Airbnb Price Prediction using Pyspark

Term Project Report

Submitted By :

Kaamya Ravikumar - 015400138

Akshaya Srinivasan - 015967406

Yash Jobanputra - 015383862

Department of Applied Data Science, San Jose State University, San Jose

DATA 228 - Big Data Technologies and Applications

Professor Dr Ming-Hwa Wang

Contents

Acknowledgement

We would like to take this opportunity to express our gratitude to Professor Dr. Ming-Hwa Wang. His teaching on the Big Data Technologies & Applications topics have been extremely helpful toward the completion of this project. We would also like to thank him for being encouraging and supportive whenever we encounter a problem during the class. In addition, we would like to thank all the contributors mentioned in the bibliography. Their works were vital for the success of this project. Lastly, we thank all the team members for dedicatedly pushing this project into completion.

Abstract

Estimating the right pricing range for the property rentals on the Airbnb website is a key factor to attract the numerous tourists and travellers. With more hosts getting added to the airbnb network and more exquisite rental properties on the market, pricing the rentals has become a challenging task for every owner. There could be any number of factors that influence the price for a rental including the size of the property, its distance from the sightseeing places in that city, seasonal charges like Christmas or New year, the neighbourhood in which it is located, extra amenities provided by the owners like parking space, swimming pool etc. Our research project will aim to identify and extract the most likely features from the selected dataset to develop a predictive model that would predict the optimal price for a rental property in a dynamic way. The source of data for this project would be multiple csv files from the Airbnb website. The website consists of listings in different cities across the world and we would be choosing 3 csv files for our project. We would be doing an end to end exploratory data analysis to understand the data and the features, perform feature engineering and extract the necessary features to develop a high performance predictive model to determine the price and present a comparative study on different machine learning models on this dataset.

Introduction

Airbnb is a service that allows owners to rent out their own spaces for tourists. People can rent out private rooms for individual guests, spaces for multiple people to share or sometimes even the entire property. In recent times Airbnb has become popular as not many could afford to stay in a hotel and sometimes it can even be difficult to find a hotel room in a busy area. Since the company commenced in 2008, over 100 million people have booked rentals through Airbnb. A host accurately describes the property including its location, price, property description including amenities and features that can help match their space with the right guests. These become the listings on the airbnb website that allows guests to select the rental that suits their needs.

Objective

Price prediction models are critical components in the business models of online marketplaces in today's world. Since the main role of these models would be to discover the prices and enable smooth transactions between buyers and sellers, organizations like Airbnb would gain high benefits from these models to improve their performance and guide the hosts who present the listings on their page. The main objective behind this project would be to develop a model that would help hosts decide on the pricings for their properties dynamically and guarantee increased accessibility on the website.

The end deliverable for this project would be a statistical model that would be built by analyzing over 0.3 GB of data collected from various sources. This goal would be to provide an understanding to the owners and tourists about those features that would be most important in determining the price and an optimal price that should be quoted for a property. We would present a comparative study of several regression models that would be developed by us for this

dataset. In the process we would be performing end to end exploratory data analysis of the dataset, perform data cleaning and wrangling, data modelling and present the final study from a storytelling perspective using different visualizations.

Problem Statement

The price for each rental listing on the airbnb website is a decision that is completely made by the owners. They are allowed to compare the prices of other listings in the same area and determine their choice and are even allowed to charge extra cost for any added amenities that they might provide within their property to the guests. As the traffic on Airbnb in terms of hosts and guests is seeing an upward trend in recent times quoting the right price for a property is of primary importance. A solid machine learning model that would aid this purpose would be an apt solution to this.

The hosts lack a reliable thumb rule of what would be an optimum price for their property keeping in mind different factors including location, distance and seasonal impact. A wrong price quote would correspond to loss of guests who visit and stay at their property. Likewise a thumb rule needs to be put in place for the guests as well who need to understand whether the price they see for a specific property on Airbnb is worth the property or not. To address this a dependable model that would be beneficial to both the owners and tourists would be delivered at the end of this project.

Project Relevance

For this project, we would be sourcing data from the Airbnb website for multiple locations to build the prediction model. The overall size of the dataset is expected to be very large and this makes it a right choice for a Big data analytics project. We would be implementing the data pre-processing, feature engineering and model development using Apache Spark

libraries and pyspark which is the python API for Spark, Spark SQL would be the base programming languages used to implement this. This makes it a very relevant project to be executed as part of this course.

Existing Approaches and why this is a better approach

We have referenced a lot of existing literature reviews for this project which we would be explaining in the upcoming sections. The authors who have done research on this topic in the past have come up with several models that were best suited for their dataset. With this project we aim to provide a comparative study of different models for predicting the price and provide a consolidated report on why a particular model would fare better and why another model would not which was not identified in any paper that we had referred previously.

Area of Scope and Investigation

The availability of airbnb rentals during peak holiday seasons has always been a concern for tourists who travel. For our future work we would work to build a model that would take the different features of an airbnb rental to predict its availability. Also depending on the output accuracy of our model, we will work to implement more advanced feature selection techniques like the Principle Component Analysis based feature selection techniques to improve the efficiency.

We plan to take into consideration the impact every season would have on the pricing of an airbnb property. For eg: the price of an airbnb would be at the highest during long weekends, christmas , new year or thanksgiving and would not be so heavily priced during off-seasonal months like August or September. We would like to source our data keeping in mind this particular goal as our existing dataset does not have this information.

We would also like to work on a model that would predict a specific range in which the pricing of a rental would fall into instead of predicting the specific value of a rental price. Doing this would give the hosts a more generalized idea to quote a price for their property.

Theoretical Basis and Literature Review

Definition of Problem

As the traffic on Airbnb in terms of hosts and guests is seeing an upward trend in recent times quoting the right price for a property is of primary importance. A solid machine learning model that would aid this purpose would be an apt solution to this.

The hosts lack a reliable thumb rule of what would be an optimum price for their property keeping in mind different factors including location, distance and seasonal impact. A wrong price quote would correspond to loss of guests who visit and stay at their property. Likewise a thumb rule needs to be put in place for the guests as well who need to understand whether the price they see for a specific property on Airbnb is worth the property or not. To address this a dependable model that would be beneficial to both the owners and tourists would be delivered at the end of this project.

Related Research

Several authors have contributed to the study of developing a price prediction model in the past. (Hoormazd et al. 2020) have conducted a study using the New York airbnb listings to come up with a model for predicting the price using the algorithms such as Ridge Regression, Support Vector Regression,etc. Price being a continuous variable can be modelled only with regression models and not classification. The biggest insight that was gained from this paper was the usage of LASSO regularization for feature selection. The baseline model that was

developed with the features that were selected manually was used for comparison against the models that were developed using features selected from the LASSO regularization technique. The mathematical representation of this technique is given by

Residual Sum of Squares + λ * (Sum of the absolute value of the magnitude of coefficients)

where λ => denotes amount of shrinkage

λ=0 implies all features are considered and it is equal to linear regression where only the residual sum of squares are considered to build the model

$\lambda=\infty$ implies no feature is considered. This shows that as λ moves closer to infinity more and more features are eliminated.

Bias increases with increase in λ

Variance increases with decrease in λ

The resulting comparison showed the increased efficiency in terms of evaluation metrics like decreased MSE and MAE and high value of R^2 of the models. According to their research, the baseline Linear Regression model that was developed using manual feature selection performed the least with a R^2 score of $-5.1*E^{-13}$ and MAE and MSE scores of 96895.82 and $2.4*E^{13}$. The models that were developed after applying LASSO regression technique performed far better compared to the baseline model. Amongst those, the SVR model performed the best with a very high R^2 score of 69% and very less MAE and MSE error percentage of 0.27 and 0.14 whose success could be attributed to the usage of the Kernel Trick which involved performing computations on non-linear data in the same dimension without having to convert to a higher dimensional space. In our project, we would build the regression model using algorithms that have not been discussed in this work and provide a comparison of those.

The authors of another paper we studied, (Dhillon et al. 2021), have done an analysis and price prediction of Airbnb listings for the cities in the United States. The paper covers processes such as outlier handling, feature selection based on insights from descriptive, prescriptive, and exploratory analysis, and Price prediction using multiple machine learning algorithms. The objective of the paper was to make optimal predictions about the prices of Airbnb housing by identifying significant features like location, minimum and maximum numbers of bookings per month, etc. and training the prediction model based on these features. The outcome was that the customers would be able to afford their preferred accommodation and the hosts would be able to increase their occupancy factor with comparative pricing. The data considered was 305564 rows x 108 columns. Data cleaning was performed by removing duplicates, nulls, and unnecessary variables. For identification and removal of outliers, the unsupervised machine learning algorithm called One Class Support Vector Machine (SVM) was used. This algorithm creates a decision boundary based on the given data. Anything that lies outside of that boundary is classified as an outlier and is excluded from the data. By detecting the outliers in the continuous target variable price, they have aimed to increase the prediction accuracy of models that were to be built. After data cleaning, exploratory data analysis was performed with the aim of understanding the data and selecting the required features for the model. Various visualizations depicting the univariate and bivariate distributions of features and the impact of each feature on Price have been constructed. A few examples include distribution of house type of the listings, number of listings over the years, patterns of price change over time, overall ratings of the listings and correlation plots of numeric features. After feature selection, prediction has been done using Linear Regression, Logistic Regression and Random Forest algorithms. In linear regression, the price of Airbnb listings is predicted using selected features as independent

variables. The mean square error was calculated to check how accurate the model is for predicting the prices based on certain features. For Logistic Regression, the logistic equation obtained from the training model and the probability of the price is determined using the equation. The accuracy of the logistic regression classifier on the test set was 0.78. In the Random Forest algorithm, features such as minimum nights, maximum nights, availability 365, number of reviews, review scores rating and review scores accuracy were used for the price prediction and an accuracy of 87% was achieved. The performance of the 3 models were compared and Random Forest was identified as the best prediction model for this case.

(Sainaghi,2020) mentioned in his research that some features that could impact the rates such as host characteristics, location, customer reviews, destination characteristics and external comparison. (Voltes-Dorta et al.,2020) claimed that previous studies on rental prices on Airbnb have focused on location, room and host characteristics to determine the rental price. Their approach is to consider these same factors across property types, locations, and seasons using ordinary least squares and geographically weighted regression methods. (Kalehbasti et al., 2019) provided the information on finding the best price prediction model by selecting the most relevant features using regression algorithms. The results of the study led to a satisfactory accuracy of the proposed model. (Akarsu et al., 2020) showed that the authors try to determine the importance of the information disclosed about the deals listed on Airbnb towards the consumers' purchasing behavior. Their results suggest that recommendations published on the platform have a significant impact on purchase intentions. In the last research, (Xu et al., 2021) applied social exchange theory to investigate how the Airbnb platform influences experience and authenticity. They highlighted the importance of the platform and its features in enhancing the attractiveness of the service. In spite of all the research that has happened in the past, the authors

of the current presenting paper did not find any approach based on the segmentation of customers into categories to better identify these factors. The methodology they followed consisted of a large collection of reviews and rating scores posted by customers on the Airbnb platform through the Inside Airbnb tool. The data is of London Homes from December 2009 and April 2020. The collected data contains 1,513,966 reviews left by customers and covers thousands of accommodations listed on the platform. They cleaned the missing and unnecessary data and the remaining data was approx 100000. One of the major problems they faced was that the data collected did not provide any indication of the customer category. Therefore, they used algorithms derived from Natural Language Processing (NLP) to segment the opinions according to the categories. Using syntactic dependency parsing, which highlights the syntactic structure of a sentence as shown in this figure to detect the relationships that its elements have, they identified the opinions where the subjects refer to the individual person. This is how they had categories i.e. 38,543 individuals, 5,495 couples, and 10,874 family reviews. Next step was to calculate the coefficients. They had used two regression algorithms based on weights to calculate the coefficients applied to each of the 6 elementary indicators rated by the customer on Airbnb. Regression algorithm is a supervised learning model adapted to quantitative data. It is a method of modeling a target variable according to independent features. It is used when the variables are continuous. The authors identified the degree of importance of the 6 elementary indicators like accuracy, cleanliness, check-in, communication, location and value. These all summarize the level of satisfaction. They have expressed the global score (GS) as a function of the elementary scores using the following formula:

$$GS = \beta 1\ AC + \beta 2\ CL + \beta 3\ CH + \beta 4\ CO + \beta 5\ LO + \beta 6\ VA$$

Where, AC, CL, CH, CO, LO, and VA stand for accuracy, cleanliness, check-in, communication, location, and value, respectively, and β1, β2, β3, β4, β5, and β6 are the coefficients acts on elemental scores. After calculating the global scoring, they have started training the machine learning model. They have trained both MLR and GBR algorithms on the segmented data where they have differentiated 38,543 reviews for individuals, 5,495 reviews for couples, and 10,874 reviews for families by dividing them into two batches, like 80% for the train set and 20% for the test set. The inputs of the model are the AC, CL, CH, CO, LO and VA features and its output is the GS target. After training the two models they calculated the coefficient of determination, which represented a measure of the quality of the prediction of a linear regression, whose values were 0.660, 0.684, 0.712 respectively for the individuals, couples and family using the Multiple Linear Regression model and 0.70, 0.71 and 0.74 with the Gradient Boosting Regression model. The coefficients calculated for accuracy using the GBR algorithm is clearly the largest compared to the MLR But overall, both algorithms give fairly consistent results. After exploring these results, they concluded that customers' expectations of P2P hosting, in this case at Airbnb, are relatively different. Families were more focused on accuracy of the description written by the host, where the couples were more inclined towards cleanliness, value, and accuracy. On the other side the individuals were interested in the communication from the host side. There were some factors which the other authors did not consider like age, gender, cultural and geographical dimensions, amenities. As per the authors, these factors could also impact the overall score of the model.

Advantage of Existing Research

Based on the papers we have referenced, one paper has given extensive emphasis on outlier detection and visualizations while another paper has used sophisticated feature selection

techniques such as LASSO regression to shortlist the features required to build the model. The other paper focuses on categorizing the customers into three different groups i.e. (Individuals, Couples and family) so that one can prioritize the preferences to rent the accommodation as per their requirements.

Disadvantage of Existing Research

In one of the papers, a limited set of features were considered in developing the predictive model. Though the authors were able to provide a comparative study of the different regression algorithms with the existing features, including more features would change the way the final model would perform. In another paper, the information about implementation of models and their specifics were limited. The details about tools and technologies used for data sourcing and implementation were unknown. Also, the specific reason behind choosing a particular model such as Logistic Regression over other regression models for prediction of a qualitative target is unknown.

Our Solution to the Problem

In our project, we aim to include a wider range of features for the data analysis and feature selection. Also, we will be considering data of two different cities, and do a comparative study taking locality into consideration. We will perform effective feature selection using a combination of wrapper and filter methods. Additionally, we will include details of the data source, tools used, data wrangling performed and reasoning behind the implementation of specific models in our report.

How is our Solution Different from others

Apart from taking inspiration from the papers that we have referenced to develop our models, for our project, we would be using datasets with more features than used in the original

papers. This would the final performance of the model and requires advanced feature selection techniques

How is our Solution better

In our proposed solution, as we would be incorporating a wide range of features to build the final model, the model thus developed would be more generalized in terms of the learning and that would in turn have a positive impact on the final accuracy and model efficiency.

Hypothesis

The goal of our project is to effectively predict the prices of Airbnb listings and do a comparative study on the prices in San Francisco and San Diego. In this section, we present different hypotheses that help us understand the significance of various features on our target variable Price. Rejecting or failing to reject each of the hypotheses will take us one step closer to the goal of efficient price prediction. Removing insignificant attributes from further analysis will reduce the noise introduced to the model, thereby increasing its accuracy. The Airbnb dataset we have considered includes information about listings across 2 cities – San Francisco and San Diego.

We will be focusing on testing each hypothesis with (Frost et al., 2021) a parametric and/or a non-parametric test, depending on the distribution of the attributes considered. Parametric hypothesis tests such as t-test and ANOVA assume that the test attributes are normally distributed and compare means of the population. Non-parametric tests such as Mann-Whitney and Kruskal-Wallis tests compare medians of the population and hence work effectively with abnormal distributions as well. We will be using parametric tests wherever applicable as they have more statistical significance when compared to non-parametric tests. For testing the relationship between two continuous variables, we'll be using a two-sample t-test

and/or Mann-Whitney U test. For the tests between a continuous and a categorical variable, we'll be using two-way ANOVA test and/or Kruskal-Wallis test.

In this project, we will first be performing exploratory data analysis to understand the relationship between two or more attributes. We will then compare these results with those obtained from hypothesis tests to check if the relationship is statistically significant. From the comparison, only the attributes that show strong impact on the target variable, Price will be considered for feature selection and dimensionality reduction for predictive modeling.

Based on the dataset considered, we aim to test the following hypotheses in this analysis:

Hypothesis 1

Goal: Identify effects of seasonality and cyclicity on Price of listings and take that into consideration while building the model, to arrive at a normalized baseline for prediction.

Question 1: Does the price of a listing increase or decrease during specific times of every year? Perhaps during holidays, festive or vacation times?

Null Hypothesis(H_0): The price of a listing remains constant throughout all the seasons and festivals.

Hypothesis 2

Goal: Identify the effect of a host having superhost tag on the price of listings with similar attributes

Question: Do listings with the same attributes such as room type, amenities, etc. have different prices based on whether they are a superhost?

Null Hypothesis(H_0): The price of Airbnb listings with similar features do not differ significantly based on the superhost tag.

Hypothesis 3

Goal: To understand the importance of ratings and reviews on the price of listings

Question: Are listings with better reviews and ratings priced higher than the others?

Null Hypothesis(H_0): The number of reviews and ratings do not have a considerable impact on the price of listings

Hypothesis 4

Goal: To deduce if the pricing of listings with instant bookable feature vary significantly as compared to those which do not have the feature

Question: Are instantly bookable listings priced lower/higher than other listings with similar features?

Null Hypothesis(H_0): Price of listings do not depend on the ability to instantly book the accommodation

Hypothesis 5

Goal: Discover if host profile attributes such as]response time, verification, and number of listings controlled by a host have an impact on price

Question: Are the listings of hosts with better profile attributes (verified, multiple listings, etc.) priced at a higher rate than others?

Null Hypothesis(H_0): Host profile attributes do not have a significant impact on the price of listings

Hypothesis 6

Goal: Determine if host controlled features have more control over price of listings than features that can't be controlled by hosts

Question: Is the impact of features such as amenities, minimum and maximum nights, etc. on price more than features such as reviews per month, count of neighborhood listings, etc?

Null Hypothesis(H_0): Host controlled attributes such as amenities, minimum and maximum nights, booking features do not have more significance on price than the features that cannot be controlled by the host in each neighborhood

Methodology

We will be using the following methodology as our approach for effective price prediction of Airbnb listings.

Figure 1

Analysis Methodology

Note. Steps involved in price prediction of Airbnb listings

Data Collection

Airbnb has kept its data publicly available and is not copyrighted. The website that we have collected the data from is http://insideairbnb.com. One can collect data from any region where Airbnb is available. The advantage of collecting the data from Inside Airbnb website is the data they are listing on the website is we always get cleaned, accumulated and examined data. Also they have made sure that no private information is being shared of any of their customers. Airbnb has intentionally kept some of the reviews as spam and they have claimed that these spam reviews do not affect the statistical analysis of the data.

Figure 2

listings.csv

host_id	host_url	host_name	host_since	host_location	host_response_time	host_response_rate	host_acceptance_rate	host_is_superhost	host_thumbnail_url	host_picture_url	host_neighbourhood	host_listings_count
615	https://ww	Joe	2008-07-07	Denver, Colorad	within an hour	100%	100%	t	https://a0.muscache.c	https://a0.muscac	Virginia Village	2
666	https://ww	Jennifer & Gi	2008-07-08	Denver, Colorad	within an hour	100%	93%	t	https://a0.muscache.c	https://a0.muscac	Highland	2
783	https://ww	Jason	2008-07-11	Denver, Colorad	N/A	N/A	N/A	f	https://a0.muscache.c	https://a0.muscac	Five Points	1
933	https://ww	Jill	2008-07-21	Denver, Colorad	within a few hours	100%	95%	t	https://a0.muscache.c	https://a0.muscac	North Park Hill	2
933	https://ww	Jill	2008-07-21	Denver, Colorad	within a few hours	100%	95%	t	https://a0.muscache.c	https://a0.muscac	North Park Hill	2
990	https://ww	Alexandra	2008-07-23	Denver, Colorad	within a few hours	100%	50%	f	https://a0.muscache.c	https://a0.muscac	North Capitol Hill	1
2150	https://ww	Joanne	2008-08-16	Denver, Colorad	N/A	N/A	N/A	t	https://a0.muscache.c	https://a0.muscac	Baker	3
135298	https://ww	Rick	2010-05-30	Denver, Colorad	within an hour	100%	100%	t	https://a0.muscache.c	https://a0.muscac	West Highland	1
666	https://ww	Jennifer & Gi	2008-07-08	Denver, Colorad	within an hour	100%	93%	t	https://a0.muscache.c	https://a0.muscac	Highland	2
385864	https://ww	Joe	2011-02-10	Denver, Colorad	within an hour	100%	50%	t	https://a0.muscache.c	https://a0.muscac	Capitol Hill	1
489506	https://ww	Doreen	2011-04-06	Denver, Colorad	within an hour	100%	100%	t	https://a0.muscache.c	https://a0.muscac	North Park Hill	4
745200	https://ww	Susan	2011-06-26	Denver, Colorad	within a few hours	100%	25%	f	https://a0.muscache.c	https://a0.muscac	Ballpark	9
526354	https://ww	Michael	2011-04-21	Denver, Colorad	within a day	67%	20%	f	https://a0.muscache.c	https://a0.muscac	Washington Park	0
923646	https://ww	Kate	2011-08-04	Denver, Colorad	within a few hours	100%	92%	t	https://a0.muscache.c	https://a0.muscac	South Park Hill	1
10730	https://ww	William	2009-03-19	Westminster, Col	within an hour	100%	100%	f	https://a0.muscache.c	https://a0.muscac	Highland	1
1238201	https://ww	Dave	2011-10-02	Denver, Colorad	N/A	N/A	50%	f	https://a0.muscache.c	https://a0.muscac	LoDo	1
1475137	https://ww	Lee	2011-12-05	Portland, Oregon	within an hour	100%	77%	t	https://a0.muscache.c	https://a0.muscac	LoDo	2
1154806	https://ww	Ellen	2011-09-13	Denver, Colorad	within an hour	100%	96%	t	https://a0.muscache.c	https://a0.muscac	Five Points	1
930851	https://ww	Nicholas	2011-08-06	Denver, Colorad	within a day	100%	86%	f	https://a0.muscache.c	https://a0.muscac	Washington Park West	1
269892	https://ww	Maura	2010-10-24	Denver, Colorad	within a day	100%	78%	t	https://a0.muscache.c	https://a0.muscac	Baker	1
10951	https://ww	Tricia	2009-03-21	Denver, Colorad	within a few hours	100%	83%	t	https://a0.muscache.c	https://a0.muscac	City Park West	1
2370964	https://ww	Andrea	2012-05-14	Denver, Colorad	within an hour	100%	93%	f	https://a0.muscache.c	https://a0.muscac	Clayton	1
2563107	https://ww	Matthew	2012-06-06	Denver, Colorad	within an hour	100%	100%	t	https://a0.muscache.c	https://a0.muscac	City Park West	1

Note: A sample listings.csv raw dataset consisting of different variables

Figure 3

listings.csv

amenities	price	minimum_nights	maximum_nights	minimum_minimum_nights	maximum_minimum_nights	minimum_maximum_nights	maximum_maximum_nights	minimum_nights_avg_ntm	maximum_nights_avg_ntm
["Shampoo"	$99.00	1	400	1	2	1125	1125	1.4	1125.0
["Room-dar	$135.00	29	90	29	29	90	90	29.0	90.0
["Shampoo"	$179.00	185	365	185	185	365	365	185.0	365.0
["Shampoo"	$54.00	2	300	1	13	22	300	2.7	299.4
["Shampoo"	$52.00	1	365	1	21	365	365	2.3	365.0
["Shampoo"	$33.00	30	180	30	30	180	180	30.0	180.0
["Shampoo"	$99.00	1	120	1	1	1125	1125	1.0	1125.0
["Shampoo"	$85.00	4	730	3	4	1125	1125	4.0	1125.0
["Shampoo"	$110.00	29	90	29	29	90	90	29.0	90.0
["Shampoo"	$135.00	5	1124	5	10	1124	1124	7.9	1124.0
["Shampoo"	$90.00	1	1125	1	1	1125	1125	1.0	1125.0
["Room-dar	$67.00	90	1125	90	90	1125	1125	90.0	1125.0
["Shampoo"	$400.00	5	21	5	5	21	21	5.0	21.0
["Room-dar	$45.00	29	60	29	29	60	60	29.0	60.0
["First aid kit	$78.00	30	180	30	30	180	180	30.0	180.0
["Shampoo"	$165.00	30	90	30	30	90	90	30.0	90.0
["TV with sta	$165.00	30	365	30	30	365	365	30.0	365.0
["Mini fridge	$81.00	2	7	2	2	7	1125	2.0	1033.1
["Shampoo"	$223.00	29	500	29	29	500	500	29.0	500.0
["Shampoo"	$55.00	7	1125	7	7	1125	1125	7.0	1125.0
["Shampoo"	$55.00	3	30	3	3	30	30	3.0	30.0
["Shampoo"	$156.00	3	30	3	3	30	30	3.0	30.0
["Shampoo"	$130.00	60	1125	60	60	1125	1125	60.0	1125.0
["Shampoo"	$50.00	30	1125	30	30	1125	1125	30.0	1125.0

Note. A sample listings.csv raw dataset consisting of different variables

The information gathered in the form of records is kept in relational databases. The listings dataset is a record type dataset since every entry specifies the distinct id and its associated attributes as the columns. With each listing id, the dataset contains previous customer ratings. The review dataset contains anywhere from 3 to 120 visitor feedbacks for each listing. The collected data has different types of listing like Home or Apartment, Private Room or Shared Room. This data was uploaded with clear filters and provides a robust base for any public research or debate. Location information was uploaded anonymously by AirBnb. This data includes the latitude and longitude information of each region in the catalog. However, not any street or home address is summarized from it. Apart from this, this data is tied up with the geographical maps, and it can be used to optimize the locality within a city. The size of the datasets would be hundreds of megabytes. Some of the attributes for reviews dataset include host_id, which is host identification number, date is when the review was stated, listing_id, the

reviewer's name is the reviewer attribute, reviewer's remarks are comments by the customers, and reviewer's id is the customer id. Likewise, every listing has about 25,690 unique ids and 75 characteristics in the listings.csv file. The listings id is listing_id, the name of the listing would be name, the name of the host is the host_name, the room_types provided by the housing are categorised into the full home/apartment, private or hotel room and shared room. The neighborhood_group are the areas in San Francisco and San Diego cities where Airbnb properties are available. Other attributes include the number of bedrooms and bathrooms in the house, the price of the property in dollars, and so on.

We have used the Inside Airbnb web portal to collect the data from different countries. We are going to merge the San Francisco and San Diego region's data, and make it one source file in order to make use of most of the information.

Algorithm Design

Data Cleaning

After collecting the data, the next step is to clean the data and make it ready for analysis. Since the datasets contain textual, numerical, null, and zero-valued records, normalization and cleaning will be required. To start with, all the columns that are not related and relevant to the price prediction, such as url, host name, etc. will be removed. Numeric columns with fewer missing values will be treated by replacing the missing values with mean/median. Next, all the attributes that have a significant amount of missing values or nulls will be removed. We will then determine and eliminate variables with repeated and irreversible missing fields and set them to zero. This will help in reducing noise in prediction models and improve their accuracy. Some of the columns which have multi-valued attributes will be converted into multiple appropriate rows/ columns in order to get the data to be in a completely relational format. Additionally, numeric

columns with special characters such as '$', ',', etc. will be treated and converted into true numeric format and the precision of floating point columns will be fixed to an appropriate value.

Outlier Handling

Once the data is cleaned and formatted, we will be checking if the value that each row holds is within the expected limits. The data points that diverge very much from the normal data points in a specific dataset are called Outliers. If we want the accurate outcome of our model,we should identify these unusual patterns very carefully and remove them from the dataset. The main problem with the outliers is that we don't have any labelled data to use as a training dataset for this irregular recognition. The outlier observation and removal is a very crucial part while building the model for better precision and accurate output. In this model, if the price prediction for Airbnb contains the outliers, it can infect the execution of the all-inclusive model. Values that are extreme and rare, which tend to skew the results of analysis will be removed from analysis, unless we find a clear reason for the out of limit value. Identifying such extreme values will be done using either of the methods such as IQR limits, SVM one-class algorithm, etc.

Exploratory Data Analysis

Once the data is clean, formatted and analysis ready, we will analyse the data with the help of multiple plots to understand it's distribution and what it represents. Firstly, we will find the descriptive statistics such as mean, median, mode, range, skewness and kurtosis. These univariate plots will show whether the data points are normally distributed. Post univariate analysis, we start adding features one by one and analyse the impact each feature has on one-another using bivariate and multivariate plots. (Carrillo, 2020) With these plots, we will be able to recognize significant correlations between one or more attributes and identify whether a particular attribute can be considered a feature or not. For instance, significance in the

relationship between price and host will imply that customers prefer to rent accommodation from verified hosts and hence, hosts can work towards getting verified in order to increase occupancy rate. Apart from these, an extensive EDA will help us identify underlying patterns within the data and arrive at valuable recommendations and derive insights out of it.

Hypothesis Testing

With a fair understanding of the relationship between attributes from EDA, we will be checking for their statistical significance with the help of hypothesis tests. Based on the distribution of the data observed for each attribute from the univariate analysis, we will choose between parametric and non-parametric tests. We will be using a significance level of 5% in order to reject or fail to reject the null hypothesis.

Feature Selection

Based on the results from hypothesis tests and EDA, we will choose the attributes that have shown significance towards the target variable Price as features. If deemed necessary, we will further refine the features using the backwise elimination method based on the performance of the predictive models. If the initially selected combination of features introduce a substantial amount of noise to the model, refinement of the selected features will be required. However, if the selected combination of features does not turn out to be good predictors of price, we will re-access the features and try different combinations of them until we find the most predictive set of features.

Dimensionality Reduction

Once the feature selection is done, we need to reduce the number of features into 2, so that it will be easier for the human brain to understand patterns they form with the target

variable. We aim to do the dimensionality reduction using the Vector Assembler method. The reduced 2-dimensional features will be used for predictive modeling.

Predictive Modeling

Once we have all the features ready, we will be building various machine learning models to predict the price of Airbnb listings. We will be working on different machine learning algorithms (Spark 3.2.0 Documentation.) like Linear Regression, Gradient Boosting Algorithm, Ridge and Lasso Regression etc. to predict the best possible price for the customers as well as for the property owners. After building the models, we will train them using our training dataset, evaluate the performance, optimize the model parameters to improve accuracy and finally choose the model that is the best fit for this case.

Languages Used

In this project, we will be using PySpark to perform data analysis on a large Airbnb dataset. Spark is a cluster-based computing platform for distributed data processing on clusters of computers. When datasets get very big for a single computer to process quickly, Spark's leverage becomes clearly visible as it works on distributed computing through cloud servers.

Pyspark is an API of Python for Spark, and it allows users to use RDDs(Resilient Distributed Datasets) in a Python context. The language also provides examples of regression estimators being conducted on the Inside Airbnb listings dataset from various regions. The pricing acts as the target variable. Following packages are few of the packages we will be using as a part of implementation: Python 3, Numpy, Pandas, Scikit-learn, PySpark, mlib

Tools Used

For this analysis we will be programming in PySpark and Python using Jupyter Notebook and Databricks. The Jupyter Notebook is an open source tool that helps in creating and sharing documents that contain live texts, code, and visualizations. Databricks is a cloud based web platform to compute and extract all the Spark codes along with automated cluster handling.

How to Generate Output

Once we build our predictive models using appropriate machine learning techniques, we will be validating the performance of these models using test data. We will be splitting our data into train and test with a 75-25 ratio. The train data will be used to train the model and validation will be done using the test data, to evaluate the accuracy of these models. We will compare the performance of these models based on performance metrics such as RMSSE, accuracy, precision, etc. and use the model with the smallest RMSSE as the final model to predict the price of Airbnb listings.

How to Test Against Hypotheses

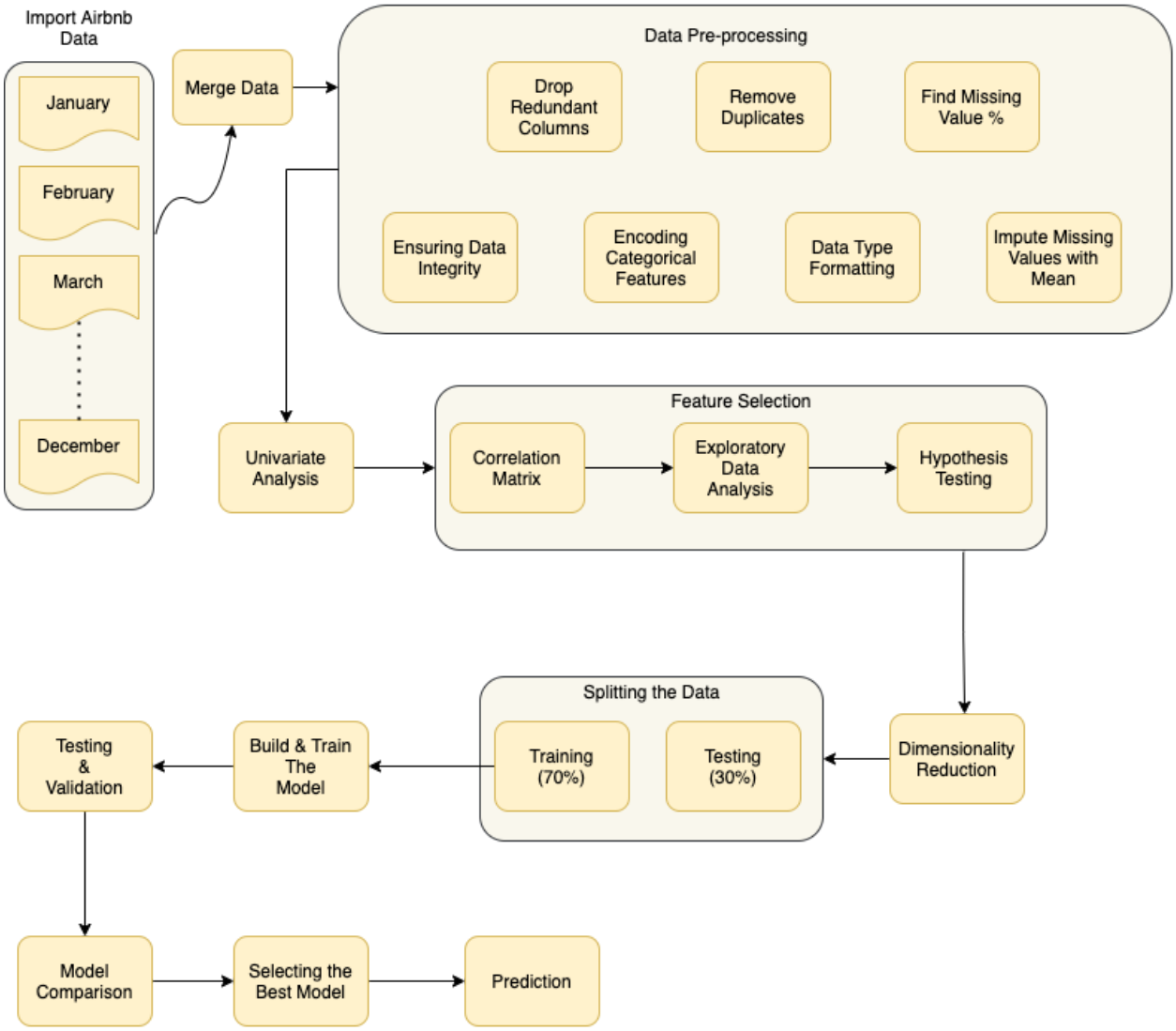
If the accuracy of our prediction is less than 90% on average for each listing in the testing set, our test may fail to prove enough strength on predicting the further listings in the prediction window. With newer data getting added to the Airbnb website everyday, applying our model to the newer data and finding the accuracy will be another way to test our model and the hypotheses we have.

Implementation

Design Document & Flowchart

Figure 4

Project Flowchart



Note. Flowchart created using draw.io

Code Implementation

We have started code implementation with importing the Spark functions, statements, libraries setting up the spark context, and initializing the schema. We started implementing the code locally with Jupyter, but with increasing data we moved to Databricks platform. The next step was to import the dataset. There were a total of 12 files; each file was representing the month of the year. We created a cluster on Databricks cloud and imported all the csv files. We merged the csv files through the Sparksql function. Total size of the combined dataset is 320 MB. To make sure the data was imported correctly, we displayed the data and schema in the next step.

The data was imported with duplicates; hence, it was crucial to remove the duplicates as it might impact the outcome and the probability of getting biased results would increase. So, to avoid that, we removed the duplicates in the next step. There were loads of missing values in the dataset. We first checked the missing values percentage for each column, and removed the ones with highest percentages of missing values in the range of 75-80%. The output would be stored in the temporary table 'Airbnb_Data' to retrieve the data easily.

In the next step, we have done the data cleaning. There are several things we did in cleaning the data, and will be explaining them in detail. We started with correcting the data format like, the timestamp column had both date and time, so extracted the date and time with correct format that too separately. Then, we checked the number of years the host had the property, and the location of the property. Once identified, we converted the categorical columns to the numerical ones. We checked the junk values in the dataset; converted them into the null or the 0 values to balance the dataset and avoid the imbalance in the outcome. We have also checked for the unnecessary percentage symbols and removed it from 'host_response_rate', and

‘host_acceptance_rate’ columns. Also, convert the datatype of these columns into integers. We have checked the null and missing values and filled them into the mean value. The columns like ‘host_is_superhost’, ‘host_has_profile_pic’, and ‘host_identity_verified’ have the true or false values. We converted those values into flag values like 0 or 1.

In the dataset, there was a column named ‘room_type’. There were different categories like 'Shared room', ‘Entire apartments’, ‘Hotel room’, ‘'Entire home/apt', 'Private room', 'Private room in rental unit', 'Private room in apartment', 'Private room in house', 'Private room in residential home', 'Entire condominium', 'Entire condominium (condo)' and 'Entire loft'. We have categorized them into the range of 0 to 3. For all the categories starting with ‘Entire’ word were set to 0, for the ‘Shared’ categories 1 was set, for the ‘Hotel rooms’ we have set 2. And for all the ‘Private’ categories flag 3 was set. Moreover, there was a column named ‘property’ which had too much irrelevant data in the column, so we removed the column as it was not a good feature for our outcome. We also kept the auto increment for the ‘amenities’ column, and removed the dollar sign from the amount column to maintain the consistency in the dataset.

The next part was to check the missing and duplicate percentage values again after the encoding part was done. After that we kept all the categorical and numerical features manually and defined them to standardize our data. After defining all the columns, we have displayed them. The next and the crucial part was to show the descriptive statistics of the dataset after cleaning it. In the descriptive statistics, we showed count, mean, standard deviation, minimum, maximum values, etc.,

Once we have calculated the descriptive statistics, we need to select the most important features for our machine learning modeling. To find that, we have calculated the Correlation Matrix. From this, we can identify the relationship between most two relevant features. Here, if

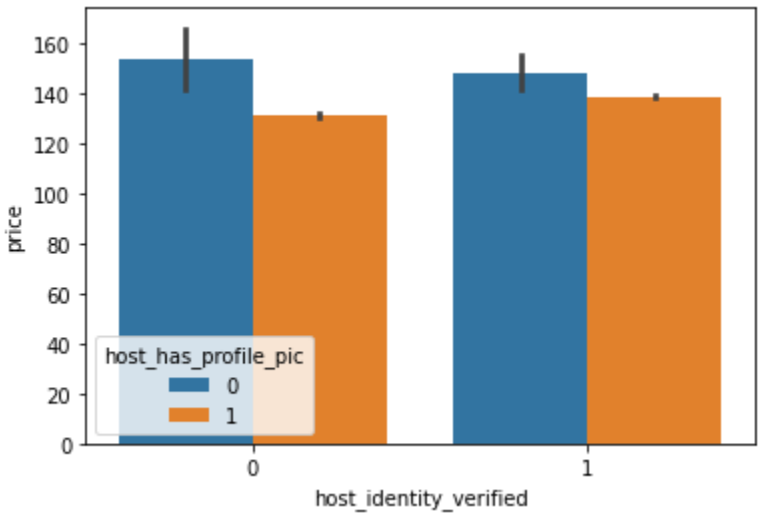
we find a very high correlation between any two columns, then we need to remove it as it would not be the best feature for our model. We are taking the 'price' column as target features here, hence, any feature that has a higher relation with the 'price' column will be the important one. From this, we can select the most significant features for the next stage, and also do the hypothesis testing based on the features.

Exploratory Data Analysis

We have performed many Exploratory Data Analysis in order to select the relevant features to execute the modeling. The EDA was an important factor that helped us to select the attributes as we had loads of features**.** We can also explore and understand our data clearly with the help of the EDA. If we had not selected the most relevant features, it could impact the behavior of the output. Here are some EDA outputs that helped us to select appropriate attributes.

Figure 5

Exploratory Data Analysis - I

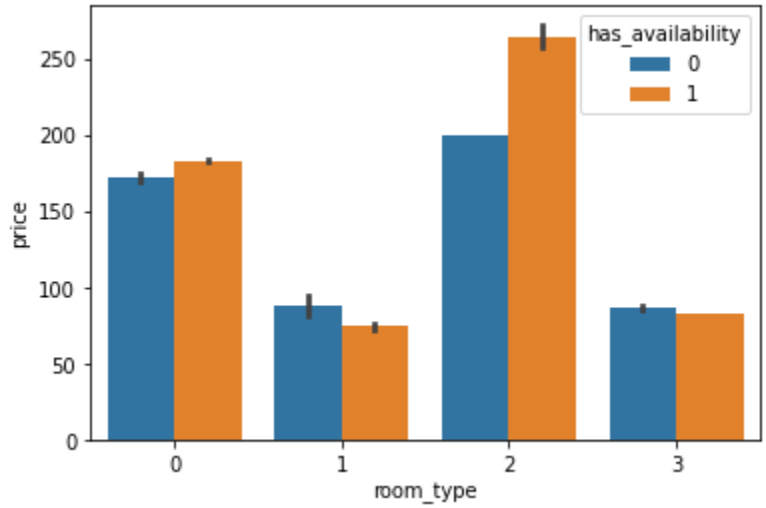


Note. Host Identity vs Price

Above figure shows the host identification in the form of a bar chart. It represents the relationship between the price and the host's profile picture. If the host has no profile picture, the price of the property tends to be higher than the property where the host has the profile picture.

Figure 6

Exploratory Data Analysis - II

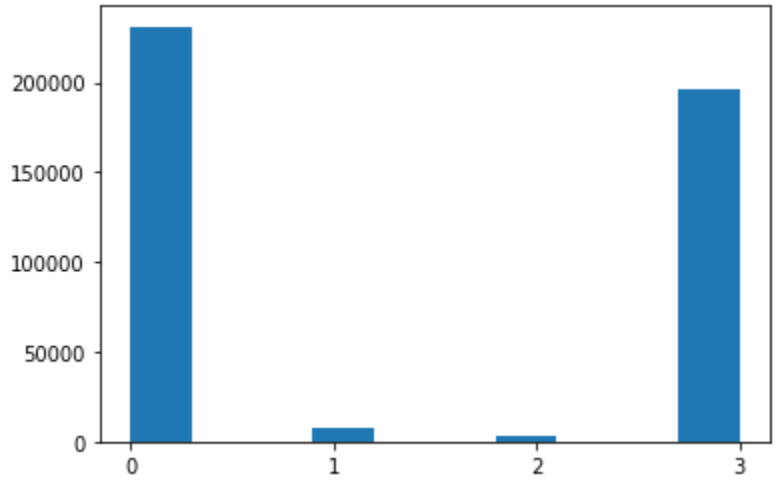


*Note.*Room type vs Price

This Bar chart explains the relationship between the room_type and the price of the property. As we have mentioned the room types above in the code implementation part, for all the categories starting with 'Entire' word were set to 0, for the 'Shared' categories 1 was set, for the 'Hotel rooms' we have set 2. And for all the 'Private' categories flag 3 was set. Here, Hotel rooms were mostly in demand. On the other side, private properties were not in the least demand and their prices were low too.

Figure 7

Exploratory Data Analysis - III

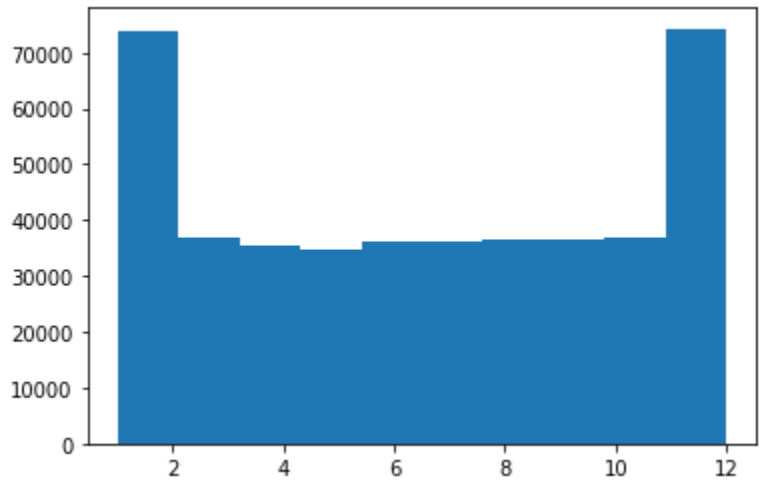


Note. Room type vs Price of Property

In this figure, the x axis shows the room types and the y axis shows the number of properties booked. Type 0 and type 3 i.e., entire properties and private properties were having around 200000 bookings; and, on the other side, type 1 and type 2 had less than 50000 bookings. Hence, this histogram would be very helpful to check which properties were having more bookings.

Figure 8

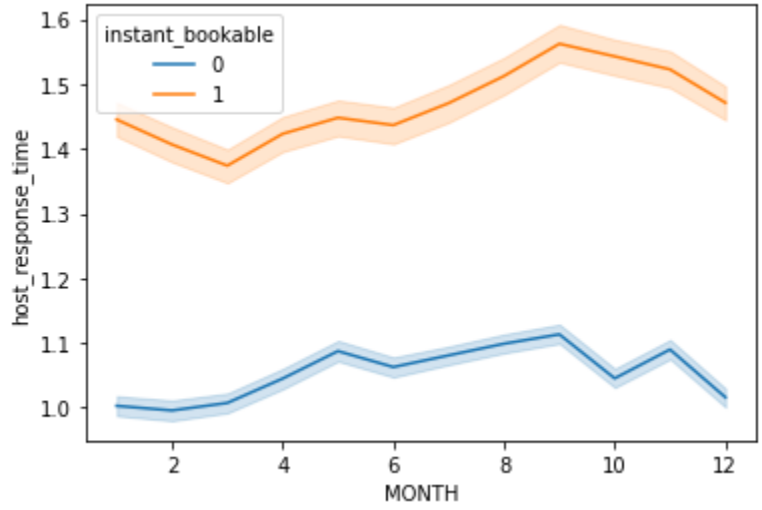
Exploratory Data Analysis - IV



In the above histogram, x axis represents the months of a year and y axis represents the bookings happening during the time. It clearly seemed that in the month of January and December, there were more bookings happening in comparison to the rest of the months.

Figure 9

Exploratory Data Analysis - IV

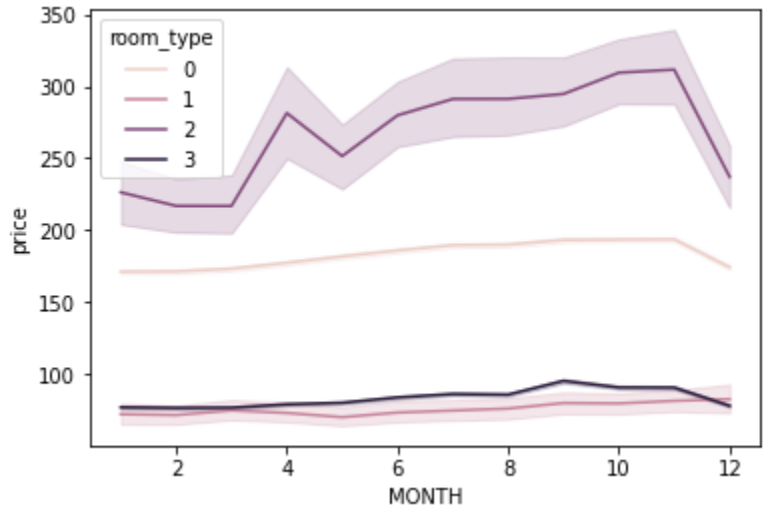


Note. host response time vs Month

The above line chart shows the relationship between the host_response_time vs month. The host's response time also impacts as per the duration of the year. Moreover, instant bookings were dependent on the response time of the host. The orange line represents the instant booking available, whereas the blue line represents the instant bookings were not available.

Figure 10

Exploratory Data Analysis - V

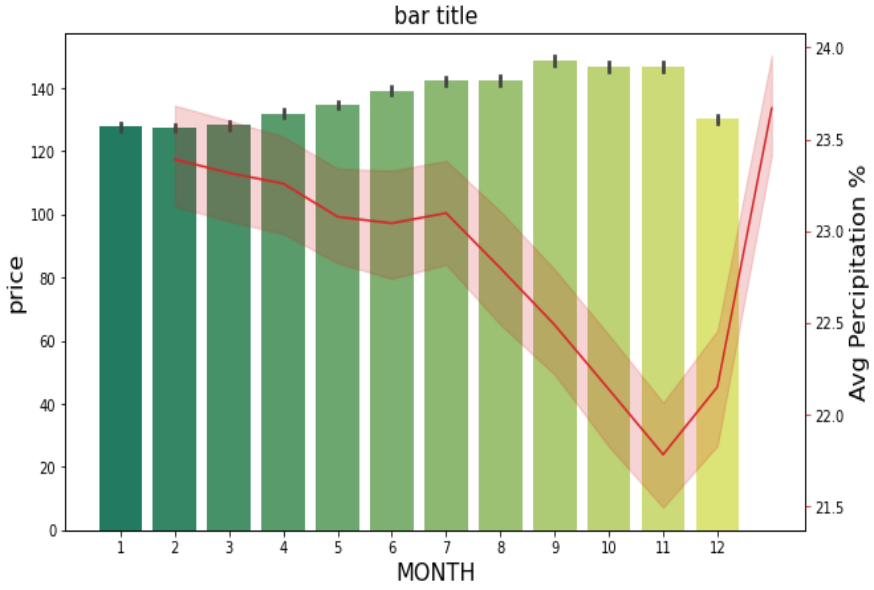


Note. Month vs Price of property

In the above line graph, the x axis shows the month and the y axis shows the price. Room type 1 and 3 (shared and private properties respectively) are listed with a low price range. On the other side, Room type 0 and 2 (entire and hotel rooms) are listed with a higher price range. Also, there is a significant change in the type 0 and 2 properties during the end of the year.

Figure 11

Exploratory Data Analysis - VI

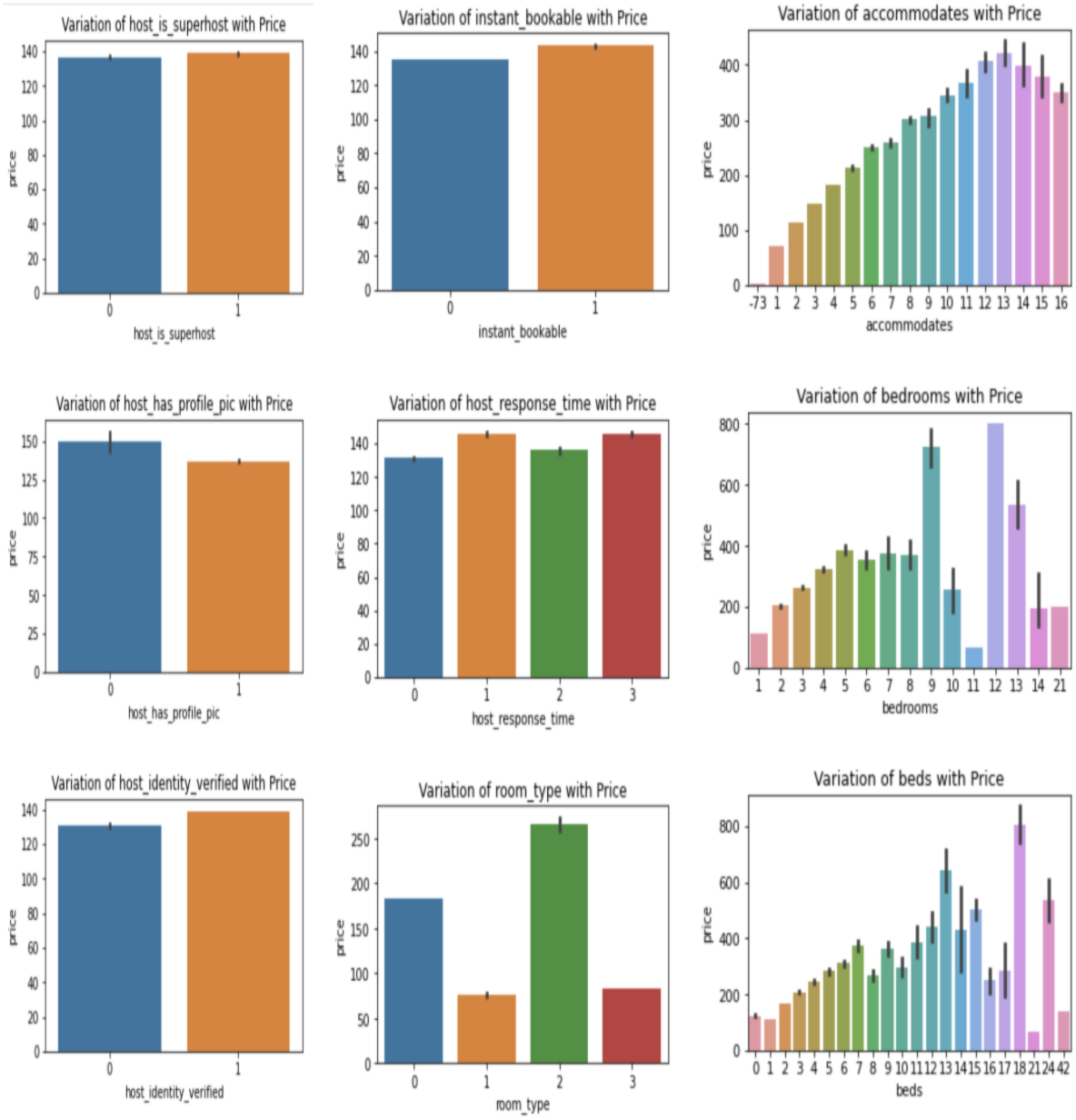


Note. Bar graph for Month vs Price

The above figure shows the combination of a bar chart and a line chart. Here, the x axis shows the months, and the y axis shows the price for the bar chart. It also shows the average precipitation in terms of line charts. That means the bookings are decreasing in the last quarter of the year when there are minimum nights for any booking. However, the last month of the year is despite any of these factors and there is a significant increase on the booking in the last month of the year.

Figure 12

Exploratory Data Analysis - VII



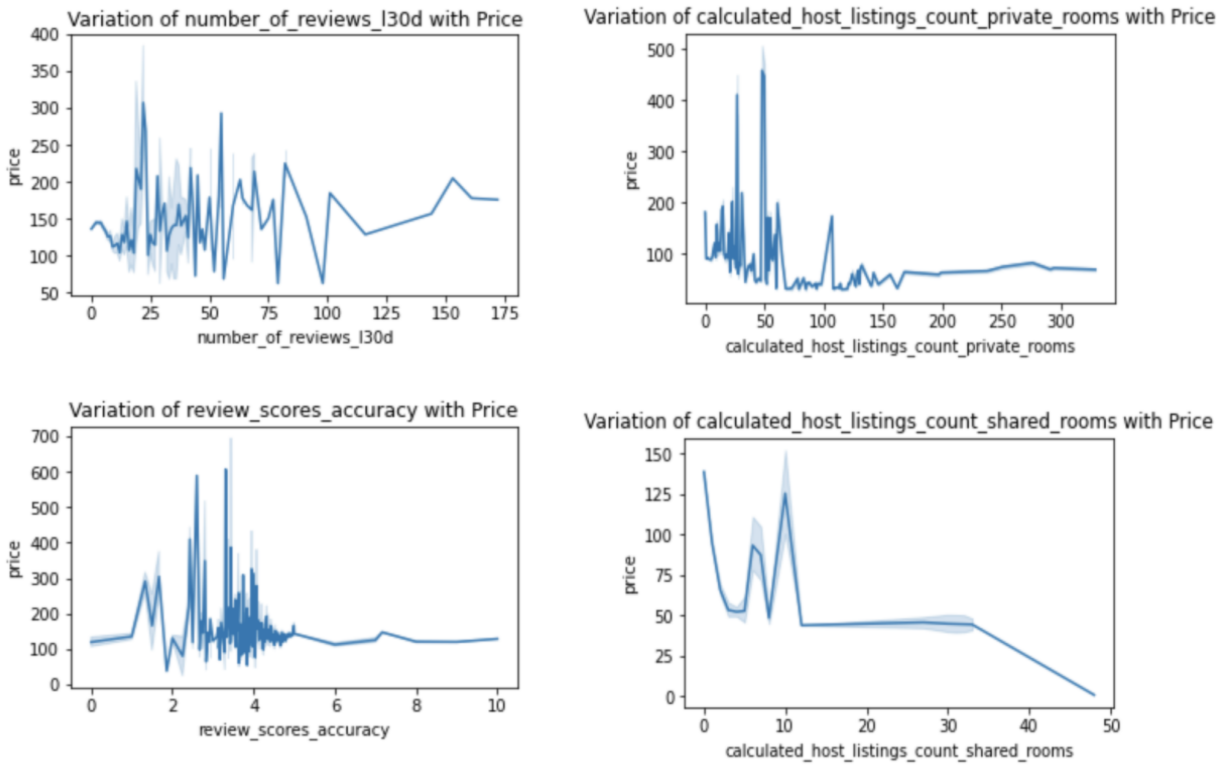
Note. Variation of features with the target variable

The above part of the Exploratory Data Analysis shows the Variation with Target for Categorical Features. To show the relationship of the Price attribute with all the other attributes we have used a bar chart. There are multiple factors play crucial role in predicting the price of

the property like if the host is superhost or not, of the host has profile picture or not, if the identity of the host is verified or not, if the property that host has listed instantly bookable or not, what is the response time of the host to book the property, if there are various room types available to choose or not, the number of amenities in the property, how many bedrooms are there in the property, if the property is available at the time of booking. As per the above listed factors, the price of the property is supposed to be changed accordingly. For example, the price will increase if the number of beds are more in the listing. Also, if the host is verified, the chances are more that the service provided by the host will be authenticated and standard, hence, the price of that property tends to be higher than the non-verified host.

Figure 13

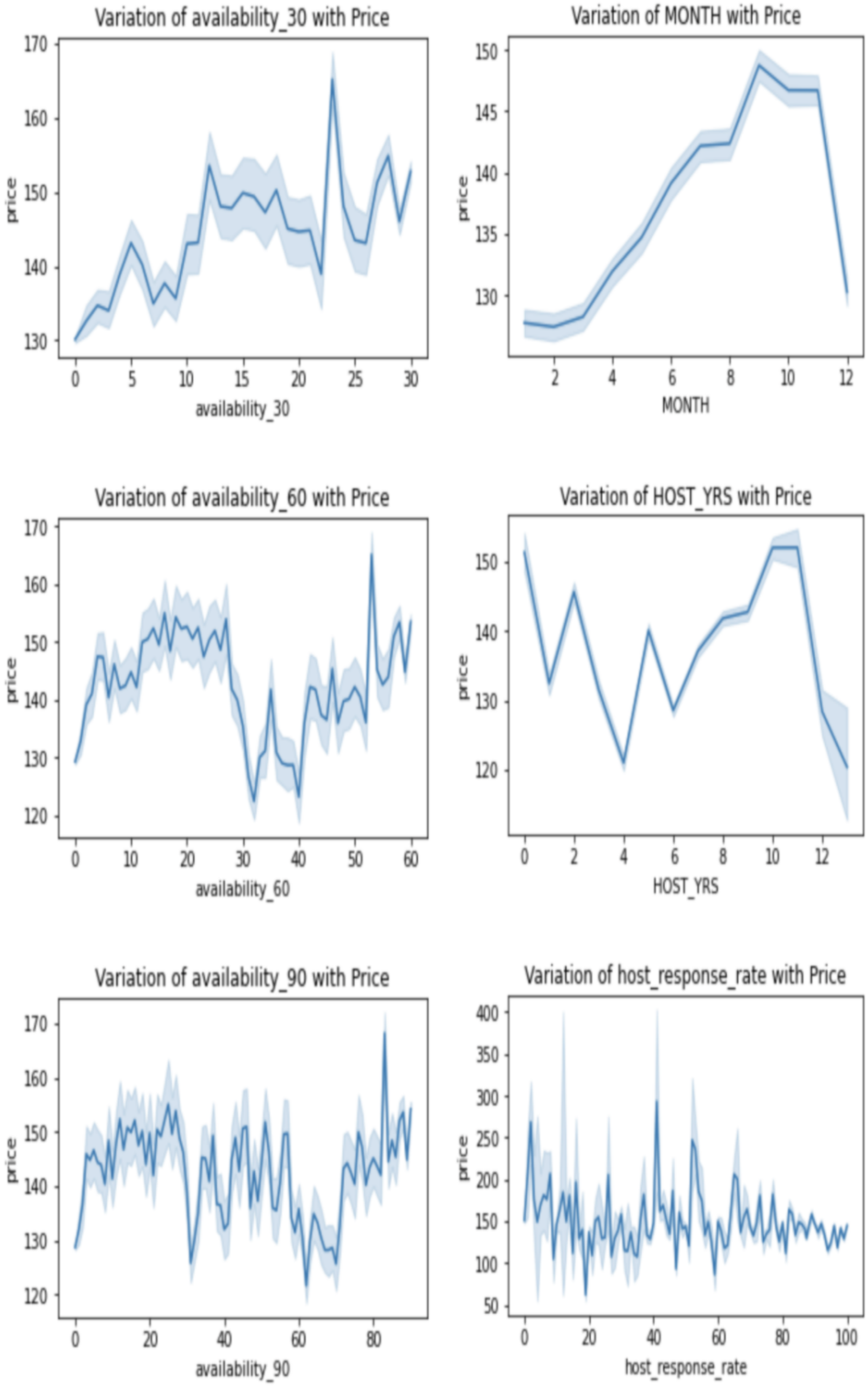
Exploratory Data Analysis - VIII

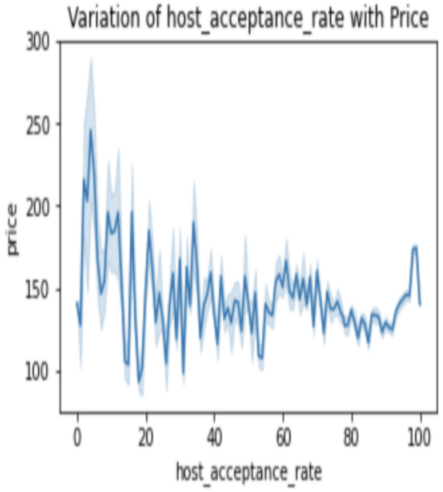


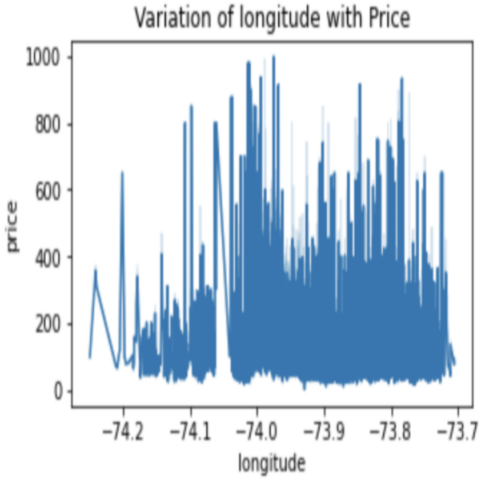
Note. Variation of numerical features with target variable

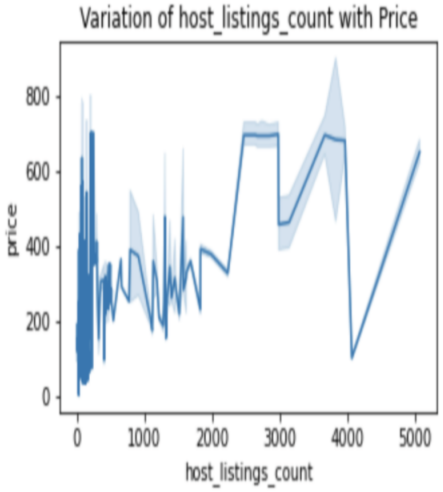
Figure 14

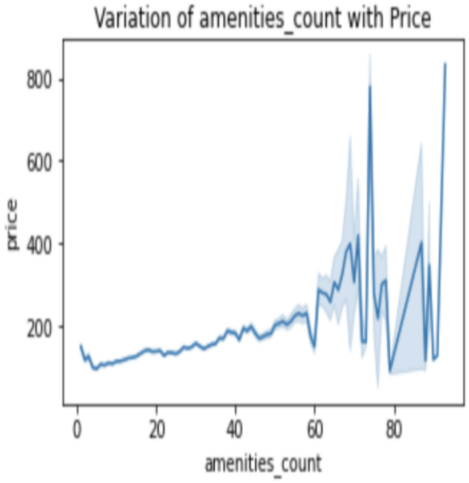
Exploratory Data Analysis - XI

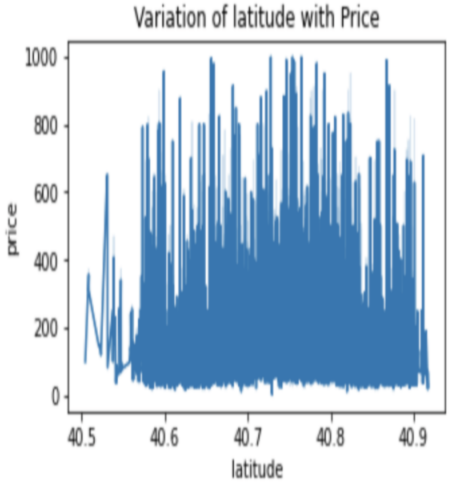


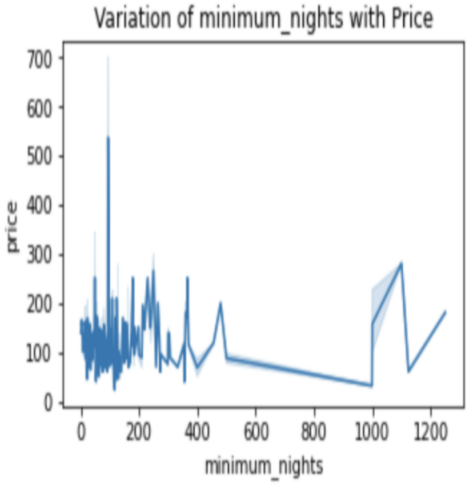

Variation of host_acceptance_rate with Price
price
host_acceptance_rate
300
250
200
150
100
0
20
40
60
80
100

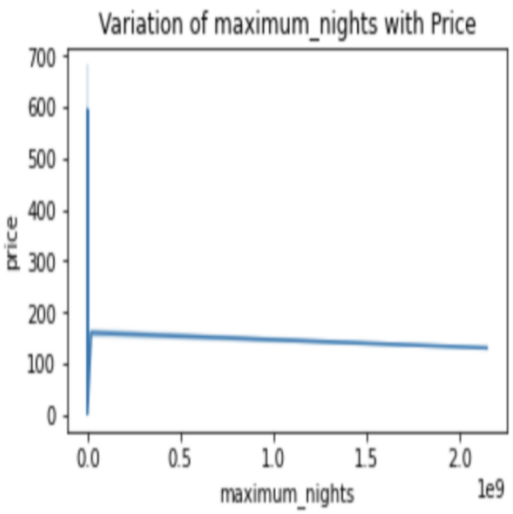

Variation of longitude with Price
price
longitude
1000
800
600
400
200
0
−74.2
−74.1
−74.0
−73.9
−73.8
−73.7

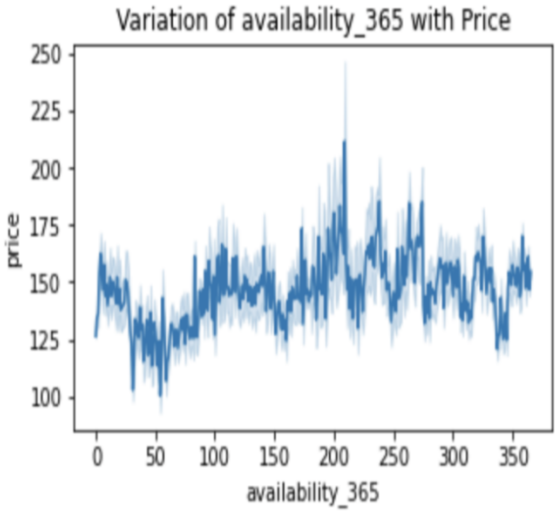

Variation of host_listings_count with Price
price
host_listings_count
800
600
400
200
0
0
1000
2000
3000
4000
5000

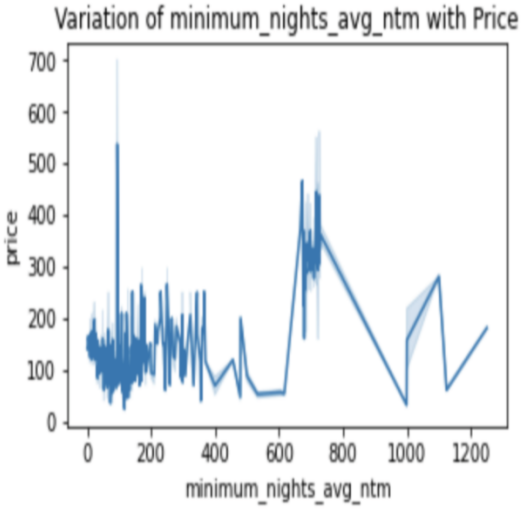

Variation of amenities_count with Price
price
amenities_count
800
600
400
200
0
20
40
60
80


Variation of latitude with Price
price
latitude
1000
800
600
400
200
0
40.5
40.6
40.7
40.8
40.9

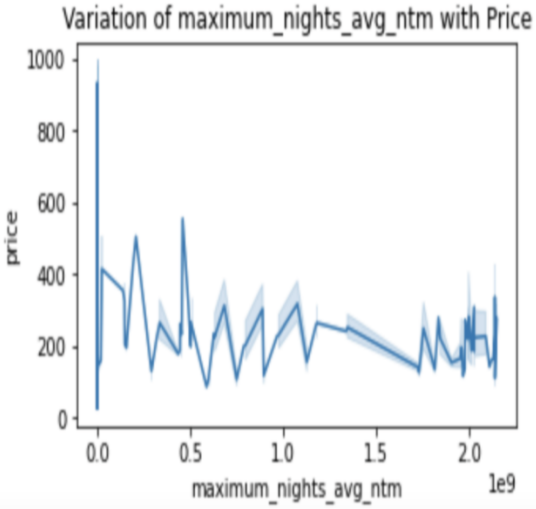

Variation of minimum_nights with Price
price
minimum_nights
700
600
500
400
300
200
100
0
0
200
400
600
800
1000
1200

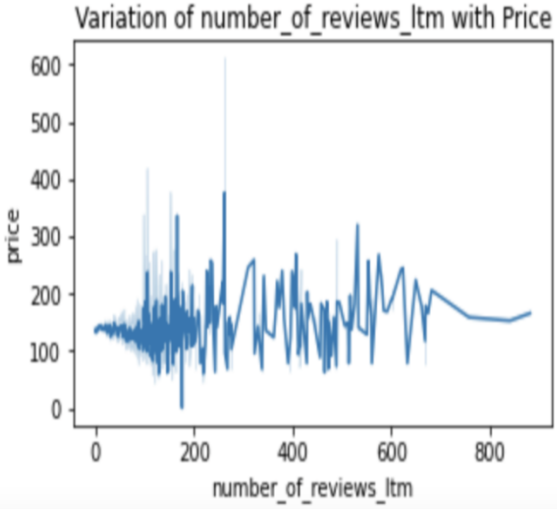

Variation of maximum_nights with Price
price
maximum_nights
1e9


Variation of availability_365 with Price
price
availability_365


Variation of minimum_nights_avg_ntm with Price
price
minimum_nights_avg_ntm


Variation of number_of_reviews with Price
price
number_of_reviews


Variation of maximum_nights_avg_ntm with Price
price
maximum_nights_avg_ntm
1e9


Variation of number_of_reviews_ltm with Price
price
number_of_reviews_ltm

Note. Variation of target variable with numeric features

The above part of the Exploratory Data Analysis shows the Variation with Target for Numeric Features. To show the relationship of the Price attribute with all the other attributes we have used a line chart. Here different variations of availability are shown with respect to the price. Apart from that the properties are listed for the whole month, and the price for those properties constantly increases in the second and third quarter of the year and then eventually decreases in the last quarter of the year. Moreover, the variance of host listings count, acceptable rate, price according to longitude and latitude, amenities count that impacts the price, and the minimum nights listings can also impact the price of the property. In the line chart, the reviews of prices, and the availability of the property at the pick time also matters to determine the price.

Data Analysis & Discussion

Output Generation

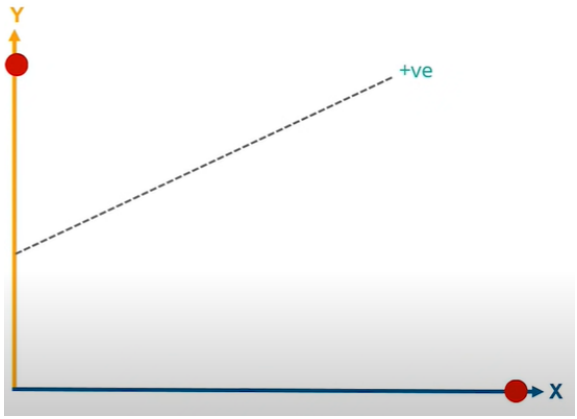
Model Implementation

Linear Regression. In simple linear regression, what we are trying to find is the correlation between the x and y variables that every value of x has a corresponding value of y in it if the target variable is continuous. As a core concept of linear regression, we can say that the data is modelled using a straight line. The output/prediction of a linear regression model is the value of the variable itself. In case of linear regression the goodness of the model is measured using evaluation metrics like loss functions like RMSE, MAE, R squared. One case scenario for linear regression is to analyze the impact of pricing changes on consumer behavior. In our case if an airbnb rental changes its quoted price several times then it can record the quantity itself or each price level and then perform a linear regression with the several features of the rental as its

dependent variable and price as its independent variable. This would result in a line that would depict the extent to which the customers make their choice of rental as the price is increasing and this result would help the hosts with future pricing decisions. Consider having an independent variable on the X axis and a dependent variable on the Y axis. If the independent variable is increasing on the X axis and the dependent variable is increasing on the Y-axis, this would result in a positive regression line as seen in Figure 15

Figure 15

Positive Regression Line.

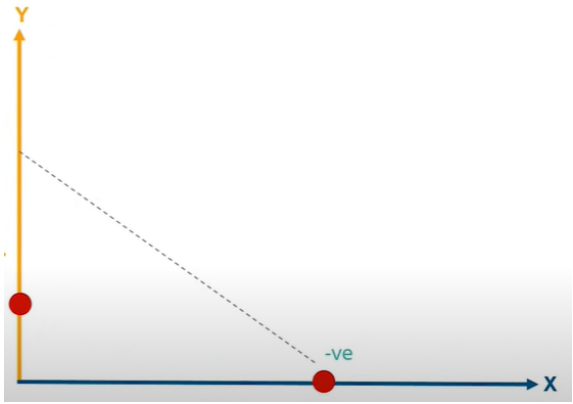


Note. Image source from (Gawali 2021)

Similarly if the independent variable is increasing on the X axis and the dependent variable is decreasing on the Y-axis, this would result in a negative regression line as seen in Figure 16 as the slope of the line is negative.

Figure 16

Negative Regression Line

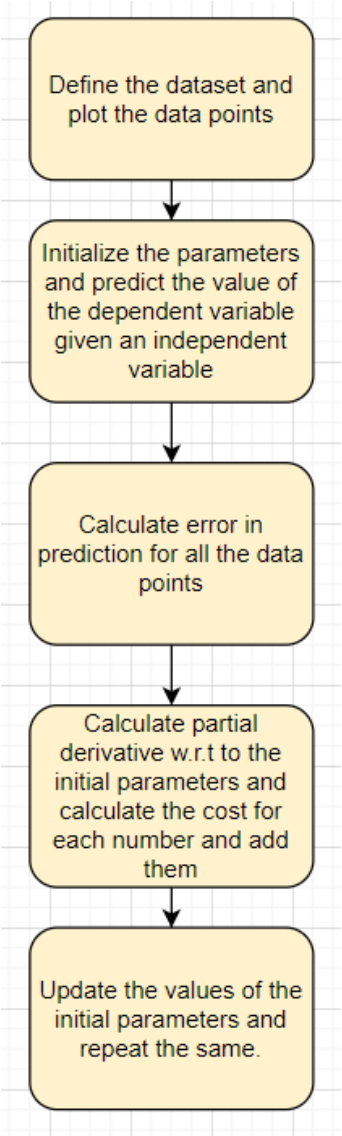


Note. Image source from (Gawali 2021)

The line mentioned above is the line of linear regression given by the equation y = mx + c which shows the relationship between the independent and the dependent variable. The goal of this algorithm is to reduce the difference or distance between the estimated and the predicted value and the best fit line will be the one which has the least difference. A high level implementation of the Linear Regression algorithm is as seen in Figure 17

Figure 17

Algorithm of Linear Regression



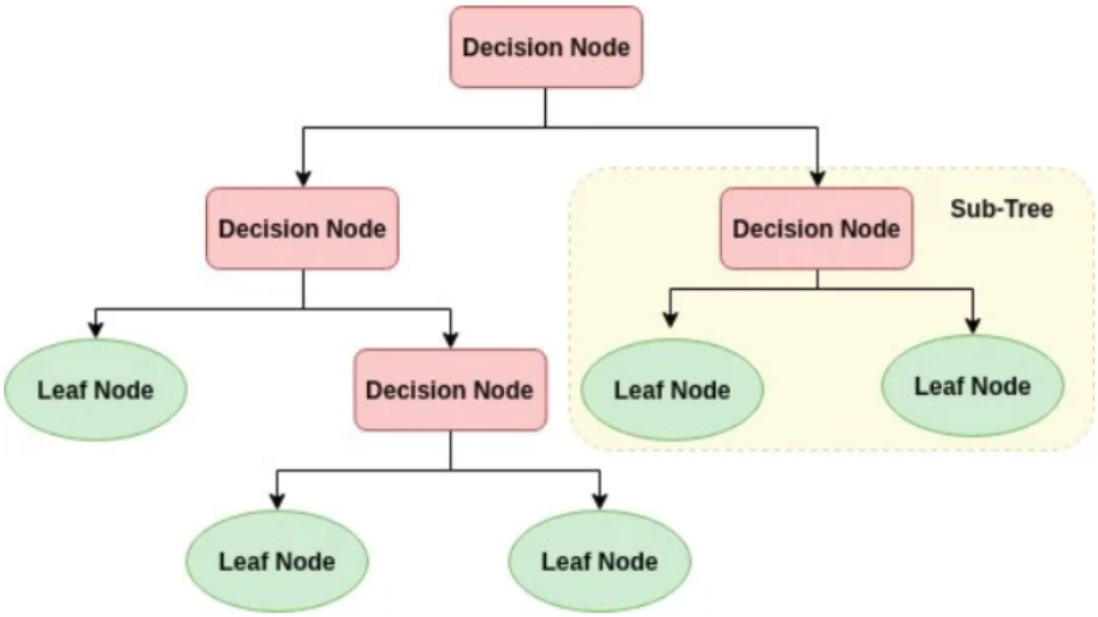
Note. Image created using draw.io (Reference : Gawali 2021)

Decision Tree Regression. A decision tree is a supervised machine learning model which predicts a target variable by learning decision rules by associated features. Decisions are made using a flowchart/tree like structure. The path from the root to leaf is known as a classification rule. Each of the internal nodes represent an attribute on which a decision is made. This

algorithm is highly suited to solve both classification and regression problems. Decision Tree regression is a non-linear regression technique. In the case of regression trees, the value obtained by the terminal node is the mean response of observation falling in that region. Hence if an unseen data point falls in that region, we can make a prediction with an average or mean value. Decision tree regressor is useful when the target variable is continuous in nature and in this our target variable is price. Figure 18 presents a pictorial representation of the decision tree algorithm and Figure 19 explains the steps involved in the decision tree algorithm on a very high level.

Figure 18

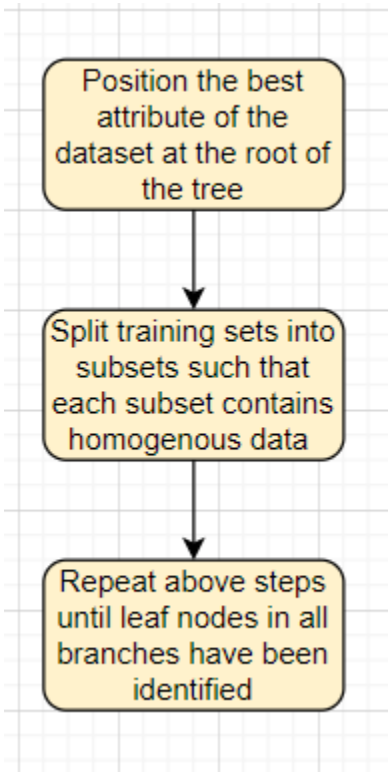
Pictorial Representation of Decision Tree Algorithm



Note. Image sourced from (Navlani 2018)

Figure 19

Decision Tree Algorithm



Note. Image created using draw.io

Mathematically, it can be explained using two concepts which are Entropy and Gini Index (ankit nitjsr Feb 2019)

$$Entropy = -\sum_{i=1}^{n} p_i * \log(p_i)$$

$$Gini\,index = 1 - \sum_{i=1}^{n} p_i^2$$

Where i represents the number of classes

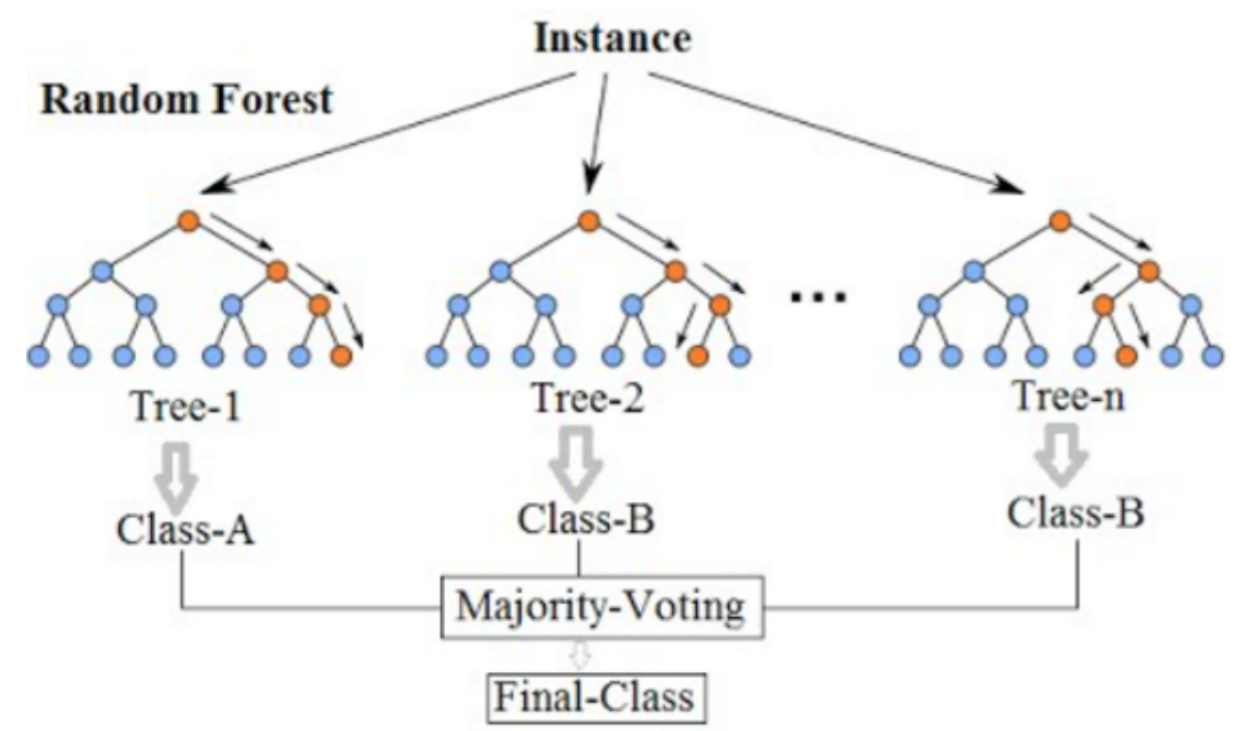
Random Forest Regression. Random forest regression is a type of supervised machine learning technique which is very effective when other types of machine learning algorithms are computationally expensive. Ensemble learning is a technique which takes predictions from multiple machine learning algorithms or predictions from the same algorithm multiple times in order to make more accurate predictions. Predictions from an individual model might not give that much accurate results and hence ensemble learning techniques to address this purpose. Thus a model composed of several machine learning models is called an ensemble learning model. Several models like decision tree, KNN and SVM are used on the same data to get the combined predictions from them. Likewise several decision trees can be trained together to get a final prediction. Ensemble learning comes in two forms which are boosting and bagging. In boosting learners are learnt sequentially with early learners fitting simple models to the data and then analyzing data for errors. Consequent trees which are random samples are fit and at every step the goal is to improve accuracy from the prior tree. Bagging on the other hand is used to create several subsets of data from training samples chosen randomly with replacement. Each collection of subset data is used to train their decision tree.

Random Forest utilizes the bagging technique. Two major drawbacks of decision trees can be overcome using random forest algorithms. One is that decision trees are computationally expensive and the second is decision trees are very sensitive to data on which they are trained which may result in deviations if the underlying data gets changed. Random Forest is an ensemble learning technique that is used for both regression and classification problems. We will be implementing random forest as a regression in our project. It creates multiple decision trees during the training process and combines the result of predictions from each decision tree to determine the final output. Trees in the random forest algorithm run in a parallel manner. The

pictorial representation and the algorithm of Random forest has been explained in the Figures 20 and 21

Figure 20

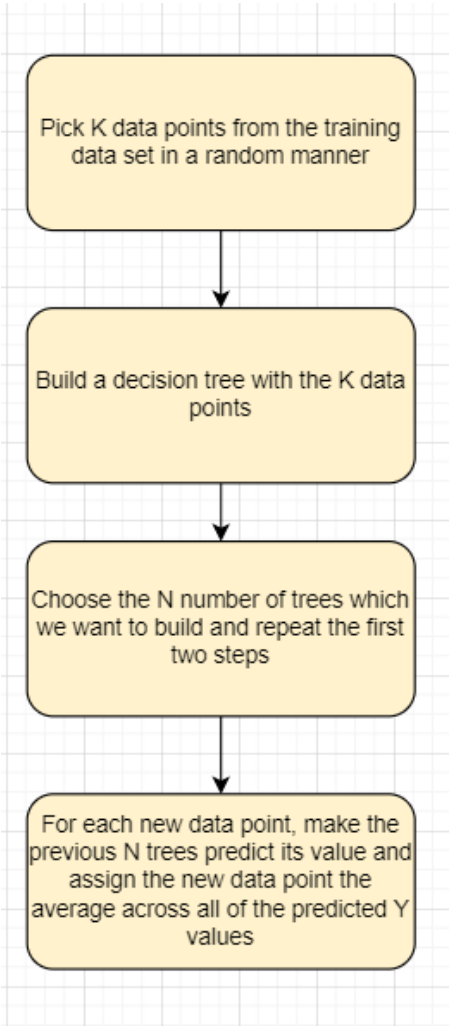
Visual Representation of Random Forest



Note. Image Reference from (Singh 2019)

Figure 21

Algorithm design of Random Forest



Note. Image created sing draw.io

Mean Squared Error is used to represent how data branches out from each node in the tree. Mathematically it can be explained as follows (Schott 2019)

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (fi - yi)^2$$

where

N => number of data points

fi => value returned by the model

yi => Actual value of data point

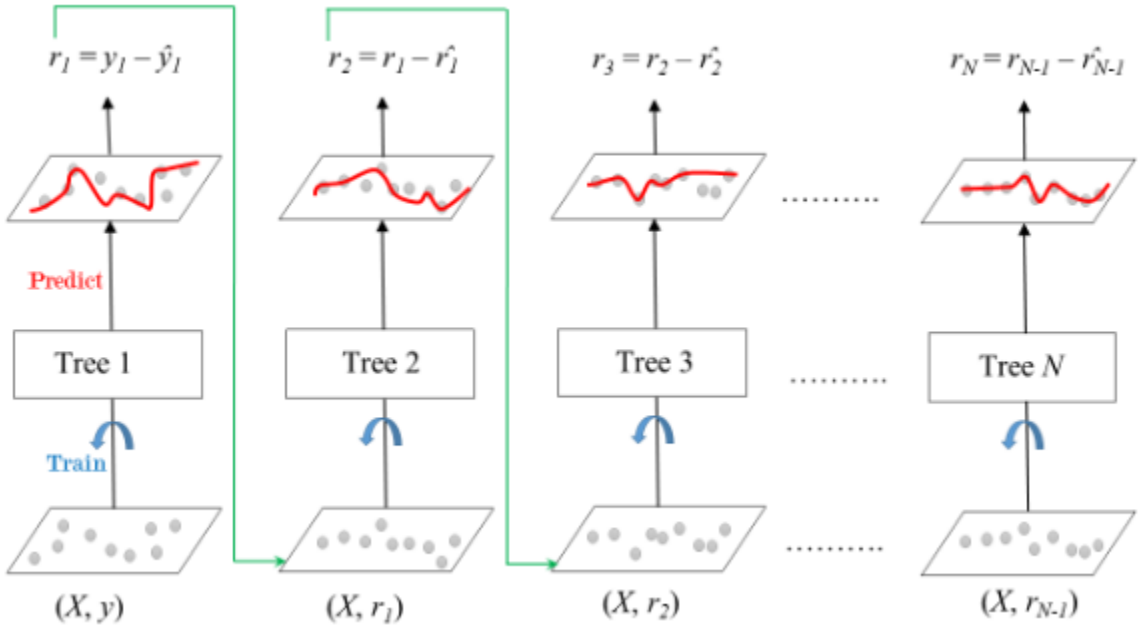
Gradient Boost Regression. When gradient boost is used to predict a continuous value like price, it is called using Gradient Boost for Regression. The complexity of the algorithm is due to the fact that it was configured to be used in a wide variety of ways. Though in most cases, only a single configuration is used to predict continuous values like the price and one configuration is used to classify samples into different categories. Gradient boost algorithm holds a lot of similarity to the AdaBoost Algorithm. For predicting price using some features, AdaBoost starts by building a short tree called stumps from the training data and the significance which the stump has towards the final output is based on how well it compensates for the previous errors. Then Adaboost builds the next stump based on errors made in the previous stump. This process continues until the algorithm finds a perfect fit.

In contrast gradient boost starts working by making a single leaf instead of stumps or trees. The leaf represents an initial guess for the weights of all the samples. The first guess while trying to predict a continuous value is usually the mean. The algorithm then builds a tree which is based on the errors made previously. The errors in this case is the difference between the observed and predicted weights. The difference here is that a tree is considered bigger with more branches than a stump. In general terms, the number of leaves to be built is usually set between 8 and 32. So in short, gradient boost builds trees like the AdaBoost algorithm but the size of the tree is larger than the stumps in AdaBoost. Both gradient boost and AdaBoost scale the trees but gradient boost scales all trees by the same amount. Similar to AdaBoost, gradient boost builds a number of trees as asked by us or some additional trees if the model is poorly fitted. When the

observed and predicted values have a very accurate match then we call it having a low bias but a very high variance. To address this problem gradient boost uses a learning rate which is a value between 0 and 1, to scale the contribution from the new tree. Figure 22 gives the pictorial representation of the Gradient Boost algorithm.

Figure 22

Pictorial Representation of Gradient Boost

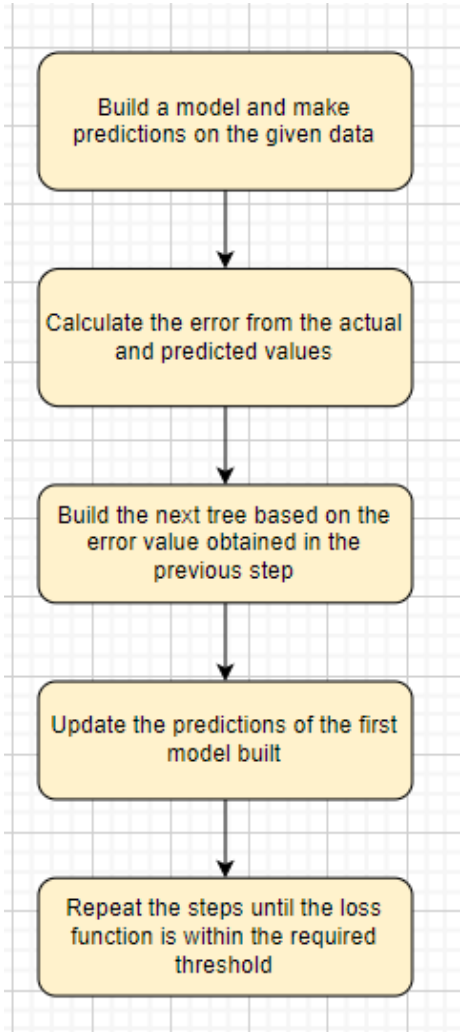


Note. Image Reference from (

The step by step algorithm of gradient Boost is seen in Figure 23

Figure 23

Algorithm of Gradient Boost Algorithm



Note. Image created using draw.io

Mathematically, it can be explained as follows (Saxena March 2021)

$$\text{Loss} = \underbrace{L(y, F_n(x))} + \gamma_n L\left(H(x, -\frac{dL}{dF_n(x)})\right)$$

where Fn(X) represents the difference between actual and predicted values

Output Analysis

Evaluation Metrics

We have tested the accuracy of our model by using R squared and Root Mean Squared Error (RMSE) evaluation metrics.

Root Mean Squared Error. The RMSE metric is nothing but the square root of mean squared error represented by the below formula (Shrivastava 2019)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Where N is the total number of observations

It basically tells us how concentrated the data is around the best fit. The reason for introducing squared is to bring the errors on the same scale as the target values scale. In other words, RMSE will have the same units as the target variable whereas in case of MSE it will have squared units. RMSE has the advantage of penalizing large errors more which proves to be more appropriate in some cases.

R Squared Error: R squared error is another evaluation metric used to determine the performance of the regression model. R squared is used to explain to what extent the variance of one variable affects the variance of the second variable. Variance is a measure of how far a set of numbers is spread out from its average value. If the R squared of a model is 0.5, half of the observed variance can be explained by the model's inputs. In other words, it implies 50% of the data fits the model perfectly and falls on the regression line. The formula of R squared is as explained below (Agrawal 2021).

$$\text{R2 Squared} = 1 - \frac{SSr}{SSm}$$

SSr = Squared sum error of regression line

SSm = Squared sum error of mean line

Table 1 explains the results of the models implemented to determine the price of the rentals.

Table 1

Comparison of Evaluation Metrics

Models Implemented	RMSE	R Squared
Linear Regression	10.465	0.7
Decision Tree Regression	5.08	0.94
Random Forest Regression	5.83	0.92
Gradient Boost Regression	1.2	0.9

Note. Comparison of evaluation metrics of all the regression model

Comparison of Output Against Hypothesis Testing

As a part of feature selection for the predictive modeling and identifying the significance of each variable towards the target variable price, the following hypothesis tests were performed.

Correlation for Numerical Features

The significance of the independent variables that are of numeric data type toward the continuous target variable price were identified based on the correlation values. Pearson's correlation test was performed and the resultant correlation coefficient can be found in Figure 24:

Figure 24

Correlation for Numerical Features - I

```
Correlation of each independent varable with the dependant variable Price:

MONTH                                  0.001247
HOST_YRS                               0.068665
host_response_rate                     0.009739
host_acceptance_rate                   0.005846
host_listings_count                    0.024805
latitude                              -0.041300
longitude                             -0.068421
accommodates                           0.722844
bedrooms                               0.660890
beds                                   0.766607
amenities_count                        0.209593
price                                  1.000000
minimum_nights                         0.001486
maximum_nights                         0.000606
minimum_minimum_nights                 0.002582
maximum_minimum_nights                 0.026036
minimum_maximum_nights                -0.020914
maximum_maximum_nights                -0.028531
```

Note. Correlation Coefficient of numeric features with target variable

Figure 25

Correlation for Numerical Features - II

```
minimum_nights_avg_ntm                          0.025187
maximum_nights_avg_ntm                         -0.027083
has_availability                                0.062683
availability_365                                0.051219
number_of_reviews                               0.034356
number_of_reviews_ltm                           0.031559
number_of_reviews_l30d                          0.034701
review_scores_rating                            0.002616
review_scores_accuracy                          0.005462
review_scores_cleanliness                       0.014353
review_scores_checkin                           0.004379
review_scores_communication                     0.004657
review_scores_location                          0.003285
review_scores_value                            -0.000062
calculated_host_listings_count                 -0.039863
calculated_host_listings_count_entire_homes     0.069854
calculated_host_listings_count_private_rooms   -0.132450
calculated_host_listings_count_shared_rooms    -0.067416
reviews_per_month                               0.009692
Name: price, dtype: float64
```

Note. Correlation Coefficient of numeric features with target variable

Parametric Test for Categorical Features

In order to identify the significance of each categorical feature towards the dependent variable, parametric hypothesis test names t-test was performed. In this test, the means of the population are compared and the test statistic would be a measure of difference between the means of these populations. The results of the t-test can be found in Figure 26.

Figure 26

Parametric Test for Categorical Features

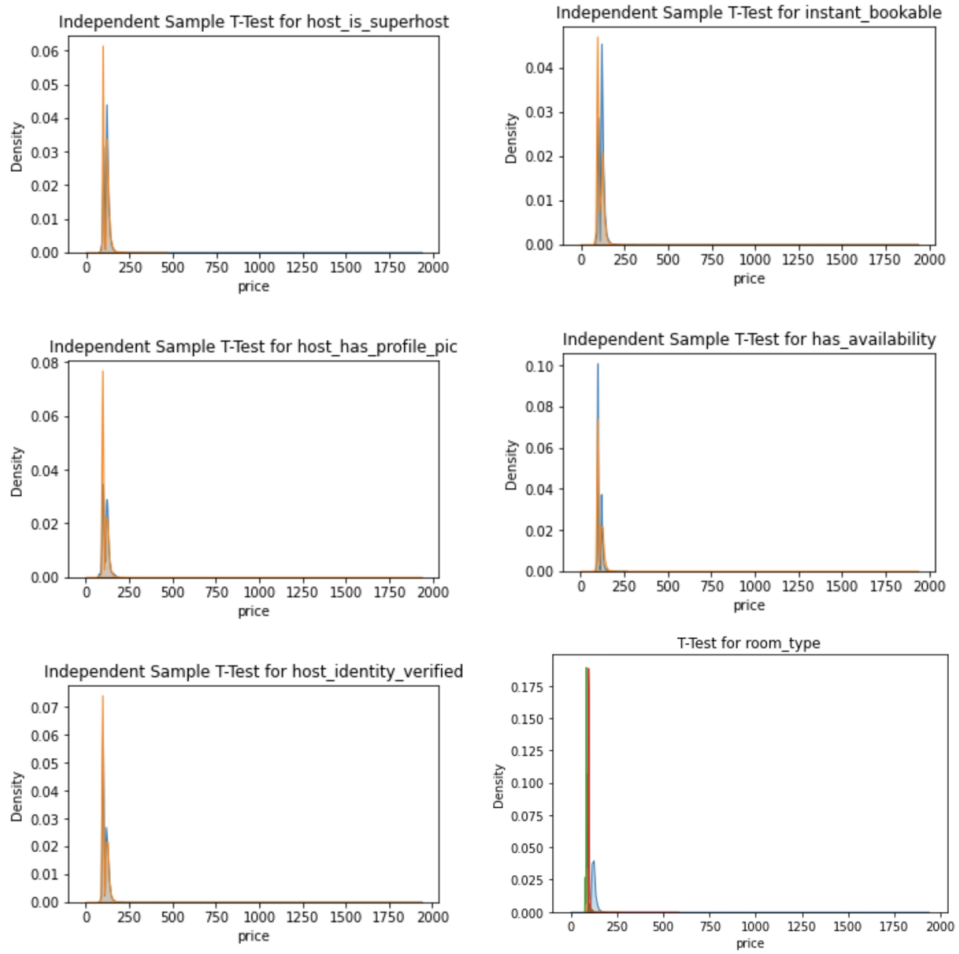
```
T-test statistics for  host_is_superhost :
     Test statistic = -10.977183
     p-value for two tailed test = 0.000000
T-test statistics for  host_has_profile_pic :
     Test statistic = 0.192208
     p-value for two tailed test = 0.847583
T-test statistics for  host_identity_verified :
     Test statistic = -2.435759
     p-value for two tailed test = 0.014869
T-test statistics for  instant_bookable :
     Test statistic = 9.117121
     p-value for two tailed test = 0.000000
T-test statistics for  has_availability :
     Test statistic = -26.377575
     p-value for two tailed test = 0.000000
```

Note. Test statistics of parametric hypothesis test

In addition, the distribution of the t-test statistics were also plotted to visualize the difference in means between the populations. The distributions can be found in the below images.

Figure 27

Parametric Test Distribution Categorical Features



Note. Test statistics distribution of parametric hypothesis test

Non-Parametric Test for Categorical Features

In order to cross-verify the significance identified through the parametric tests, equivalent non-parametric tests were also performed. The median of the population would be compared in the non-parametric tests, which would deem better in cases where the population distribution is skewed. The results of the Wilcoxon signed test can be found in Figure 28.

Figure 28

Non-Parametric Test for Categorical Features

```
wilcoxon-test statistics for  host_is_superhost :
     Test statistic = 18098480.000000
     p-value =  2.1078491466656995e-32
wilcoxon-test statistics for  host_has_profile_pic :
     Test statistic = 212069.000000
     p-value =  0.4820988241037667
wilcoxon-test statistics for  host_identity_verified :
     Test statistic = 20152589.000000
     p-value =  0.0002447643420745337
wilcoxon-test statistics for  instant_bookable :
     Test statistic = 18749444.500000
     p-value =  6.328713876912896e-18
wilcoxon-test statistics for  has_availability :
     Test statistic = 17629271.000000
     p-value =  1.644279379633813e-126
```

Note. Test statistics of non-parametric hypothesis test

Based on the observations from the above mentioned hypothesis tests, the below mentioned table gives a summary of whether the hypotheses considered were rejected or were failed to reject.

Table 2

Comparison of Output Against Hypothesis Testing

S No	Goal	Null Hypothesis (H_0)	Hypothesis Result
1	Identify effects of seasonality and cyclicity on Price of listings	The price of listings do not have a strong pattern of cyclicity and seasonality	Reject the null Hypothesis
2	Identify the effect of host having a superhost tage on the price of listings	The price of Airbnb listings of hosts with superhost tage do not differ significantly from others	Failed to reject the null Hypothesis
3	Understanding the importance of ratings and reviews on the price of listings	The number of reviews and ratings do not have a considerable impact on the price of listings	Failed to reject the null Hypothesis

S No	Goal	Null Hypothesis (H_0)	Hypothesis Result
4	Deduce if having an instant bookable feature impacts the price of listings	Price of listings do not depend on the ability to instantly book the accommodation	Reject the null Hypothesis
5	Discover if host profile attributes such as response time and verification have an impact on price	Host profile attributes do not have a significant impact on the price of listings	Reject the null Hypothesis
6	Determine if host controlled features have more control over price of listings than features that can't be controlled by hosts	Host controlled attributes (amenities, minimum and maximum nights, etc.) do not have more significance on price than the features that cannot be controlled by the host	Reject the null Hypothesis

Note. Output of Hypothesis Test

Abnormal Case Explanation

During the exploratory data analysis, we came across few observations that stood out and seemed abnormal. Firstly, we observed a huge difference in the number of listings that were of type hotel or shared rooms as compared to entire units or private rooms. This is possibly due to the fact that hotels and shared rooms can be booked by the customers directly on a walk-in basis, through their official websites or from other third party websites.

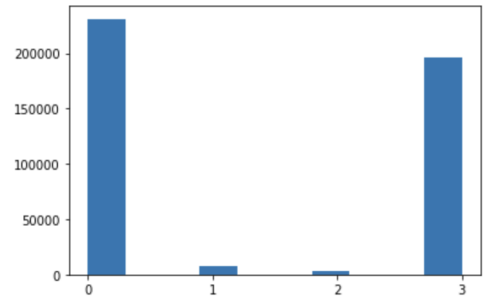
Figure 29

Abnormal case - Room Type Listings

Cmd 17

```
values = [0, 1, 2, 3]
plt.hist(Airbnb_Listings_df['room_type'])
plt.xticks(values)
# Room Type 0 -> Entire unit
# Room Type 1 -> Shared room
# Room Type 2 -> Hotel
# Room Type 3 -> Private room
```

```
Out[16]: ([<matplotlib.axis.XTick at 0x7f2160151d90>,
  <matplotlib.axis.XTick at 0x7f2160151d60>,
  <matplotlib.axis.XTick at 0x7f2160151460>,
  <matplotlib.axis.XTick at 0x7f2160096af0>],
 [Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, '')])
```

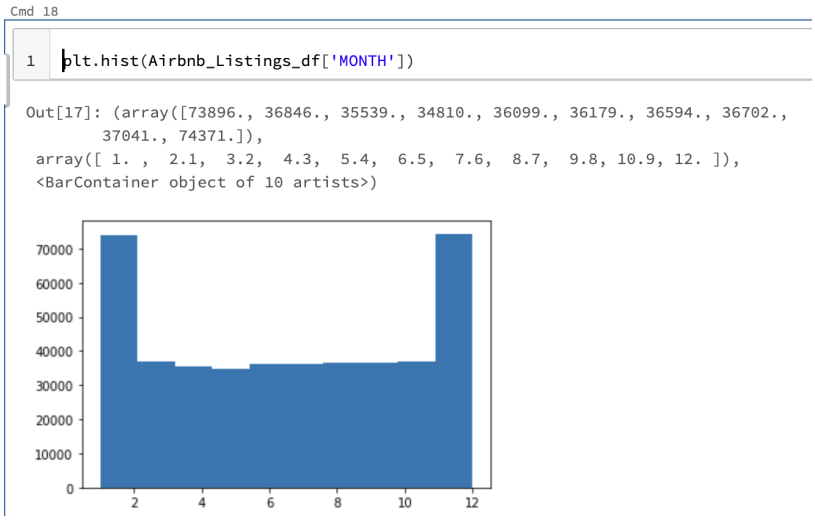


Note. Distribution of number of listings by room type

Another peculiar observation is the number of listings available during different months of the year. While the number of listings are fairly the same during February to November, we can see that the number almost doubles during the months of December and January. This might be because of the end of the year vacation and time off taken by the majority of the public.

Figure 30

Abnormal case - Impact of Month on the Price



Note. Impact of Month on Price

Conclusions & Recommendations

Summary & Conclusions

We have imported the Airbnb listings data of San Francisco for 12 months, cleansed and pre-processed it. We have performed Exploratory Data Analysis to derive insights and identify significant features. We have also used hypothesis testing to further select only the relevant features. Regression models such as Linear Regression, Gradient Boosting Regression, Decision Tree Regression and Random Forest Regression were implemented.

The overall fitting and the performance of the models requires improvement but we could see that the Gradient Boosting algorithm has the lowest RMSE values amongst the models that were implemented. Linear Regression has the highest error value of 10 indicating least performance efficiency compared to the other models.

For the future works, we could increase the scope of our investigation by performing textual analysis on a few columns like neighbourhood, amenities, etc. and test to see if those

columns would have a significant contribution towards increasing the performance of the models when considered as features.

Bibliography

Carrillo, G. (2020, January 29). *Predicting airbnb prices with machine learning and location data.* Medium. Retrieved November 3, 2021, from https://towardsdatascience.com/predicting-airbnb-prices-with-machine-learning-and-location-data-5c1e033d0a5a

Classification and regression. Classification and regression - Spark 3.2.0 Documentation. (n.d.). Retrieved November 3, 2021, from https://spark.apache.org/docs/latest/ml-classification-regression.html

Dhillon, J., Eluri, N. P., Kaur, D., Chhipa, A., Gadupudi, A., Eravi, R. C., & Pirouz, M. (2021, January). *Analysis of Airbnb Prices using Machine Learning Techniques*. In 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0297-0303). IEEE. https://ieeexplore.ieee.org/abstract/document/9376144?casa_token=gWtY49sqT_8AAAAA:JnnkTgXdn5fbRqD_23FL4Vrdjx7qPrCJqrCutIBRLoy8xgUPhMEzEQEIlOMBS_77Ebja0-rQJHw

Frost, J., Gabriel, Tanios, S., Zeb, Adrian, Pathak, D. R. R., Maria, Mohammed, R., L, K., Ashrif, M. N., Craggnon, B., Aruna, Liew, E., Lisa, Heather, Rousso, B. Z., Vivian, Mukhles, Garg, A., … Lucas. (2021, October 5). *Nonparametric tests vs. parametric tests.* Statistics By Jim. Retrieved November 3, 2021, from https://statisticsbyjim.com/hypothesis-testing/nonparametric-parametric-tests/

P Rezazadeh, L.Nikolenko, *Airbnb Price Prediction using Machine Learning and Sentiment*

Analysis, January 2020, https://www.researchgate.net/publication/334783073_Airbnb_Price_Prediction_Using_Machine_Learning_and_Sentiment_Analysis

Regression techniques in machine learning. Analytics Vidhya. (2020, October 18). Retrieved November 3, 2021, from https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/

Sainaghi Ruggero, *Determinants of price and revenue for peer-to-peer hosts. The state of the art* International Journal of Contemporary Hospitality Management, 2020 https://www.researchgate.net/publication/348011416_Determinants_of_price_and_revenue_for_peer-to-peer_hosts_The_state_of_the_art

A. Voltes-Dorta, A. Sanchez-Medina *Drivers of Airbnb prices according to property/room type, season and location: A regression approach.* Journal of Hospitality and Tourism, Vol. 45, 2020 https://www.sciencedirect.com/science/article/pii/S1447677020302023

P. Kalehbasti, L. Nikolenko, and H. Rezaei, *Airbnb price prediction using machine learning and sentiment analysis,* 2019 https://www.researchgate.net/publication/334783073_Airbnb_Price_Prediction_Using_Machine_Learning_and_Sentiment_Analysis

TN Akarsu, P. Foroudi, TC. Melewar *What makes Airbnb likeable? Exploring the nexus between service attractiveness, country image, perceived authenticity and experience from a social exchange theory perspective within an emerging economy context*

International Journal of Hospitality Management, Vol. 91, 2020 https://www.sciencedirect.com/science/article/pii/S0278431920301870

Xu, X., Zeng, S., & He, Y. *The impact of information disclosure on consumer purchase behavior on sharing economy platform Airbnb* International Journal of Production Economics, 2021 https://www.researchgate.net/scientific-contributions/Liubov-Nikolenko-2160610531

R Singh, "Mathematics behind Random Forest and XGBoost", October 2019. https://medium.com/analytics-vidhya/mathematics-behind-random-forest-and-xgboost-ea8596657275

M Schott, "Random Forest Algorithm for Machine Learning", April 2019, https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb

S Saxena, "Gradient Boosting Machines for Data Scientists", March 2021, https://www.analyticsvidhya.com/blog/2021/03/gradient-boosting-machine-for-data-scientists/

S Gawali , "Linear Regression in Machine Learning", June 2021, https://www.analyticsvidhya.com/blog/2021/06/linear-regression-in-machine-learning/

R Agrawal, "Know the best evaluation metrics for your evaluation model", May 2021, https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/

Appendices

Program Source Code

```
#Importing Libraries and setting up SparkContext
from pyspark.sql.functions import *
import pyspark.sql.functions as fn
from pyspark.sql.types import StructType, StructField,
StringType,DateType,DecimalType,IntegerType,ArrayType,LongType,BooleanType,DoubleType
,FloatType
from pyspark.sql import *
import sys
from pyspark import SparkContext, SparkConf
sc=SparkContext.getOrCreate()
import pyspark.sql.types as typ
import pyspark.ml.feature as ft
import pyspark.ml.evaluation as ev
import pyspark.ml.classification as cl
from pyspark.ml import Pipeline
import pyspark.mllib.linalg as ln
import pyspark.mllib.stat as st
from decimal import Decimal
import pandas as pd
import glob
from pyspark.ml.regression import DecisionTreeRegressor
```

```
from pyspark.ml.regression import GBTRegressor
from pyspark.ml.regression import RandomForestRegressor
from pyspark.ml.regression import FMRegressor
from pyspark.ml.regression import LinearRegression
from pyspark.ml.tuning import ParamGridBuilder, TrainValidationSplit
from pyspark.ml.evaluation import RegressionEvaluator
# df = pd.read_csv('/Users/akshayasrinivasan/Desktop/SJSU/Data_228_Big_Data/Term
Project/Data/Airbnb_Listings_Data/listings_Oct_6.csv')
# , header=False, schema = dataset_schema)
path = r'/Users/akshayasrinivasan/Desktop/SJSU/Data_228_Big_Data/Term
Project/Data/Airbnb_Listings_Data' # use your path
all_files = glob.glob(path + "/*.csv")
li = []

for filename in all_files:
    df = pd.read_csv(filename, index_col=None, header=0, na_values=['N/A'])
    li.append(df)

df = pd.concat(li, axis=0, ignore_index=True)
df.dtypes
df=df.replace(to_replace=r'f', value=False, regex=False)
df=df.replace(to_replace=r't', value=True, regex=False)
df.head()
```

```
print(df.columns)
df =
df.drop(['listing_url','scrape_id','description','neighborhood_overview','picture_url','host_url','host
_about','neighbourhood', 'host_thumbnail_url', 'host_picture_url'], axis=1)
print(df.columns)
# spark.conf.set("spark.sql.execution.arrow.enabled", "true")
df.to_csv('/Users/akshayasrinivasan/Desktop/SJSU/Data_228_Big_Data/Term
Project/Data/Airbnb_Data.csv', sep=',')
# Initializing the schema of the dataset
dataset_schema = StructType([
  StructField("id",IntegerType(),True),
  StructField("last_scraped",StringType(),True),
  StructField("name",StringType(),True),
  StructField("host_id",IntegerType(),True),
  StructField("host_name",StringType(),True),
  StructField("host_since",StringType(),True),
  StructField("host_location",StringType(),True),
  StructField("host_response_time",StringType(),True),
  StructField("host_response_rate",StringType(),True),
  StructField("host_acceptance_rate",StringType(),True),
  StructField("host_is_superhost",StringType(),True),
  StructField("host_neighbourhood",StringType(),True),
  StructField("host_listings_count",FloatType(),True),
```

StructField("host_total_listings_count",FloatType(),True),

StructField("host_verifications",StringType(),True),

StructField("host_has_profile_pic",StringType(),True),

StructField("host_identity_verified",StringType(),True),

StructField("neighbourhood_cleansed",StringType(),True),

StructField("neighbourhood_group_cleansed",StringType(),True),

StructField("latitude",FloatType(),True),

StructField("longitude",FloatType(),True),

StructField("property_type",StringType(),True),

StructField("room_type",StringType(),True),

StructField("accommodates",IntegerType(),True),

StructField("bathrooms",FloatType(),True),

StructField("bathrooms_text",StringType(),True),

StructField("bedrooms",FloatType(),True),

StructField("beds",FloatType(),True),

StructField("amenities",StringType(),True),

StructField("price",StringType(),True),

StructField("price_old",StringType(),True),

StructField("minimum_nights",IntegerType(),True),

StructField("maximum_nights",IntegerType(),True),

StructField("minimum_minimum_nights",FloatType(),True),

StructField("maximum_minimum_nights",FloatType(),True),

StructField("minimum_maximum_nights",FloatType(),True),

StructField("maximum_maximum_nights",FloatType(),True),

StructField("minimum_nights_avg_ntm",FloatType(),True),

StructField("maximum_nights_avg_ntm",FloatType(),True),

StructField("calendar_updated",StringType(),True),

StructField("has_availability",BooleanType(),True),

StructField("availability_30",IntegerType(),True),

StructField("availability_60",IntegerType(),True),

StructField("availability_90",IntegerType(),True),

StructField("availability_365",IntegerType(),True),

StructField("calendar_last_scraped",StringType(),True),

StructField("number_of_reviews",IntegerType(),True),

StructField("number_of_reviews_ltm",IntegerType(),True),

StructField("number_of_reviews_l30d",IntegerType(),True),

StructField("first_review",StringType(),True),

StructField("last_review",StringType(),True),

StructField("review_scores_rating",FloatType(),True),

StructField("review_scores_accuracy",FloatType(),True),

StructField("review_scores_cleanliness",FloatType(),True),

StructField("review_scores_checkin",FloatType(),True),

StructField("review_scores_communication",FloatType(),True),

StructField("review_scores_location",FloatType(),True),

StructField("review_scores_value",FloatType(),True),

StructField("license",StringType(),True),

```
  StructField("instant_bookable",StringType(),True),
  StructField("calculated_host_listings_count",IntegerType(),True),
  StructField("calculated_host_listings_count_entire_homes",IntegerType(),True),
  StructField("calculated_host_listings_count_private_rooms",IntegerType(),True),
  StructField("calculated_host_listings_count_shared_rooms",IntegerType(),True),
  StructField("reviews_per_month",FloatType(),True)
])
# Importing the datasets
Airbnb_Listings_df = spark.read.format("csv").option("header",
"true").load("dbfs:/FileStore/shared_uploads/akshaya.srinivasan@sjsu.edu/Airbnb_Merged_Data
-1.csv")
Airbnb_Listings_df.head()
# Viewing the datasets
print("Schema for Airbnb Listings:")
Airbnb_Listings_df.printSchema()

print("Imported Dataset for Airbnb Listings:")
display(Airbnb_Listings_df)
# Data Cleaning - Removing duplicates
# Removing duplicate values
Airbnb_Listings_df = Airbnb_Listings_df.drop_duplicates()
display(Airbnb_Listings_df)
# Data Cleaning - Identifying Missing Value %
```

```
# Identifying the missing value % in each column
Airbnb_Listings_df.agg(*[round(((1 - (fn.count(cnt) / fn.count('*')))*100),3).alias(cnt + 'NULL %') for cnt in Airbnb_Listings_df.columns]).display()

# Dropping columns where majority of the data is missing
Airbnb_Listings_df = Airbnb_Listings_df.drop(Airbnb_Listings_df.license)
Airbnb_Listings_df = Airbnb_Listings_df.drop(Airbnb_Listings_df.bathrooms)
Airbnb_Listings_df = Airbnb_Listings_df.drop(Airbnb_Listings_df.name)
Airbnb_Listings_df = Airbnb_Listings_df.drop(Airbnb_Listings_df.host_name)

# Creating a temporary view of the dataframe
Airbnb_Listings_df.createOrReplaceTempView("Airbnb_Data")
# Data Cleaning - Null handling, Formatting the data types and encoding categorical variables
%sql

CREATE OR REPLACE TEMPORARY VIEW AIRBNB_FORMATTED_DATA AS
(
  SELECT
    id,                                 -- 3831
    INT(SUBSTRING(last_scraped,0,CHARINDEX('/',last_scraped)-1)) AS MONTH,
-- 2021-07-04
    host_id,                            -- 4869
```

```
    CASE WHEN RIGHT(HOST_SINCE,2)>21 THEN 2021-1900-RIGHT(HOST_SINCE,2)
WHEN RIGHT(HOST_SINCE,2)<=21 THEN 2021-2000-RIGHT(HOST_SINCE,2) END AS
HOST_YRS,-- 2008-12-07
    COALESCE(SUBSTRING(host_location,0,CHARINDEX(',',host_location)-1),NULL) AS
host_city,                          -- New York, New York, United States
    COALESCE(RIGHT(host_location,CHARINDEX(',',REVERSE(host_location))-1),NULL)
AS host_country,
    CASE host_response_time
        WHEN 'a few days or more' THEN 0
        WHEN 'within a day' THEN 1
        WHEN 'within a few hours' THEN 2
        WHEN 'within an hour' THEN 3
        ELSE 0
    END AS host_response_time,                  -- within an hour
    COALESCE(INT(TRIM('%' FROM host_response_rate)), (SELECT AVG(INT(TRIM('%'
FROM host_response_rate))) FROM AIRBNB_DATA)) AS host_response_rate,    -- 83%
    COALESCE(INT(TRIM('%' FROM host_acceptance_rate)), (SELECT AVG(INT(TRIM('%'
FROM host_acceptance_rate))) FROM AIRBNB_DATA)) AS host_acceptance_rate,   -- 92%
    CASE host_is_superhost WHEN 'FALSE' THEN 0 ELSE 1 END AS host_is_superhost,
-- false
    coalesce(INT(host_listings_count) ,(select avg(INT(host_listings_count)) from
Airbnb_Data)) as host_listings_count,                    -- 1.0
```

```
    CASE host_has_profile_pic WHEN 'FALSE' THEN 0 ELSE 1 END AS
host_has_profile_pic,                    -- true
    CASE host_identity_verified WHEN 'FALSE' THEN 0 ELSE 1 END AS
host_identity_verified,                  -- true
    coalesce(FLOAT(latitude) ,(select avg(FLOAT(latitude)) from  Airbnb_Data)) as latitude,
-- 40.68494
    coalesce(FLOAT(longitude) ,(select avg(FLOAT(longitude)) from  Airbnb_Data)) as
longitude,                               -- -73.95765
    CASE room_type
      WHEN 'Entire rental unit' THEN 0
      WHEN 'Shared room' THEN 1
      WHEN 'Entire apartment' THEN 0
      WHEN 'Hotel room' THEN 2
      WHEN 'Entire home/apt' THEN 0
      WHEN 'Private room'THEN 3
      WHEN 'Private room in rental unit' THEN 3
      WHEN 'Private room in apartment' THEN 3
      WHEN 'Private room in house' THEN 3
      WHEN 'Private room in residential home' THEN 3
      WHEN 'Entire condominium' THEN 0
      WHEN 'Entire condominium (condo)' THEN 0
      WHEN 'Entire loft' THEN 0
      ELSE 0
```

```
    END AS room_type,                                -- Entire home/apt
    coalesce(INT(accommodates),(select INT(avg(accommodates)) from  Airbnb_Data))  AS
accommodates,                                -- 3
    coalesce(INT(bedrooms),(select INT(avg(bedrooms)) from  Airbnb_Data)) AS bedrooms,
-- 1.0
    coalesce(INT(beds),(select INT(avg(beds)) from  Airbnb_Data)) AS beds,
-- 3.0
    COALESCE((length(`amenities`) - length(replace(`amenities`, ';', '')) + 1),0) AS
amenities_count,
    coalesce(INT(TRIM('$' FROM price)),(select avg(INT(TRIM('$' FROM price))) from
Airbnb_Data)) AS price,                                -- $77.00
    coalesce(INT(minimum_nights) ,(select INT(avg(INT(minimum_nights))) from
Airbnb_Data)) AS minimum_nights,                                -- 1
    coalesce(INT(maximum_nights) ,(select INT(avg(INT(maximum_nights))) from
Airbnb_Data)) AS maximum_nights,                                -- 730
    coalesce(INT(minimum_minimum_nights) ,(select
INT(avg(INT(minimum_minimum_nights))) from  Airbnb_Data)) AS
minimum_minimum_nights,                -- 1.0
    coalesce(INT(maximum_minimum_nights) ,(select
INT(avg(INT(maximum_minimum_nights))) from  Airbnb_Data)) AS
maximum_minimum_nights,                -- 1.0
```

```
    coalesce(INT(minimum_maximum_nights) ,(select
INT(avg(INT(minimum_maximum_nights))) from  Airbnb_Data)) AS
minimum_maximum_nights,           -- 1125.0
    coalesce(INT(maximum_maximum_nights) ,(select
INT(avg(INT(maximum_maximum_nights))) from  Airbnb_Data)) AS
maximum_maximum_nights,           -- 1125.0
    coalesce(INT(minimum_nights_avg_ntm) ,(select
INT(avg(INT(minimum_nights_avg_ntm))) from  Airbnb_Data)) AS
minimum_nights_avg_ntm,           -- 1.0
    coalesce(INT(maximum_nights_avg_ntm) ,(select
INT(avg(INT(maximum_nights_avg_ntm))) from  Airbnb_Data)) AS
maximum_nights_avg_ntm,           -- 1125.0
    CASE has_availability WHEN 'FALSE' THEN 0 ELSE 1 END AS has_availability,
-- true
    coalesce(INT(availability_30) ,(select INT(avg(INT(availability_30))) from  Airbnb_Data))
AS availability_30,                    -- 0
    coalesce(INT(availability_60) ,(select INT(avg(INT(availability_60))) from  Airbnb_Data))
AS availability_60,                    -- 0
    coalesce(INT(availability_90) ,(select INT(avg(INT(availability_90))) from  Airbnb_Data))
AS availability_90,                    -- 1
    coalesce(INT(availability_365) ,(select INT(avg(INT(availability_365))) from
Airbnb_Data)) AS availability_365,                 -- 221
```

```
    coalesce(INT(number_of_reviews) ,(select INT(avg(INT(number_of_reviews))) from
Airbnb_Data)) AS number_of_reviews,                    -- 407
    coalesce(INT(number_of_reviews_ltm) ,(select INT(avg(INT(number_of_reviews_ltm)))
from  Airbnb_Data)) AS number_of_reviews_ltm,          -- 77
    coalesce(INT(number_of_reviews_l30d) ,(select INT(avg(INT(number_of_reviews_l30d)))
from  Airbnb_Data)) AS number_of_reviews_l30d,         -- 0
    coalesce(FLOAT(review_scores_rating) ,(select avg(FLOAT(review_scores_rating)) from
Airbnb_Data)) AS review_scores_rating,                 -- 4.45
    coalesce(FLOAT(review_scores_accuracy) ,(select avg(FLOAT(review_scores_accuracy))
from  Airbnb_Data)) AS review_scores_accuracy,         -- 4.59
    coalesce(FLOAT(review_scores_cleanliness) ,(select
avg(FLOAT(review_scores_cleanliness)) from  Airbnb_Data)) AS review_scores_cleanliness,
-- 4.5
    coalesce(FLOAT(review_scores_checkin) ,(select avg(FLOAT(review_scores_checkin)) from
Airbnb_Data)) AS review_scores_checkin,                -- 4.79
    coalesce(FLOAT(review_scores_communication) ,(select
avg(FLOAT(review_scores_communication)) from  Airbnb_Data)) AS
review_scores_communication,       -- 4.81
    coalesce(FLOAT(review_scores_location) ,(select avg(FLOAT(review_scores_location))
from  Airbnb_Data)) AS review_scores_location,         -- 4.72
    coalesce(FLOAT(review_scores_value) ,(select avg(FLOAT(review_scores_value)) from
Airbnb_Data)) AS review_scores_value,                  -- 4.64
```

```
    CASE instant_bookable WHEN 'FALSE' THEN 0 ELSE 1 END AS instant_bookable,
-- false
    coalesce(INT(calculated_host_listings_count) ,(select
INT(avg(INT(calculated_host_listings_count))) from  Airbnb_Data)) AS
calculated_host_listings_count,  -- 1
    coalesce(INT(calculated_host_listings_count_entire_homes) ,(select
INT(avg(INT(calculated_host_listings_count_entire_homes))) from  Airbnb_Data)) AS
calculated_host_listings_count_entire_homes, -- 1
    coalesce(INT(calculated_host_listings_count_private_rooms) ,(select
INT(avg(INT(calculated_host_listings_count_private_rooms))) from  Airbnb_Data)) AS
calculated_host_listings_count_private_rooms, -- 0
    coalesce(INT(calculated_host_listings_count_shared_rooms) ,(select
INT(avg(INT(calculated_host_listings_count_shared_rooms))) from  Airbnb_Data)) AS
calculated_host_listings_count_shared_rooms,  -- 0
    coalesce(FLOAT(reviews_per_month) ,(select avg(FLOAT(reviews_per_month)) from
Airbnb_Data)) AS reviews_per_month                    -- 5.15
FROM
   AIRBNB_DATA
  WHERE
COALESCE(SUBSTRING(host_location,0,CHARINDEX(',',host_location)-1),NULL) IS NOT
NULL AND host_location not like '%住过英国伦敦，%法国巴黎，美国芝加哥，墨西哥
%cancun%' AND availability_30 <=30 AND coalesce(INT(TRIM('$' FROM price)),(select
avg(INT(TRIM('$' FROM price))) from  Airbnb_Data)) != 0
```

```
);
SELECT * FROM AIRBNB_FORMATTED_DATA LIMIT 20;
# Creating a dataframe for the updated view
Airbnb_Listings_Selected_Features_df=spark.sql("SELECT * FROM
AIRBNB_FORMATTED_DATA")

# Re-checking if there are no more missing values
Airbnb_Listings_Selected_Features_df.agg(*[round(((1 - (fn.count(cnt) /
fn.count('*')))*100),3).alias(cnt + '_NULL %') for cnt in
Airbnb_Listings_Selected_Features_df.columns]).display()

# Removing duplicate values
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop_duplicates()
display(Airbnb_Listings_Selected_Features_df)
# Identifying Categorical and Numeric features
categorical_features = ['id', 'host_id', 'host_city', 'host_country', 'host_response_time',
'room_type', 'host_is_superhost', 'host_has_profile_pic', 'host_identity_verified', 'availability_30',
'availability_60', 'availability_90', 'instant_bookable']
print("Categorical Columns: \n", categorical_features)
numerical_datatype_columns = [item for item in Airbnb_Listings_Selected_Features_df.columns
if item not in categorical_features ]
print("Numeric Data Type Columns: \n", numerical_datatype_columns)
```

```
columns = Airbnb_Listings_Selected_Features_df.columns
print("All Column Names: \n",columns)
non_numeric_cols = ['id', 'host_id', 'host_city', 'host_country']
numerical_features = [item for item in Airbnb_Listings_Selected_Features_df.columns if item
not in categorical_features ]
# Univariate Analysis - Viewing descriptive statistics
#Summary stats of all columns
display(Airbnb_Listings_Selected_Features_df.describe())
display(Airbnb_Listings_Selected_Features_df.summary())

# Distribution of the target variable
display(Airbnb_Listings_Selected_Features_df.agg({'price': 'skewness'}))
display(Airbnb_Listings_Selected_Features_df.agg({'price': 'kurtosis'}))
# Univariate Analysis - Frequencies of Categorical features
# Basic counts/frequencies for categorical columns

display(spark.sql("SELECT COUNT(DISTINCT ROOM_TYPE) AS
DISTINCT_ROOM_TYPES, COUNT(DISTINCT host_response_time) AS
DISTINCT_HOST_RESPONSE_TIME, COUNT(DISTINCT host_has_profile_pic) AS
DISTINCT_HOST_HAS_PROFILE_PIC, COUNT(DISTINCT host_is_superhost) AS
DISTINCT_HOST_IS_SUPERHOST FROM AIRBNB_FORMATTED_DATA"))
```

```
display(spark.sql("SELECT ROOM_TYPE, COUNT(*) AS
FREQUENCY_OF_ROOM_TYPES FROM AIRBNB_FORMATTED_DATA GROUP BY 1
SORT BY 1 DESC"))
display(spark.sql("SELECT host_has_profile_pic, COUNT(*) AS
FREQUENCY_OF_HOST_HAS_PROFILE_PICS FROM AIRBNB_FORMATTED_DATA
GROUP BY 1 SORT BY 1 DESC"))
display(spark.sql("SELECT host_response_time, COUNT(*) AS
FREQUENCY_OF_HOST_RESPONSE_TIMES FROM AIRBNB_FORMATTED_DATA
GROUP BY 1 SORT BY 1 DESC"))
display(spark.sql("SELECT host_is_superhost, COUNT(*) AS
FREQUENCY_OF_HOST_IS_SUPERHOST FROM AIRBNB_FORMATTED_DATA GROUP
BY 1 SORT BY 1 DESC"))
# Correlation Matrix for numeric features
numeric_data = Airbnb_Listings_Selected_Features_df.select(numerical_features).toPandas()
corr_result = numeric_data.corr()
display(corr_result)
print("Correlation of each independent variable with the dependant variable Price: \n")
display(corr_result['price'])
# Removing Highly Correlated Independent Variables
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.review_scor
es_rating)
```

```
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.review_scor
es_cleanliness)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.review_scor
es_checkin)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.review_scor
es_communication)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.review_scor
es_location)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.review_scor
es_value)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.minimum_
minimum_nights)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.maximum_
minimum_nights)
```

```
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.minimum_
maximum_nights)
Airbnb_Listings_Selected_Features_df =
Airbnb_Listings_Selected_Features_df.drop(Airbnb_Listings_Selected_Features_df.maximum_
maximum_nights)

import seaborn as sns
import matplotlib.pyplot as plt
Airbnb_Listings_df = Airbnb_Listings_Selected_Features_df.toPandas()
binary_features_df = Airbnb_Listings_Selected_Features_df.select('host_is_superhost',
'host_has_profile_pic', 'host_identity_verified', 'instant_bookable', 'host_response_time',
'room_type', 'accommodates', 'bedrooms', 'beds', 'has_availability').toPandas()
# Exploratory Data Analysis
sns.barplot(x='host_identity_verified', y="price", hue ='host_has_profile_pic',
data=Airbnb_Listings_df)
sns.barplot(x='room_type', y="price", hue ='has_availability', data=Airbnb_Listings_df)
# Room Type 0 -> Entire unit
# Room Type 1 -> Shared room
# Room Type 2 -> Hotel
# Room Type 3 -> Private room
values = [0, 1, 2, 3]
plt.hist(Airbnb_Listings_df['room_type'])
```

```
plt.xticks(values)

plt.hist(Airbnb_Listings_df['MONTH'])

sns.lineplot(x='MONTH', y="host_response_time", hue ='instant_bookable',

data=Airbnb_Listings_df)

sns.lineplot(x='MONTH', y="price", hue ='room_type', data=Airbnb_Listings_df)

fig, ax1 = plt.subplots(figsize=(10,6))

color = 'tab:green'

#bar plot creation

ax1.set_title('bar title', fontsize=16)

ax1.set_xlabel('bar x label', fontsize=16)

ax1.set_ylabel('bar y label', fontsize=16)

ax1 = sns.barplot(x='MONTH', y='price', data = Airbnb_Listings_df, palette='summer')

ax1.tick_params(axis='y')

#specify we want to share the same x-axis

ax2 = ax1.twinx()

color = 'tab:red'

#line plot creation

ax2.set_ylabel('Avg Percipitation %', fontsize=16)

ax2 = sns.lineplot(x='MONTH', y='minimum_nights', data = Airbnb_Listings_df, color=color)

ax2.tick_params(axis='y', color=color)

#show plot

plt.show()

# EDA - Variation with Target for Categorical Features
```

```
i=1
for x in (binary_features_df.columns):
  if x!= "price":
#     plt.subplot(i,1,i)
    plt.figure(figsize=(5,3))
    sns.barplot(x=x, y="price", data=Airbnb_Listings_df)
    x_name = x
    y_name = 'price'
    plt.xlabel(x_name)
    plt.ylabel(y_name)
    plt.title('Variation of {x_name} with Price'.format(x_name=x_name))
    i+=1
continuous_features = Airbnb_Listings_Selected_Features_df.select('availability_30',
'availability_60', 'availability_90', 'MONTH', 'HOST_YRS', 'host_response_rate',
'host_acceptance_rate', 'host_listings_count',  'latitude', 'longitude', 'amenities_count',
'minimum_nights', 'maximum_nights', 'minimum_nights_avg_ntm', 'maximum_nights_avg_ntm',
'availability_365', 'number_of_reviews', 'number_of_reviews_ltm', 'number_of_reviews_l30d',
'review_scores_accuracy', 'calculated_host_listings_count',
'calculated_host_listings_count_entire_homes', 'calculated_host_listings_count_private_rooms',
'calculated_host_listings_count_shared_rooms', 'reviews_per_month').toPandas()
# EDA - Variation with Target for Numeric Features
i=1
for x in (continuous_features.columns):
```

```
  if x!= "price":
    plt.figure(figsize=(5,3))
    sns.lineplot(x=x, y="price", data=Airbnb_Listings_df)
    x_name = x
    y_name = 'price'
    plt.xlabel(x_name)
    plt.ylabel(y_name)
    plt.title('Variation of {x_name} with Price'.format(x_name=x_name))
    i+=1
# Hypothesis testing - Parametric Test for Categorical Features
from scipy.stats import ttest_ind
binary_features = binary_features_df[['host_is_superhost', 'host_has_profile_pic',
'host_identity_verified', 'instant_bookable', 'has_availability']]
non_binary_categorical_features = binary_features_df[['host_response_time', 'room_type',
'accommodates', 'bedrooms', 'beds']]
for i in binary_features.columns:
  t_value,p_value = ttest_ind(Airbnb_Listings_df['price'][Airbnb_Listings_df[i] ==
0].iloc[:10000], Airbnb_Listings_df['price'][Airbnb_Listings_df[i] == 1].iloc[:10000])
  print("T-test statistics for ", i, ":")
  print('     Test statistic = %f'%float("{:.6f}".format(t_value)))
  print('     p-value for two tailed test = %f'%p_value)
# Hypothesis testing - Viewing the test distributions
for i in binary_features.columns:
```

```
  plt.figure(figsize=(5,3))
  sns.kdeplot(Airbnb_Listings_df['price'][Airbnb_Listings_df[i] == 0], shade=True)
  sns.kdeplot(Airbnb_Listings_df['price'][Airbnb_Listings_df[i] == 1], shade=True)
  plt.title(f"Independent Sample T-Test for {i}")
from scipy import stats as stat
t_value,p_value = stat.ttest_rel(Airbnb_Listings_df['price'], Airbnb_Listings_df['room_type'])
print("T-test statistics for ", 'room_type', ":")
print('     Test statistic = %f'%float("{:.6f}".format(t_value)))
print('     p-value for two tailed test = %f'%p_value)
sns.kdeplot(Airbnb_Listings_df['price'][Airbnb_Listings_df['room_type'] == 0], shade=True)
sns.kdeplot(Airbnb_Listings_df['price'][Airbnb_Listings_df['room_type'] == 1], shade=True)
sns.kdeplot(Airbnb_Listings_df['price'][Airbnb_Listings_df['room_type'] == 2], shade=True)
sns.kdeplot(Airbnb_Listings_df['price'][Airbnb_Listings_df['room_type'] == 3], shade=True)
plt.title("T-Test for room_type")
# Hypothesis testing - Non-Parametric Test for Categorical Features
import scipy
for i in binary_features.columns:
  if i != 'host_has_profile_pic':
    stat_value,p_value = scipy.stats.wilcoxon(Airbnb_Listings_df['price'][Airbnb_Listings_df[i]
== 0].iloc[:10000], Airbnb_Listings_df['price'][Airbnb_Listings_df[i] == 1].iloc[:10000])
    print("wilcoxon-test statistics for ", i, ":")
    print('     Test statistic = %f'%float("{:.6f}".format(stat_value)))
    print('     p-value = ', p_value)
```

```
  elif i=='host_has_profile_pic':
    stat_value,p_value = scipy.stats.wilcoxon(Airbnb_Listings_df['price'][Airbnb_Listings_df[i]
== 0].iloc[:1000], Airbnb_Listings_df['price'][Airbnb_Listings_df[i] == 1].iloc[:1000])
    print("wilcoxon-test statistics for ", i, ":")
    print('     Test statistic = %f'%float("{:.6f}".format(stat_value)))
    print('     p-value = ', p_value)
# Selecting the Significant Features
Airbnb_Listings_Significant_Features_df =
Airbnb_Listings_Selected_Features_df.select('MONTH','accommodates', 'bedrooms', 'beds',
'amenities_count', 'has_availability', 'room_type', 'price')
# Dimensionality Reduction
features=Airbnb_Listings_Significant_Features_df.drop('price')
print("Selected Features: ")
print(features.columns)

assembler=ft.VectorAssembler(inputCols=features.columns, outputCol='feature')
output=assembler.transform(Airbnb_Listings_Significant_Features_df)

data=output.select("feature","price")
data.show(3)
# Splitting into train and test
train_df, test_df = data.randomSplit([0.7, 0.3])
# Gradient Boost Regressor
```

```
gbr=GBTRegressor(featuresCol='feature', labelCol="price")
gbr_model = gbr.fit(train_df)
#Prediction on test data
predictions = gbr_model.transform(test_df)
predictions.select("prediction","price","feature").show()
pred_evaluator_GBT =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="r2")
print("R2 on test data:", pred_evaluator_GBT.evaluate(predictions))

pred_evaluator_GBT =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="rmse")
print("RMSE on test data:", pred_evaluator_GBT.evaluate(predictions))
# Decision Tree Regression
dtr=DecisionTreeRegressor(featuresCol='feature', labelCol="price")
dtr_model = dtr.fit(train_df)

#Prediction on test data
predictions = dtr_model.transform(test_df)
predictions.select("prediction","price","feature").show()
pred_evaluator_DT =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="r2")
print("R2 on test data:", pred_evaluator_DT.evaluate(predictions))
```

```
pred_evaluator_DT =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="rmse")
print("RMSE on test data:", pred_evaluator_DT.evaluate(predictions))
# Random Forest Regression
rfg=RandomForestRegressor(featuresCol='feature', labelCol="price")
rfg_model = rfg.fit(train_df)

#Prediction on test data
predictions = rfg_model.transform(test_df)
predictions.select("prediction","price","feature").show()
pred_evaluator_RF =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="r2")
print("R2 on test data:", pred_evaluator_RF.evaluate(predictions))

pred_evaluator_RF =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="rmse")
print("RMSE on test data:", pred_evaluator_RF.evaluate(predictions))
# Linear Regression
lr=LinearRegression(featuresCol='feature', labelCol="price")
lr_model = lr.fit(train_df)

print("Coefficients :", lr_model.coefficients)
print("Intercept :", lr_model.intercept)
```

```
train_summary = lr_model.summary
print("Training RMSE:",train_summary.rootMeanSquaredError)
print("Training r2 Score:", train_summary.r2)

#Prediction on test data
predictions = lr_model.transform(test_df)
predictions.select("prediction","price","feature").show()
pred_evaluator_LR =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="r2")
print("R2 on test data:", pred_evaluator_LR.evaluate(predictions))

pred_evaluator_LR =
RegressionEvaluator(predictionCol="prediction",labelCol="price",metricName="rmse")
print("RMSE on test data:", pred_evaluator_LR.evaluate(predictions))
```