and ensembles. I performed calculations, that is, repeated for different samples of train and test data. In each model, I calculated four metrics, specifically the RMSE, MAE, MSE and Variance.

This project evaluates the models based on the metrics discussed in the previous section with the raw data i.e. before feature selection and after feature selection.

Below Table 13 shows the evaluation metrics like MAE, MSE, RSME and variance of the models before the feature selection is applied and Table 14 shows the evaluation metrics after the feature selection is done on the raw data. In feature selection some of the features are removed, only the important features are considered, and I have used Variance Threshold function from Scikit Learn which minimizes the features and applied on the testing and training data.

Model	MAE	MSE	RSME	Variance
Bayesian Ridge	40.046	4314.062	65.681	0.469
KNN	43.904	6199.803	78.738	0.239
Dummy	61.566	8144.491	90.246	2.220
Decision tree	28.967	4595.090	67.681	0.435
Random Forest	29.251	2908.80	53.899	0.643
Voting Ensembles	33.203	3214.157	56.693	0.605

Table 13: Model selection metrics before feature selection (Raw data)

Model	MAE	MSE	RSME	Variance
Bayesian Ridge	48.381	5857.571	76.534	0.282
KNN	48.290	6628.248	81.414	0.185
Dummy	61.566	8144.491	90.246	2.228
Decision tree	44.870	7174.132	84.700	0.117
Random Forest	44.823	5660.71	74.972	0.308
Voting Ensembles	44.304	5147.733	71.745	0.371

Table 14: Model selection metrics after feature selection

After building all the models and calculating the metrics we can see that Mean absolute error (MAE) for Voting Ensembles is very less with the raw data and Decision Tree when the feature selection is done. When we consider Variance, the variance is lowest for the KNN regressor for raw data and when the feature selection is done Decision Tree is having less variance. From above table we can also see that Lasso also has a good prediction rate. If we see with Root Mean Square Error (RSME), it is less for Random forest model on raw data and Voting Ensembles model for feature selection. This project concludes that Decision Tree model is best for predicting price for listing for raw data and Voting Ensemble model after feature selection.

D. Lessons Learned

I have learned about the different machine learning models and how they are used for predicting the optimal prices. I have also learned different techniques for pre-processing the data and make a clean data which gives best accuracy and run the models efficiently.

VI. FUTURE WORKS

The future works on this study can include studying other feature selection schemes for improving the accuracy for the model for predicting the price and also further experimentation with different other machine learning regression models for predicting the optimal price.

VII. CONCLUSION

This paper seeks to estimate Airbnb pricing based on a limited number of properties, including property characteristics, owner information on the list. Machine learning techniques (including linear regression, tree-based models, distance-based, rule- and ensemble-based, and feature selection analysis) are used to achieve the best results in terms of mean square error, mean error, root mean square error and variance. Initial experiments of the baseline model demonstrate that great features lead to greater variability and weaker performance of the model in the validation set compared to the training set.

Additionally, it helps us explore the initial question by considering how Airbnb's listing prices are in the hands of landlords, whether there is a correlation between listing prices, or whether it is purely incidental. These models not only allow me to estimate the listing price of Airbnb, but also allow me to delve deeper into the dataset and to draw trends that I had not thought of.

VIII. REFERENCES

[1]. https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4

[2]. Kotsiantis, S. B., Kanellopoulos, D., & Pintelas, P. E. (2006). Data preprocessing for supervised leaning. *International Journal of Computer Science*, *1*(2), 111-117.

[3].https://sebastianraschka.com/Articles/2014_about_feature_scaling.html

[4]. https://towardsdatascience.com/dimensionality-reduction-for-machine-learning-80a46c2ebb7e

[5].https://www.analyticsvidhya.com/blog/2015/07/dimension-reduction-methods/

[6]. https://towardsdatascience.com/ensemble-learning-in-machine-learning-getting-started-4ed85eb38e00

[7]. https://towardsdatascience.com/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222

[8].https://github.com/micts/airbnb-price-prediction/blob/master/Predictive%20Analysis%20of%20Price%20on%20Amsterdam%20Airbnb%20Listings%20Using%20Ordinary%20Least%20Squares.pdf

[9]. https://towardsdatascience.com/introduction-to-bayesian-linear-regression-e66e60791ea7

[10].https://towardsdatascience.com/handling-missing-values-in-machine-learning-part-1-dda69d4f88ca

[11]. https://codeburst.io/2-important-statistics-terms-you-need-to-know-in-data-science-skewness-and-kurtosis-388fef94eeaa

[12]. https://en.wikipedia.org/wiki/Skew

[13]. https://www.sciencedirect.com/topics/nursing-and-health-professions/correlation-analysis

[14]. https://www.geeksforgeeks.org/ml-linear-regression/

[15]. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[16]. https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/

[17]. Haghighi, M., Johnson, S. B., Qian, X., Lynch, K. F., Vehik, K., Huang, S., ... & Felipe-Morales, D. (2016). A comparison of rule-based analysis with regression methods in understanding the risk factors for study withdrawal in a pediatric study. *Scientific reports*, *6*, 30828.

[18].https://medium.com/@chiragsehra42/decision-trees-explained-easily-28f23241248

[19].https://medium.com/@rishabhjain_22692/decision-trees-it-begins-here-93ff54ef134

[20]. https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[21].https://machinelearningmastery.com/an-introduction-to-feature-selection/

[22]. https://scikit-learn.org/stable/

[23]. https://pandas.pydata.org/

[24]. https://www.anaconda.com/

[24].https://medium.com/@TheDataGyan/day-8-data-transformation-skewness-normalization-and-much-more-4c144d370e55

[26].http://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/BS704_Multivariable/BS704_Multivariable5.html