FLIGHT PRICE- PREDICTION PROJECT

Submitted by:

Durgadhar Pathak

Internship 15

ACKNOWLEDGMENT

The internship opportunity I have with Flip Robo Technologies is a great chance for learning and professional development. I am also grateful to our SME Mr. Sajid Choudhary and Ms. Sapna Verma for their valuable and constructive suggestions during the planning and development of this project. Their quick support and references helped a lot in building this project. Also, I am very thankful to our SMEs & Flip Robo Team for understanding technical issue faced by me and provide quick resolution & providing enough time for submission.

Also, I am thankful to DT support Team for their continuous effort to resolve our queries during project building.

Research papers that helped me in this project was as follows: -

1- https://www.researchgate.net/publication/335936877_A_Framework_for_Airfare_Price_Prediction_A_Machine_Learning_Approach
2- https://www.sciencedirect.com/science/article/pii/S131915781830884X
3- https://ieeexplore.ieee.org/document/8081365

Articles that helped me in this project was as follows:

1- https://www.analyticsvidhya.com/blog/2021/06/flight-price-prediction-a-regression-analysis-using-lazy-prediction/
2- https://medium.com/analytics-vidhya/regression-flight-price-prediction-6771fc4d1fb3
3- https://www.ijert.org/a-survey-on-flight-pricing-prediction-using-machine-learning

References:

1- https://machinelearningmastery.com/
2- https://scikit-learn.org/stable/
3- https://www.geeksforgeeks.org/machine-learning/
4- https://pandas.pydata.org/
5- https://www.datacamp.com/
6- https://www.ibm.com/cloud/learn/machine-learning
7- https://www.selenium.dev/selenium/docs/api/py/common/selenium.common.exceptions.html
8- https://www.oreilly.com/library/view/web-scraping-with/9781491985564/

INTRODUCTION

• Business Problem Framing

Anyone who has booked a flight ticket knows how unexpectedly the prices vary. The cheapest available ticket on a given flight gets more and less expensive over time. This usually happens as an attempt to maximize revenue based on –

1. Time of purchase patterns (making sure last-minute purchases are expensive)
2. Keeping the flight as full as they want it (raising prices on a flight which is filling up in order to reduce sales and hold back inventory for those expensive last-minute expensive purchases)

So, we have to work on a project where we collect data of flight fares with other features and work to make a model to predict fares of flights.

• Conceptual Background of the Domain Problem

This project contains three phases: -

1. **Data Collection Phase**: -WE have to scrape at least 1500 rows of data. We can scrape more data as well, it's up to us, More the data better the model. In this section we have to scrape the data of flights from different websites. The number of columns for data doesn't have limit, it's up to us and our creativity. Generally, these columns are airline name, date of journey, source, destination, route, departure time, arrival time, duration, total stops and the target variable price. We can make changes to it, we can add or we can remove some columns, it completely depends on the website from which we are fetching the data.

2. **Data Analysis Phase**: -After cleaning the data, we have to do some analysis on the data. Do airfares change frequently? Do they move in small increments or in large jumps? Do they tend to go up or down over time? What is the best time to buy so that the consumer can save the most by taking the least risk? Does price increase as we get near to departure date? Is Indigo cheaper than Jet Airways? Are morning flights expensive?

3. **Model Building Phase**: -After collecting the data, we need to build a machine learning model. Before model building do all data pre-processing steps. Try different models with different hyper parameters and select the best model.

   Follow the complete life cycle of data science. Include all the steps like.

   1. Data Cleaning
   2. Exploratory Data Analysis
   3. Data Pre-processing
   4. Model Building
   5. Model Evaluation
   6. Selecting the best model

- # Review of Literature

First, we need to collect flight price data from different websites using web scraping techniques and then need to build a machine learning model. Machine learning algorithms enable the creation of a new model using existing anonymized historical data that would be used to train the model to make better predictions not only for car prices, but also for other variables like Airline, Journey date, Departure Time etc. With use of good model, Airlines could predict the price easily. To mitigate the subjective part of the decision-making process, different scoring models are introduced to evaluate certain parameters that could affect the flight prices.

Models used: -

1- **Random Forest Regression**: - Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees.

2- **AdaBoost Regression**: - An AdaBoost regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction.

3- **k-nearest neighbors**: - K nearest neighbors is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure (e.g., distance functions). A simple implementation of KNN regression is to calculate the average of the numerical target of the K nearest neighbors.

Other models used are: - Linear Regression, Decision Tree, Ridge & Lasso regression.

Hyper Parameter tuning: - For Lasso regression, k-nearest neighbors & AdaBoost Regression algorithms, we used Grid search cross-validation technique to choose the best hyper-parameters & for implementing best model using Random Forest Regression algorithm, we have used Randomized Search CV to find best hyperparameters.

Evaluation Matrix: - Coefficient of determination (R2 score), Cross Validation Score, Mean Absolute Error, Mean Squared Error & Root Mean Squared Error.

- # Motivation for the Problem Undertaken

Flight Price Prediction Project help tourists to find the right flight price based on their needs and also it gives various options and flexibility for travelling. Different features (airline, source, destination, departure & arrival timings, Journey date etc.) helps to understand the flight price variations. Using it airlines also get benefit & required passengers. Also they will get benefit in scheduling also.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

In this project, input data is provided to the model along with the output data so it is a type of supervised learning. Also output Variable “Price" is continuous in nature so it is a Regression based problem and We have to predict the price of cars with available independent variables. We have performed regression tasks and it models a target prediction value based on independent variables and is mostly used for finding out the relationship between variables and forecasting. Data exploration is the first step in data analysis and typically involves summarizing the main characteristics of a data set, including its size, accuracy, initial patterns in the data and other attributes. We have checked statistical summary, correlation matrix, skewness, missing values & outliers in dataset and try to handle them very carefully.

Statistical Summary: summary statistics is used to summarize set of observations, in order to communicate the largest amount of information as simply as possible. It includes central Tendency, dispersion, skewness, variance, range, deviation etc.

	count	mean	std	min	25%	50%	75%	max
Airline	2295.0	15.334641	6.893732	0.0	12.0	18.0	18.0	35.0
Journey_date	2295.0	6.795207	4.749329	0.0	2.0	6.0	11.0	16.0
From	2295.0	0.967756	0.565506	0.0	1.0	1.0	1.0	3.0
To	2295.0	3.230501	2.066154	0.0	2.0	3.0	5.0	8.0
Dtime	2295.0	69.849673	35.430133	0.0	37.0	69.0	96.0	146.0
Atime	2295.0	106.884967	46.441335	0.0	76.0	111.0	144.0	179.0
Stops	2295.0	12.832680	14.319613	0.0	3.0	8.0	10.0	42.0
Price	2295.0	8188.997821	3558.418345	3635.0	5316.0	7622.0	9639.0	30832.0

Observations:

1- For input features, Arrival Time has highest standard deviation of 46.44.

2- Maximum Price of flight is 30832.0 and minimum price is 3635.0.

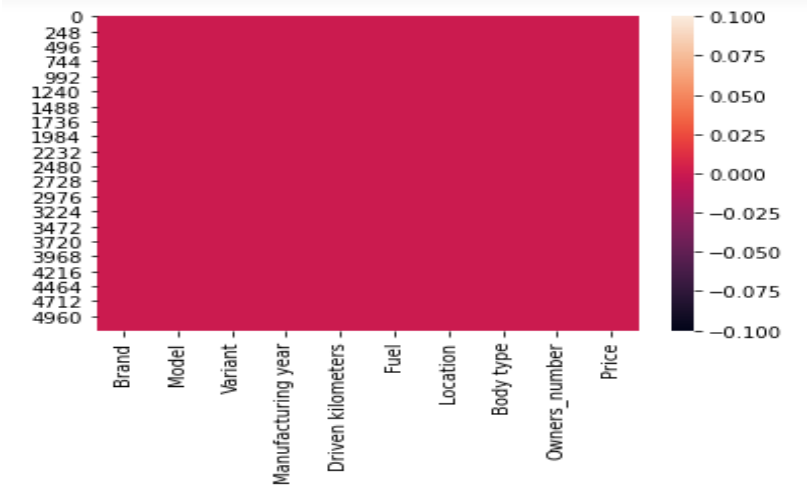
3- In input features Journey Date, Destination, Departure Timing & Stops, the value of mean is considerably greater than median so there are strong chances of positive skewness.

4- In remaining input columns, value of median is greater than mean so the columns are negatively skewed.

Correlation: After seeing many correlated values we can say that many columns have correlation values and dropping some of these will be better for our dataset.

Skewness: If the skewness is between -0.5 and 0.5, then dataset is fairly symmetrical and symmetrical distribution will have a skewness of zero. So accordingly, we are removing skewness using NumPy mathematical function cube-root transform.

Missing Data: -



Dataset has no missing values.

- Data Sources and their formats

First, we need to collect flight price data from different websites using web scraping techniques and then need to build a machine learning model. So, we have scraped flight price data from website: - www.makemytrip.com using Selenium web scraping methods. We have scraped following data: -

Dataset: - Different columns are as: -

1- Airline

2-Journey_date

3-Source (From)

4-Destination (To)

5-Departure Time (D time)

6-Arrival Time (A time)

7-Stops

8-Price

We have combined data from this website into a csv file and used it for flight price prediction project. Total dataset has 2295 rows and 8 columns. We have also tried to equalize the dataset format for better understanding and use.

Target Variable: - Target Variable is Price in this project and it is continuous in nature so we will use Regression algorithms to make our model.

- # Data Pre-processing Done

The data pipeline starts with collecting the data and ends with communicating the results & Data pre-processing is a data mining technique which is used to transform the raw data in a useful and efficient format. It involves 4 steps: - Cleaning, Formatting, Scaling, and Normalization. While building a machine learning model, if we haven't done any pre-processing like correcting outliers, handling different formats, normalization and scaling of data, or feature engineering, we might end up considering those 1% of results that are false. Some steps performed in this project are: -

- Column wise Empty cell analysis done & found no missing values.
- Checked unique values in each column to explore dataset more deeply.
- We had seen outliers in some columns so we were trying to remove them using Z scores and data loss was only 4.5%.
- Skewness was present in dataset. If the skewness is between -0.5 and 0.5, then dataset is fairly symmetrical and symmetrical distribution will have a skewness of zero. So accordingly, we removed skewness using NumPy mathematical function cube root transform.

- **Feature Encoding and Normalization**: - Features came in a variety of format, e.g., integers, floating numbers, string values, etc. So, we Checked Concise Summary of our Data Frame and we have noticed that 7 columns have object (str or mix str) data type. This was a challenge for us as these features cannot be directly used for training. To prevent regression biases towards certain features, we dropped some of the irrelevant features and make sure that we haven't lose important data. In the end, we also normalize the feature values so that all features are evaluated on the same scale.

- **Correlation:** - We checked the linear relationship between two variables. After seeing these correlated values, we had found that many columns had multicollinearity is also present between various columns. We have checked correlation between input variables and output variable "Price": -

Observations:

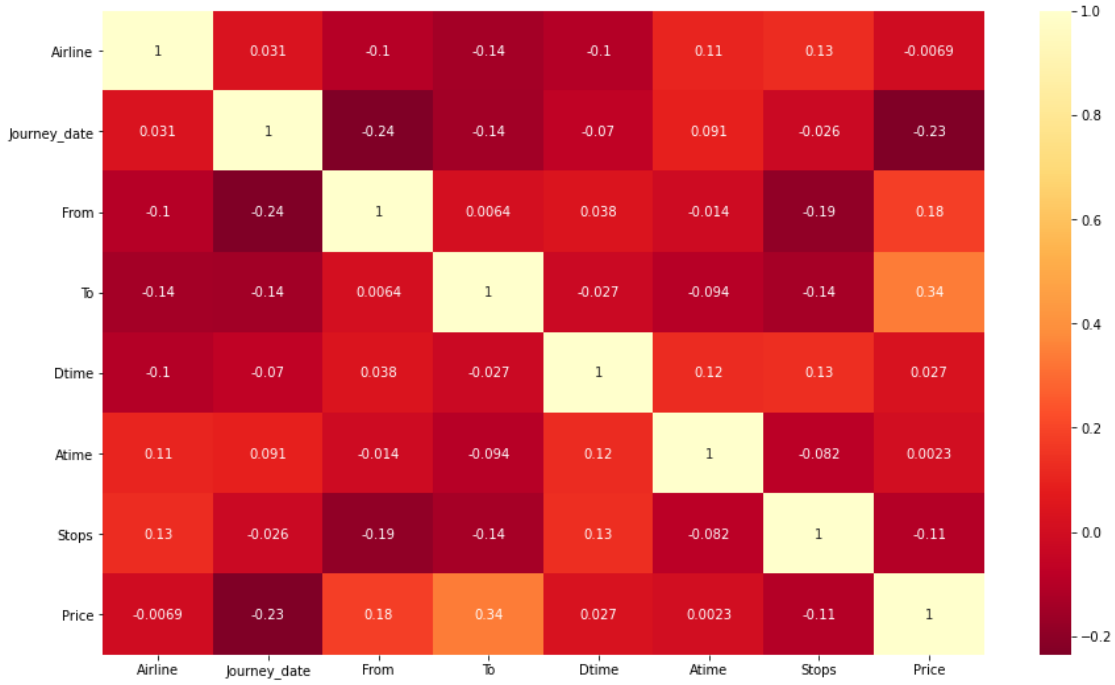
1- Destination column is most positively correlated with Price column.

2- Journey Date column is most negatively correlated with Price column.

Correlation Matrix: - A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between two variables. A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses. A correlation matrix consists of rows and columns that show the variables.

	Airline	Journey_date	From	To	Dtime	Atime	Stops	Price
Airline	1.000000	0.031479	-0.100328	-0.140966	-0.101788	0.106182	0.131230	-0.006864
Journey_date	0.031479	1.000000	-0.235533	-0.144628	-0.070290	0.090969	-0.026124	-0.231090
From	-0.100328	-0.235533	1.000000	0.006364	0.037571	-0.013603	-0.188970	0.181667
To	-0.140966	-0.144628	0.006364	1.000000	-0.027008	-0.093840	-0.135926	0.343561
Dtime	-0.101788	-0.070290	0.037571	-0.027008	1.000000	0.122771	0.132112	0.027242
Atime	0.106182	0.090969	-0.013603	-0.093840	0.122771	1.000000	-0.081600	0.002319
Stops	0.131230	-0.026124	-0.188970	-0.135926	0.132112	-0.081600	1.000000	-0.105198
Price	-0.006864	-0.231090	0.181667	0.343561	0.027242	0.002319	-0.105198	1.000000

Checking correlation using Heatmap with annotations: -



Observations: -

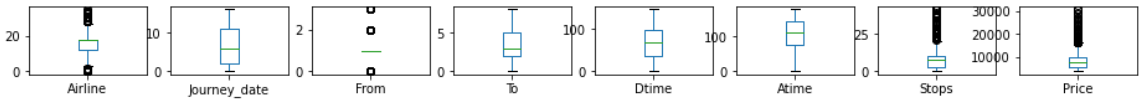
1- Price is highly correlated with Source & Destination columns.

2- Price is negatively correlated with journey Date & Stops column.

Plotting Outliers: - An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. Outliers should be investigated carefully. So, we will use graphical techniques Box Plot for identifying outliers.

Box Plot- The box plot is a useful graphical display for describing the behaviour of the data in the middle as well as at the ends of the distributions. The box plot uses the median and the lower and upper quartiles (defined as the 25th and 75th percentiles). A box plot is constructed by drawing a box between the upper and lower quartiles with a solid line drawn across the box to locate the median.

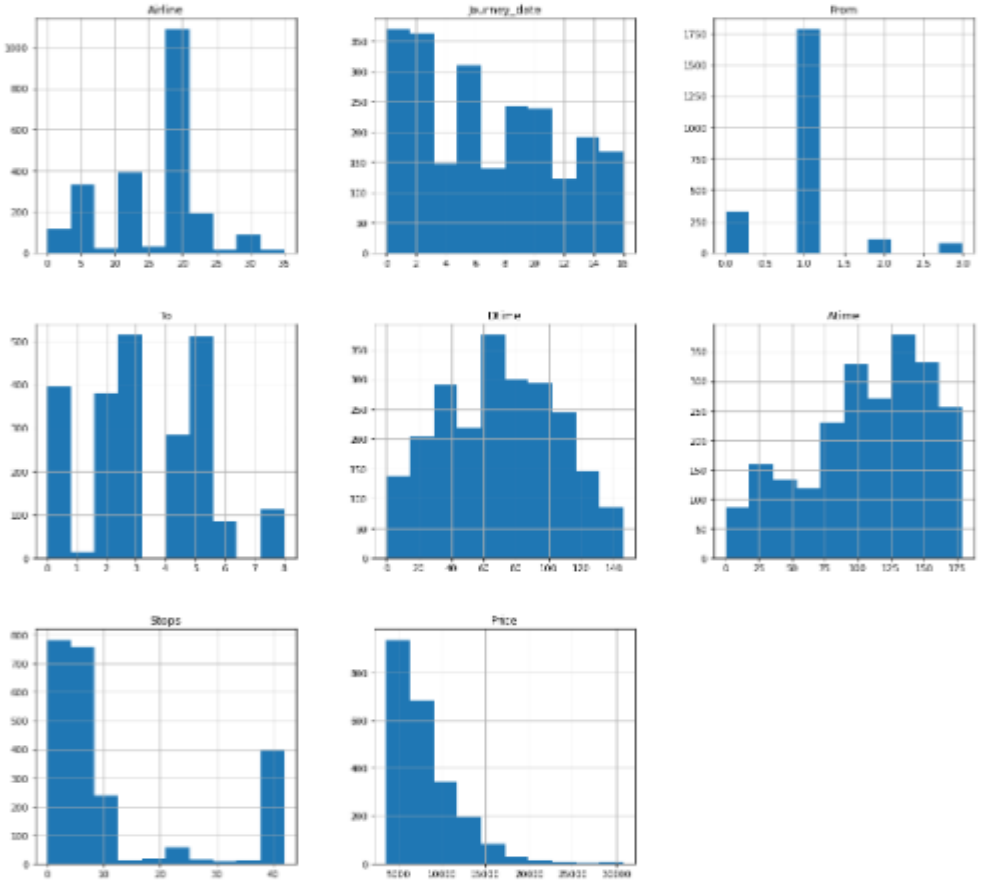
Univariate Analysis: - Univariate involves the analysis of a single variable. First, we are going to do univariate analysis using Box Plot method.



Observation:

Outliers are present in various columns.

Histogram: - we are creating histograms to get broader idea of the distribution.

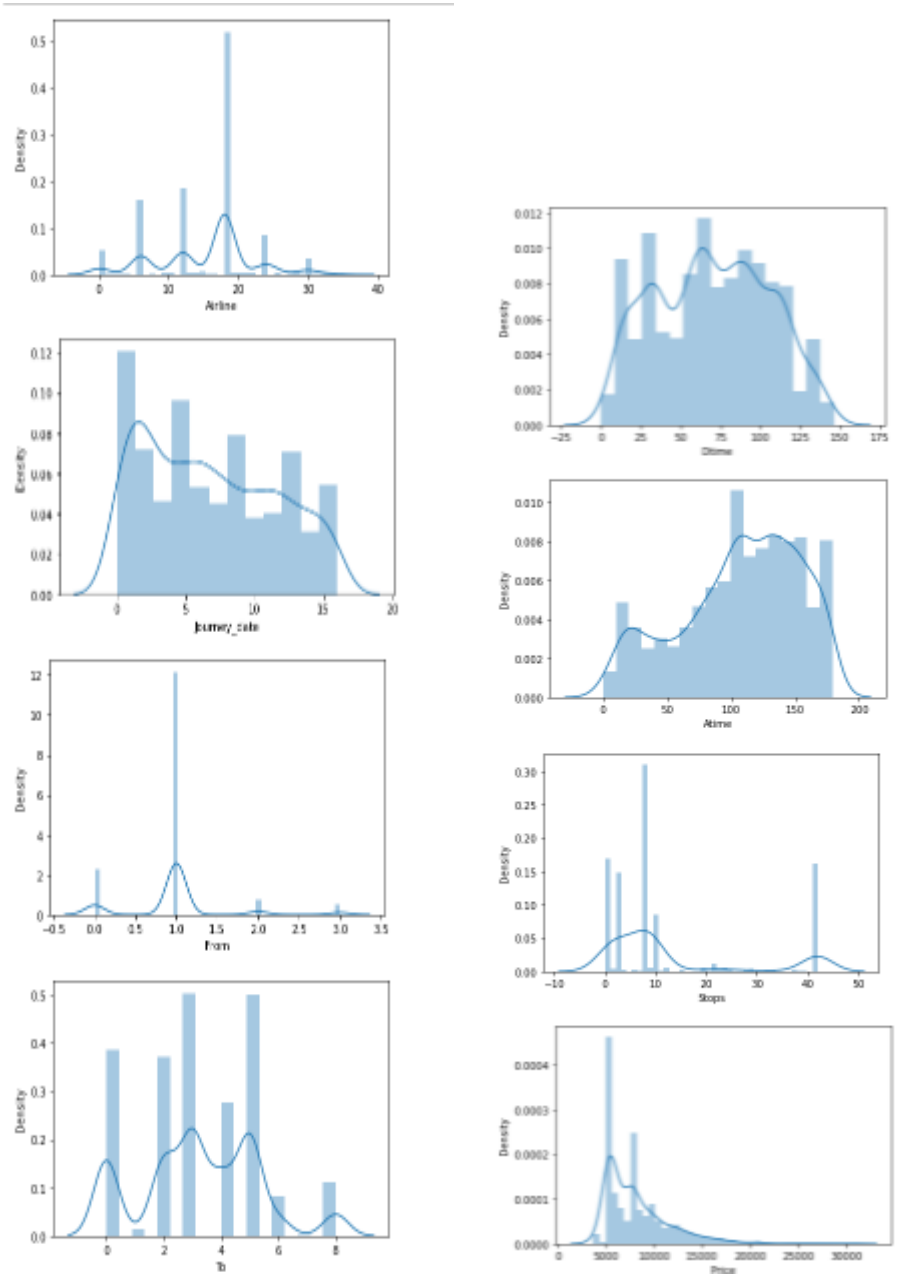


Observation:

Presence of unusual values in above histograms & also distribution is not normal in some columns and these things denote the possibility of potential outliers.

Skewness: - Skewness is a measure of asymmetry or distortion of symmetric distribution. It measures the deviation of the given distribution of a random variable from a symmetric distribution, such as normal distribution. A normal distribution is without any skewness, as it is symmetrical on both sides. Hence, a curve is regarded as skewed if it is shifted towards the right or the left. Skewness is of 2 types: - 1- Positive Skewness 2- Negative Skewness.

Distplot to check Distribution of Skewness: - Distplot plots a univariate distribution of observations. The distplot () function combines the matplotlib hist function with the seaborn kde plot () and rug plot () functions. For individual columns we are using Distplot.



Observation: Skewness is present in various columns. So, we have removed most of skewness using NumPy mathematical function cube root transform.

- Data Inputs- Logic- Output Relationships

Output Feature: - In this project Target Variable = dependent variable = y and shape of our dependent variable is (2190,1). The head of output feature is as: -

```
#Output feature
y=df1['Price']
y.head()
```

```
0    5315.0
1    5315.0
2    5315.0
3    5315.0
4    5315.0
Name: Price, dtype: float64
```

Output Variable “Price" is continuous in nature so it is a Regression based problem an d We have to predict the flight price with available independent variables.

Input Feature: - The Independent variable in this project = feature vector = x and shap e of Independent variable is 2190 rows × 7 columns. After certain transformations and analysis input feature is as: -

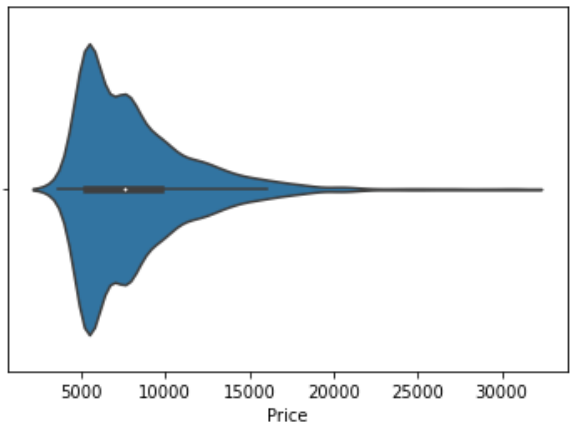
1- Data Pre-processing: - Using Standard Scaler’s fit _ transform method we have tried to bring input features(x) to common scale and modified the input feature as x1. The shape was remained same and head of input feature is as: -

	Airline	Journey_date	From	To	Dtime	Atime	Stops
0	1.272531	-1.49245	0.229692	-1.543399	0.301768	-0.099654	1.708491
1	-0.500458	-1.49245	0.229692	-1.543399	0.560798	-0.056036	1.708491
2	1.272531	-1.49245	0.229692	-1.543399	1.337888	0.968982	1.708491
3	-0.500458	-1.49245	0.229692	-1.543399	1.568137	1.056218	1.708491
4	-0.500458	-1.49245	0.229692	-1.543399	0.704704	0.838129	-0.028658

Correlation between input and output features: -In above steps, we have checked already about correlation between input variables and output variable "Price". As per observations “Destination” column is most positively correlated with output and “Journ ey Date” column is most negatively correlated with output feature.

Data Visualization: Data visualization is the graphical representation of information and data. Different visualizations for our input and output features are as: -

Exploring Output Feature: - Violin plot for our target variable & value counts are as: -



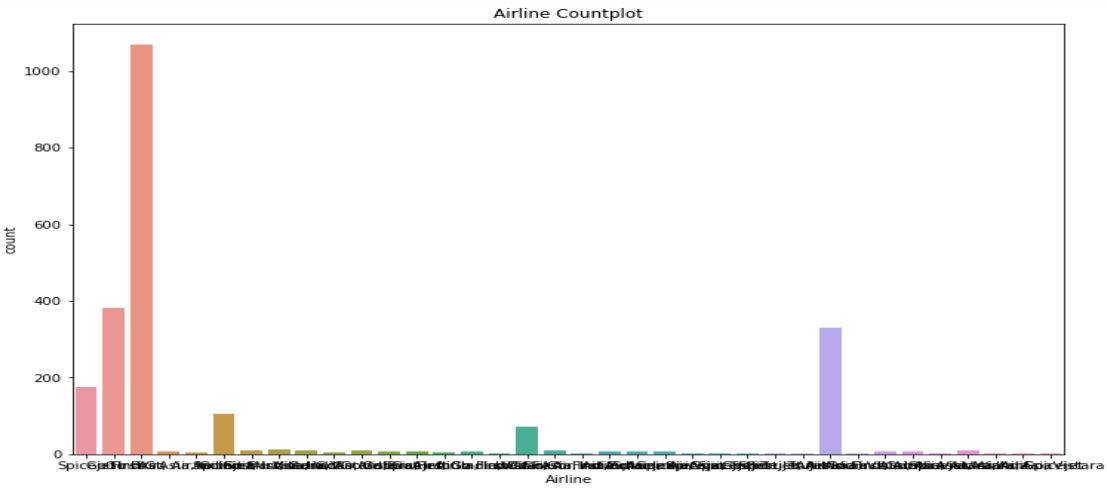

```
5316.0     199
5315.0     168
7622.0     131
7626.0     127
5060.0     101
           ...
30560.0      1
8003.0       1
15615.0      1
15143.0      1
17601.0      1
Name: Price, Length: 542, dtype: int64
```

Observation:

Maximum number of flight Prices range between 5316-5060.

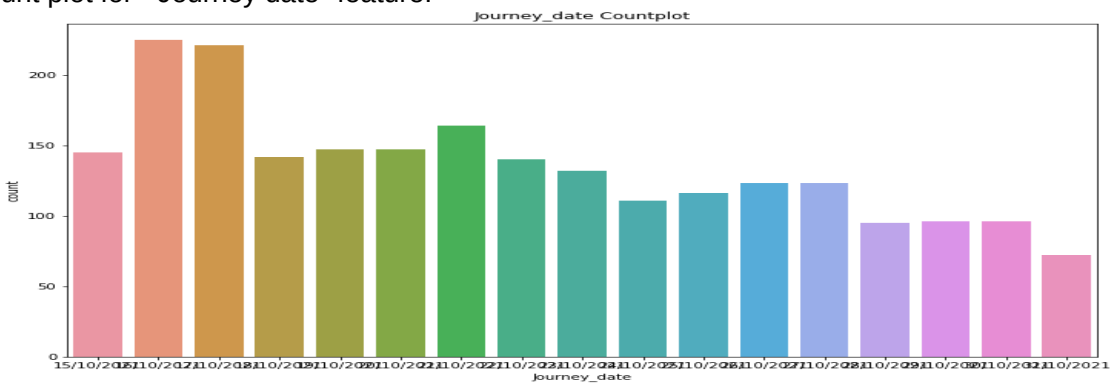
Exploring Input Features: - Plots for different input variables are as: -

1- Count plot for " Airline" feature: -



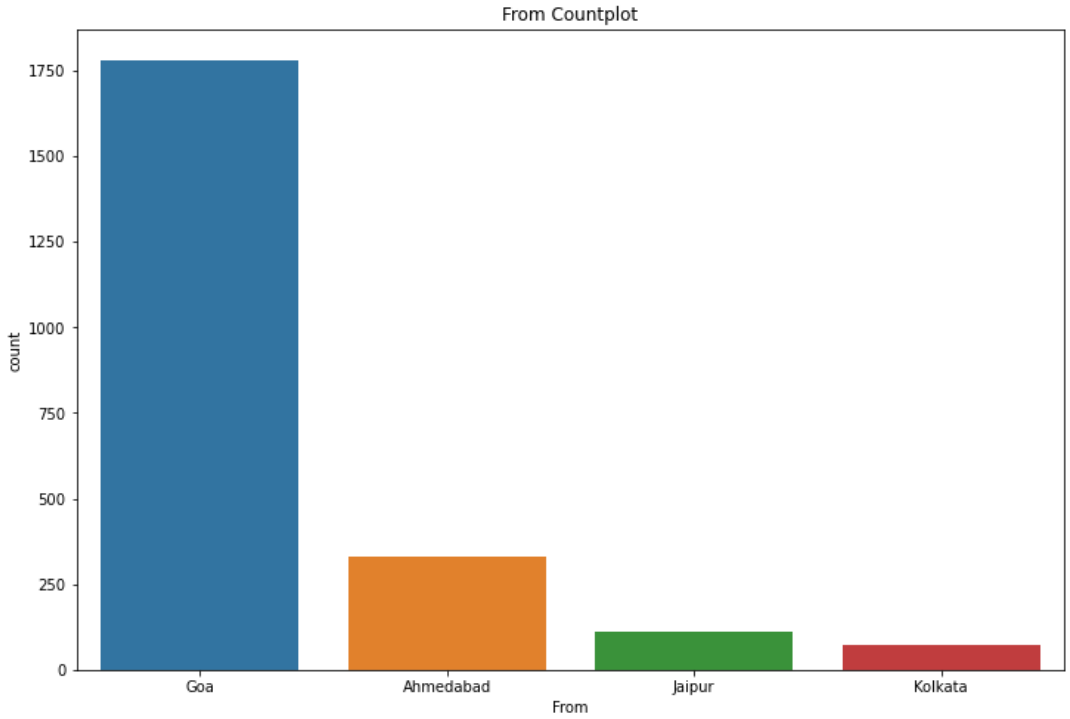
Observation: Indigo, Go First & AirAsia are most popular airlines.

2- Count plot for " Journey date" feature: -



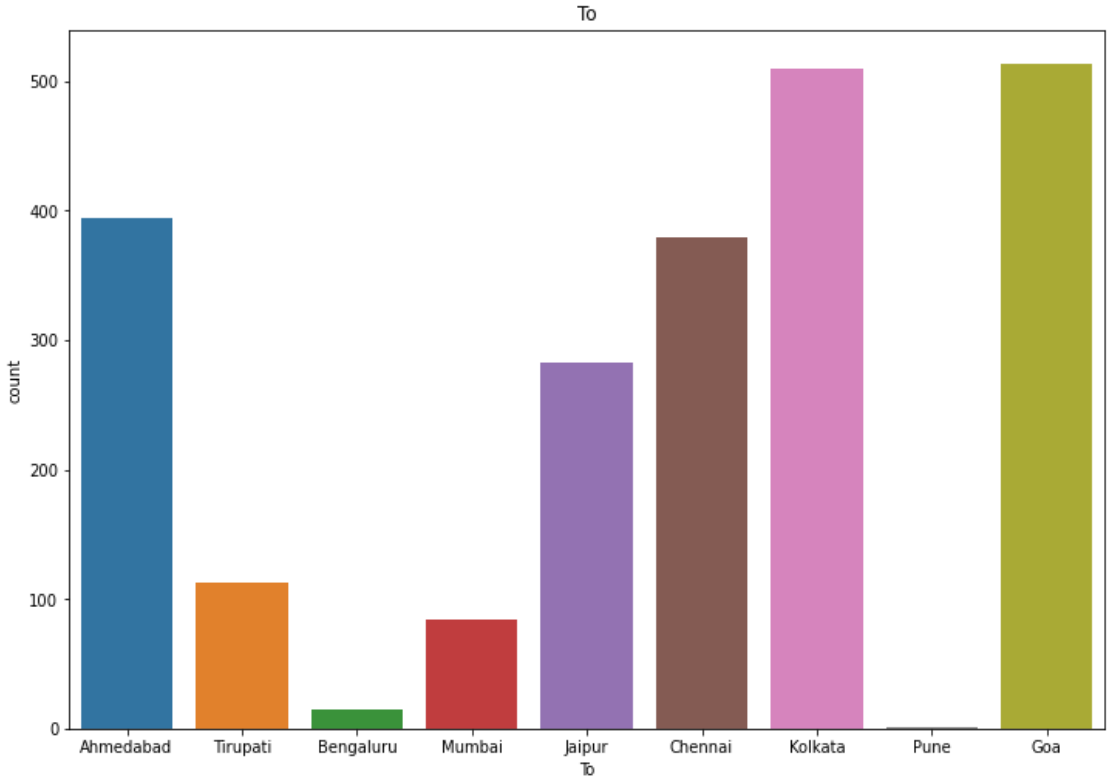
Observation: 16-10-2021 & 17-10-2021 dates have maximum bookings and 31-10-2021 have minimum bookings.

3- Count plot for “From" feature: -



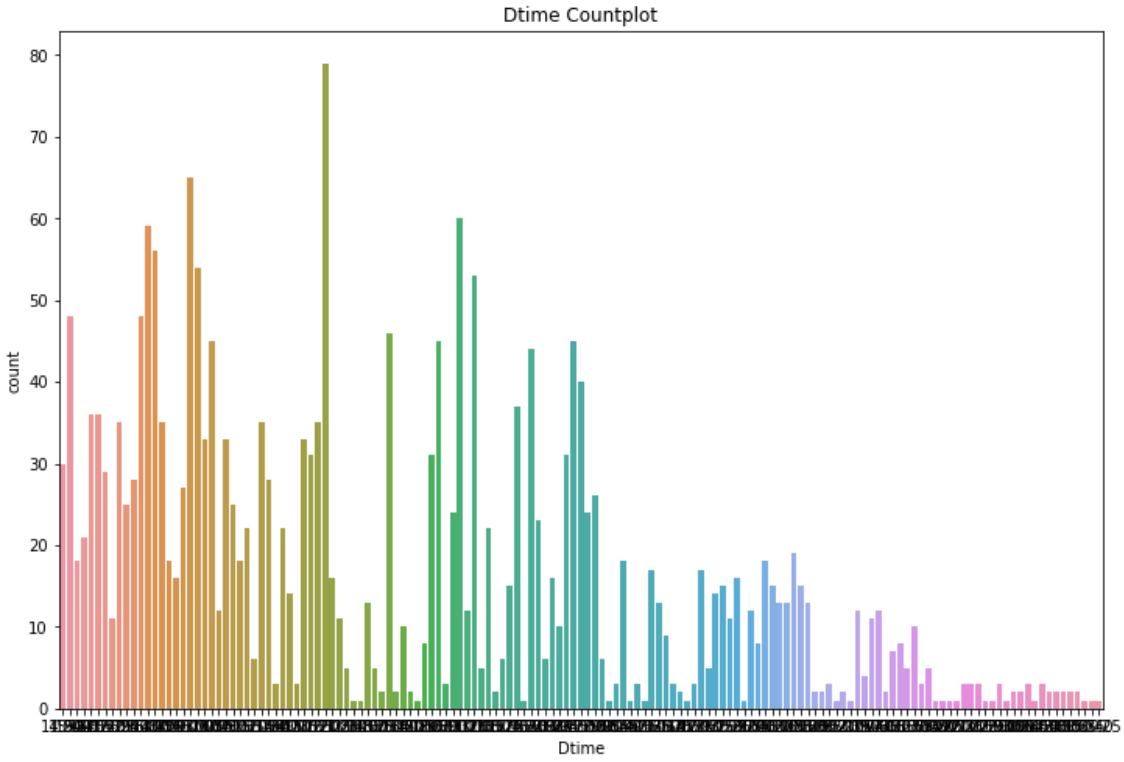
Observation: For source, Maximum bookings done from Goa and minimum bookings done from Kolkata.

4- Count plot for “To" feature: -



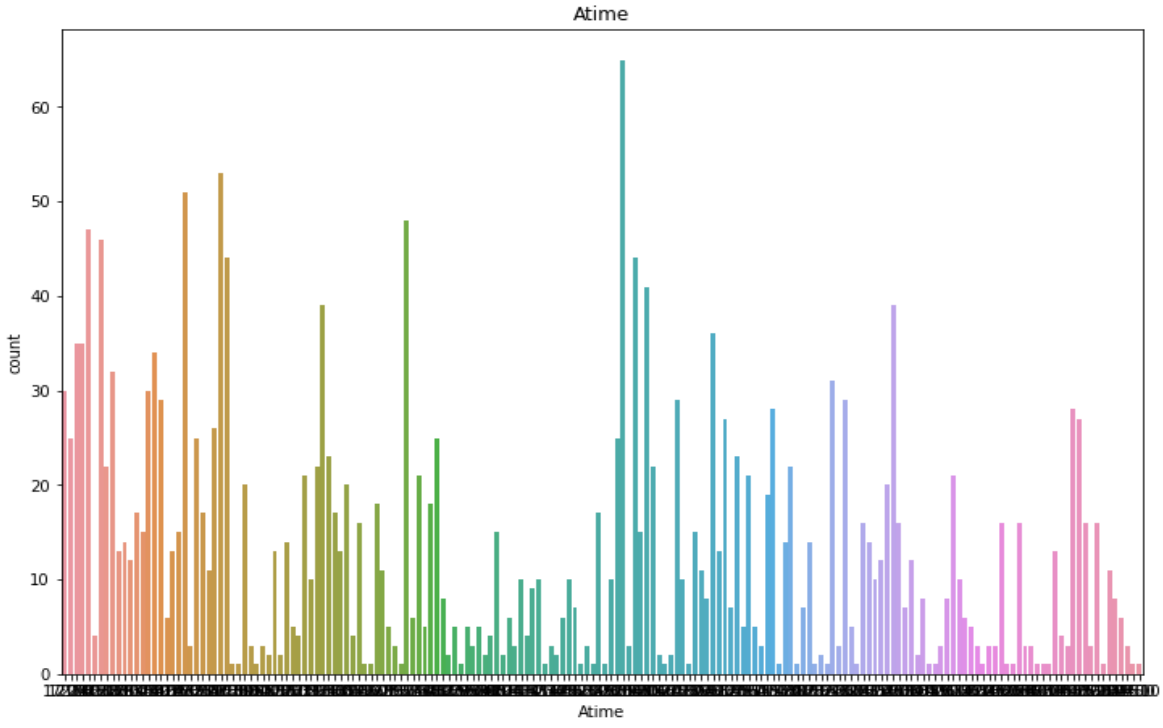
Observation: For Destination, Maximum bookings done for Goa & minimum bookings done for Pune.

5- Count plot for “D time" feature: -



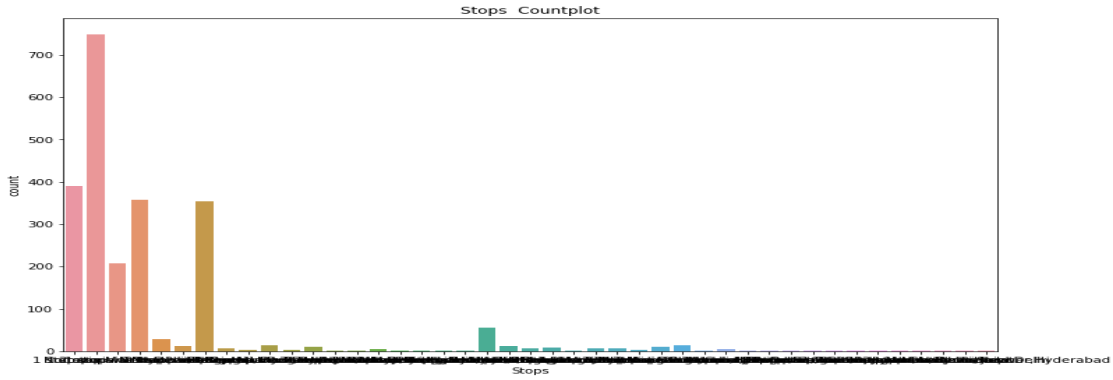
Observation: Most of the Flight departed at 12:10 & in night very few flights departed.

6- Count plot for “A time" feature: -



Observation: Maximum flights arrived at 23:15 & less flights arrived in morning.

7- Count plot for “Stops" feature: -



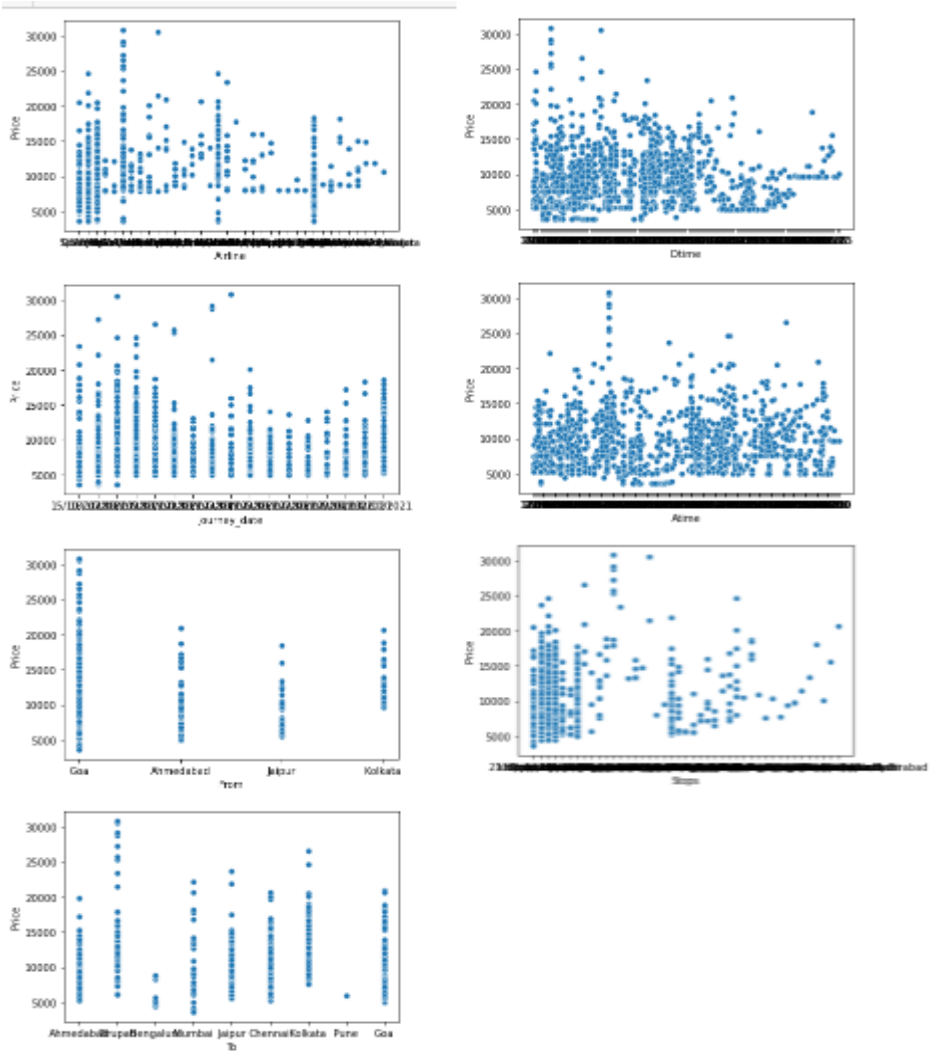
Observation: Most of the flights have stoppage at Mumbai and many flights are non-stop.

Bivariate Analysis: -

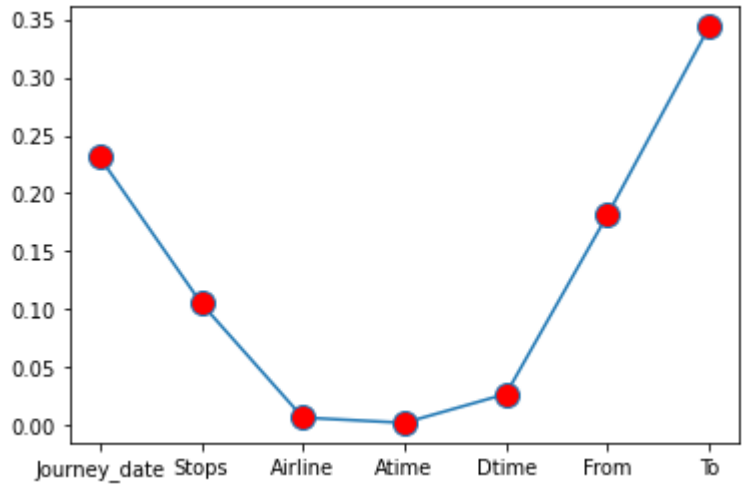
It involves the analysis of two variables, for the purpose of determining the empirical relationship between them. We are using scatterplot for this purpose.

Scatter Plot- Scatter plot reveals relationships or association between two variables.

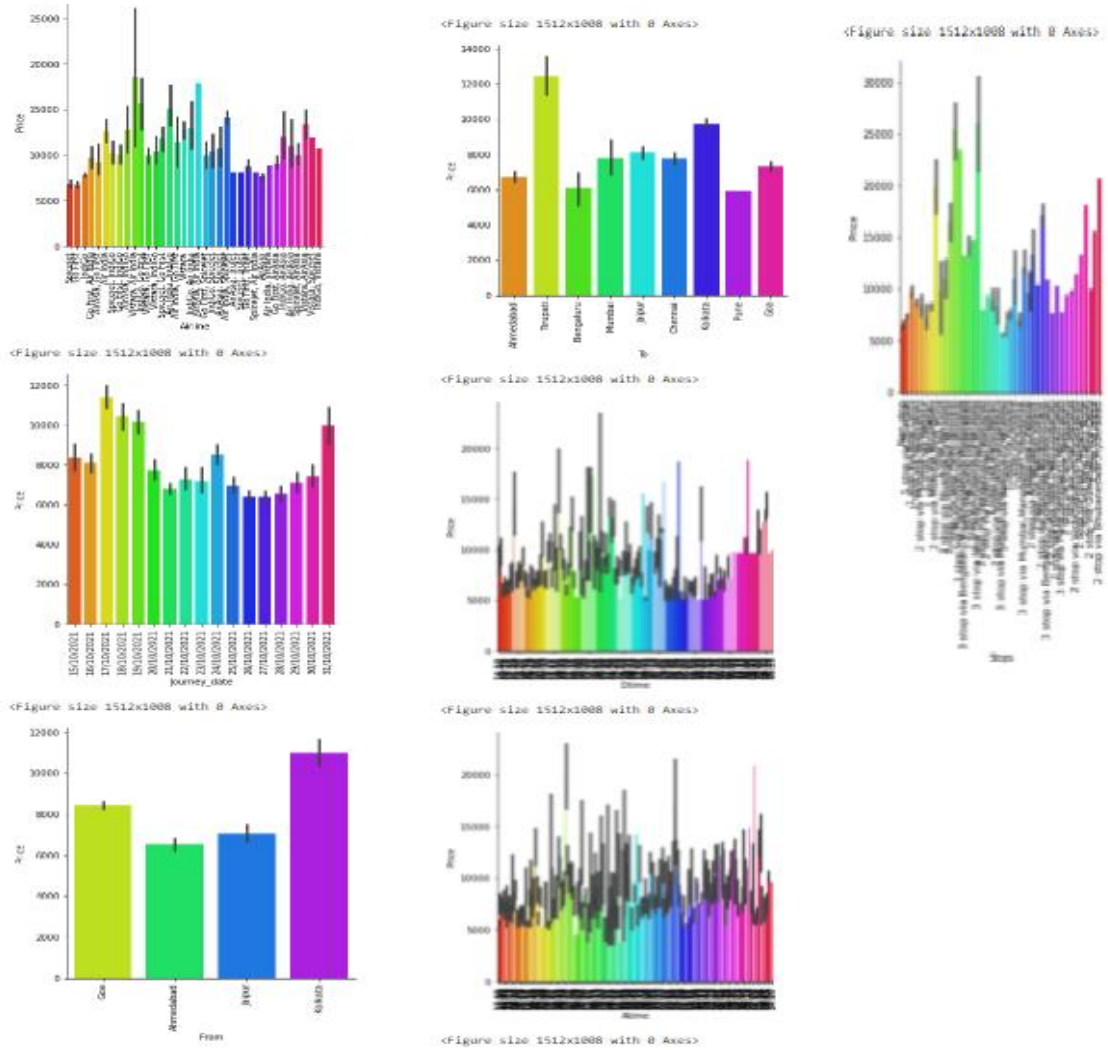
Scatter Plot between Output and Input Features: -



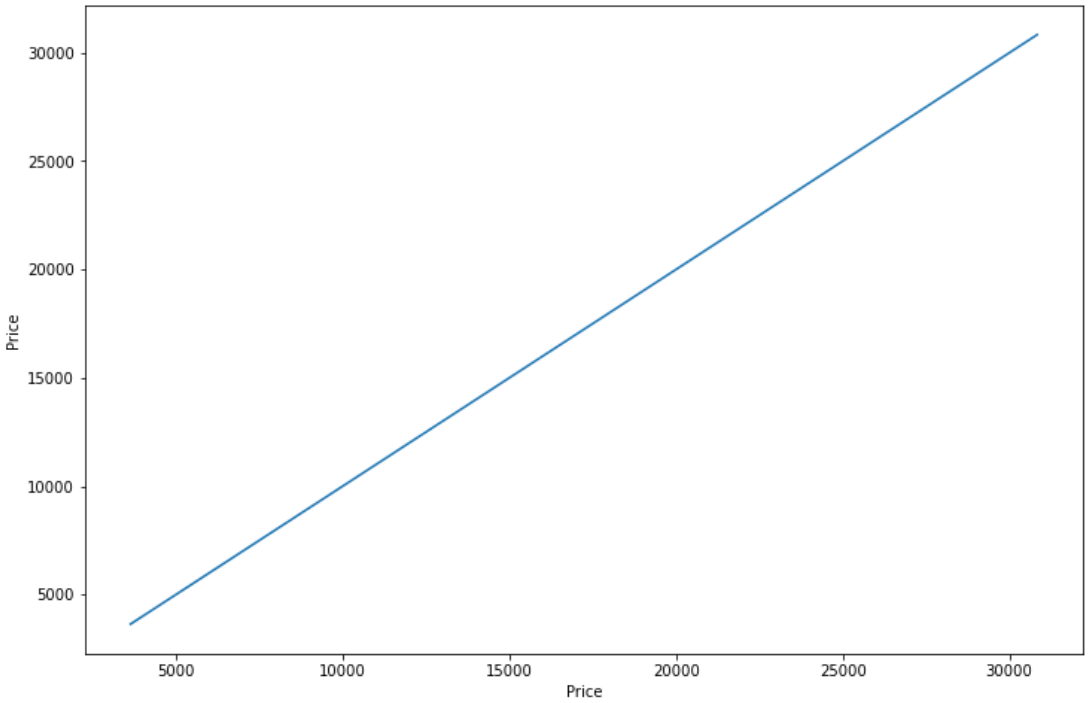
Marker Plot for checking highly correlated values with Output variable Price: -



Cat Plot between Output and Input Features (Categorical Type): -



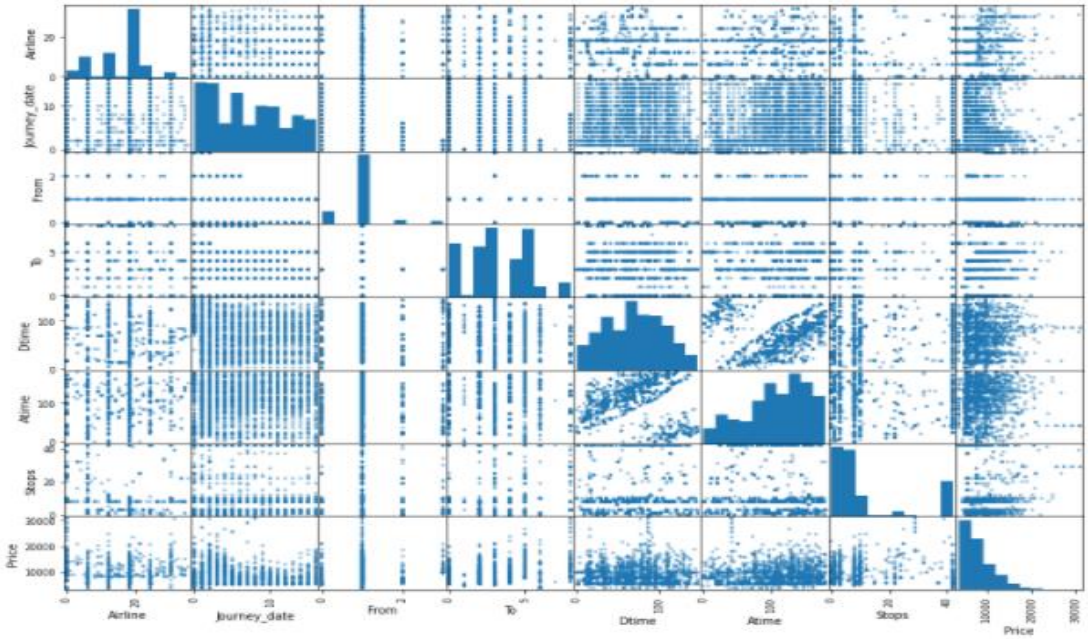
Line Plot between Output and Input Features (Continuous Type): -



Observation: Flight prices are increasing year by year.

Multivariate Analysis: - Multivariate analysis is used to study more complex sets of data. It is a statistical method that measures relationships between two or more response variables.

Scatter plot matrix: -Scatter plot matrix is a grid (or matrix) of scatter plots used to visualize bivariate relationships between combinations of variables.



Observation: Using multivariate analysis, we can look at interactions between variables. Scatter plots of all pair of attributes helps us to spot structured relationship between input variables.

- State the set of assumptions (if any) related to the problem under consideration

- By looking into the target variable "Price", we assume that this project is a Regression based problem as target variable is continuous in nature.
- We have removed irrelevant column which does not have more impact on dataset.
- We had found outliers in dataset so tried to remove them using Z scores and data loss was nearly 4.5%.
- We have tried to remove skewness using transformation method (cube root).

- Hardware and Software Requirements and Tools Used

Hardware:4GB RAM, Intel I3 Processor.
System Software: 64Bit O/S Windows 10(x64-based processor)

Software Tools: Software Tools used in this project are as: -

1- Anaconda3 (64-bit)
2- Jupyter Notebook 6.1.4
3- Python 3.8
4- MS-Office 2019 (Excel, Word, Power point)
5- Notepad
6- Google Chrome Web Browser
7- Selenium Driver

- Libraries & Packages used:

We have used Python and Jupyter Notebook to compute the majority of this project. For analysis, visualization, statistics, machine learning & evaluation, we have used these: -

1- Pandas (data analysis)
2- NumPy (matrix computation)
3- Matplotlib (Visualization)
4- Seaborn (visualization)
5- Scikit-Learn (Machine Learning)
6- SciPy (Z-score)
7- Selenium Web driver & Exceptions
8- Warnings (filter warnings) & etc. Microsoft Excel (for calculations and Data Handling).

Packages and libraries used in this project are: -

1- For Data Analysis & Visualization: -

```
#Importing Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

2- For Z score: -

```
from scipy.stats import zscore
```

3- From Scikit-Learn Library: -

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import RidgeCV
from sklearn.linear_model import Lasso
from sklearn.ensemble import AdaBoostRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
```

4- For saving the final model: -

```
import joblib
```

Function to calculate maximum R2 score at best random state: -

```
def maximumr2_score(rgn,x1,y):
    maximum_r_score =0
    for r_state in range(42,100):
        x_train,x_test,y_train,y_test=train_test_split(x1,y,random_state=r_state,test_size=0.20)
        rgn.fit(x_train,y_train)
        pred=rgn.predict(x_test)
        r2_scr=r2_score(y_test,pred)
        if r2_scr>maximum_r_score:
            maximum_r_score=r2_scr
            final_r_state=r_state
    print('Maximum r2 score for final_r_state',final_r_state,'is',maximum_r_score)
    return final_r_state
```

Model/s Development and Evaluation

- **Identification of possible problem-solving approaches (methods)**

- As per visualizations, Target Variable “Price” is continuous in nature so we will use different regression algorithms to try and find the features that have the best explanation of the target variable.
- Explored input features and their values using count plot.
- Checking missing values in dataset.
- Checking Summary Statistics to summarize set of observations as- central Tendency, dispersion, skewness, variance, range, deviation etc.
- Checking Correlation between target variable and input features using Correlation Matrix & correlation Heatmap using Seaborn.
- To check distribution & spread of data we used Histogram.
- Checked Scatter Plots between input & output feature for bivariate analysis.
- To check highly correlated values with Output variable Price used Marker Plot.
- Divided input features into category type & continuous type features.
- Checked Cat plot for category type & Line plot for continuous type features.
- Used Scatter matrix for multivariate analysis.
- Removed irrelevant columns which does not have more impact on dataset.
- Used Label Encoding to encode categorical data in to numerical format.
- Used Boxplot for summarizing variations & check outliers.
- Removed outliers using Z scores method and data loss was only 4.5%.
- Divided dataset into input and output sets to explore more briefly.
- Used Distplot to check distribution of skewness and removed skewness using NumPy mathematical function cube root transformation.

- Standardization is useful to speed up the learning algorithm and it rescales the features so that they will have the properties of the standard normal distribution with μ = 0 and σ = 1.
- We have used Standard Scaler to standardize the data.
- We will use Coefficient of determination(R2) score as our metric.
- After Splitting data in to Training & Test Sets, checked scores at best random state after applying different regression algorithms.
- Used Cross validation to check how accurately a predictive model will perform in practice.
- To check error of forecasting model, we used Error Metric (MAE, MSE, RMSE).
- To choose set of optimal hyperparameters for learning algorithm, we did Hyperparameter Tuning. We used Grid Search CV to find estimators/neighbors/alpha for learning algorithms and Randomized search CV to find best parameters for final model.
- AdaBoost is used as ensemble method to combine several machine learning techniques into one predictive model in order to decrease variance, bias & improve predictions.
- Compared all algorithms on basis of scores, plots and errors & finalized the best model.
- Implementing the best model, calculating scores/errors & also checking predicted values.
- Checked scatterplot between Predicted values and Test values.
- Saved final model using job lib.
- Test Dataset sheet checked, handled missing values, removed irrelevant columns, did feature engineering, standardization, principal component & variance analysis.
- Loading saved model to predict values for Test Dataset.
- In last, we saved predicted values of test dataset in a CSV file.

• Testing of Identified Approaches (Algorithms)

1-Linear Regression: - In statistics, linear regression is a linear approach to modelling the relationship between a scalar response and one or more explanatory variables. Linear Regression fits a linear model with coefficients w = (w1, …) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

2-k-nearest neighbors: - K nearest neighbors is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure (e.g., distance functions). A simple implementation of KNN regression is to calculate the average of the numerical target of the K nearest neighbors. We have used Grid Search CV method to find n_neighbors.

```
#using gridsearch CV to find the best parameters to use in k-nearest neighbors regression.
gridknr=GridSearchCV(knreg,neighbors,cv=10)
gridknr.fit(x1,y)
gridknr.best_params_
```

```
{'n_neighbors': 21}
```

3-Decision Tree Regression: - Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output which means that the output is not discrete.

4- **Random Forest Regression**: - Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees.

5- **Ridge Regression**: - Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where independent variables are highly correlated. This method performs L2 regularization. It reduces the model complexity by coefficient shrinkage.

6- **Lasso Regression**: - Lasso (Least Absolute Shrinkage and Selection Operator) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model. We have used Grid Search CV to find best parameters for Lasso regression.

```
#using gridsearch CV to find the best parameters to use in Lasso regression.
parameters={"alpha":[0.001,0.01,0.1,1]}
gsc=GridSearchCV(lasso_reg,parameters,cv=10)
gsc.fit(x1,y)
gsc.best_params_
```

```
{'alpha': 1}
```

7-AdaBoost Regression: - An AdaBoost regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction. We have used Grid Search CV to find best parameters.

HyperParameter Tuning to find best parameters using Grid Search CV¶

```
parameters={"learning_rate":[0.1,1],"n_estimators":[10,100],"base_estimator":[RandomForestRegressor(),DecisionTreeRegressor(
#using GridsearchCV to loop through predefined hyperparameters and fit our estimator on our training set.
gsc=GridSearchCV(abr,parameters,cv=5)
gsc.fit(x1,y)
gsc.best_params_
```

```
{'base_estimator': DecisionTreeRegressor(),
 'learning_rate': 1,
 'n_estimators': 100}
```

- # Run and evaluate selected models

Selected Models: -

1- Linear Regression ()

2- K Neighbors Regressor (n_neighbors=21)

3- Decision Tree Regressor ()

4- Random Forest Regressor ()

5- Ridge CV ()

6- Lasso(alpha=1)

7-Ada Boost Regressor (base estimator=Decision Tree Regressor (), learning_ rate=1, n_ estimators=100)

- **Key Metrics for success in solving problem under consideration:**

	Model	Maximum r2 score	Cross Validation Score	Mean absolute error	Root Mean Squared Error	Mean squared error
0	Linear Regression	20.66	-8.86	2043.37	2647.21	7007713.92
1	k-nearest neighbors	39.93	-9.56	1661.72	2403.29	5775806.56
2	Decision Tree Regression	90.46	-33.79	272.57	968.20	937402.92
3	Random Forest Regression	92.04	-8.97	502.17	872.45	761171.09
4	Ridge Regression	20.66	-8.76	2044.11	2647.24	7007875.48
5	Lasso regression	20.65	-8.86	2043.65	2647.40	7008710.78

Evaluation Metrics used: -

1- **r2 score**: - Coefficient of determination, denoted R2 or r2, is the proportion variation in the dependent variable that is predictable from the independent variables. It is a regression score function and best possible score is 1.0. We have calculated maximum & mean r2 score for models and will select model with best r2 score.

2-**Cross validation score**: - It is a score evaluated by cross-validation. When we want to estimate how accurately a predictive model will perform in practice and our goal is prediction then we use cross validation. cross validation score returns score of test fold where cross validation predicts returns predicted y values for the test fold. We will select model with best cross validation score.

Error Metrics used: - We will select model with minimum errors.

1-Mean Absolute Error (MAE)- MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. MAE score is calculated as the average of the absolute error values. MAE can be calculated as follows:

$$MAE = 1 / N * \text{sum for i to N } abs(y\_i - yhat\_i)$$

2-Mean Squared Error (MSE): - MSE is calculated as the mean or average of the squared differences between predicted and expected target values in a dataset. MSE can be calculated as follows:

$$MSE = 1 / N * \text{sum for i to N } (y\_i - yhat\_i)^2$$

3-Root Mean Squared Error (RMSE): - RMSE is an extension of the mean squared error. It's the square root of the average of squared differences between prediction and actual observation. RMSE can be calculated as follows:

$$RMSE = sqrt(1 / N * \text{sum for i to N } (y\_i - yhat\_i)^2)$$

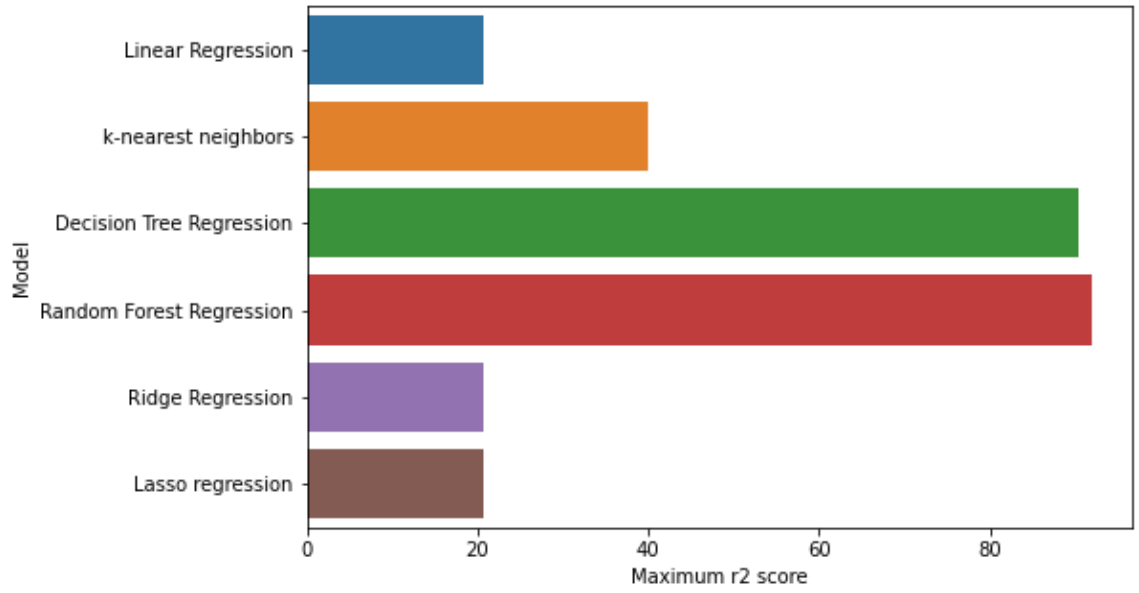
Hyperparameter Tuning: - Hyperparameter tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. We used Grid Search CV to find estimators /neighbors/alpha for learning algorithms and Randomized search CV to find best parameters for implementation of final selected model.

- Visualizations (machine learning): -

All visualizations done before applying machine learning algorithms are shown above. We have done visualizations for exploring output variable, input features, relationship & correlation between input and output features, to check outliers, skewness & missing values etc. Here we are visualizing different regression model's performances, scores after applying ensemble methods & Hyperparameter Tuning, Final model selection, relation between true & predicted values & test dataset.

1- **Before using ensemble methods**: - After applying different regression algorithms on model and using cross validation we have obtained different coefficient of determination & cross validation scores and after using Error metrics we have calculated MAE, MSE & RMSE for all models. All scores & errors are shown above. Now we are seeing visualization using Bar Plot of r2 scores: -

```
<AxesSubplot:xlabel='Maximum r2 score', ylabel='Model'>
```



Observations: After comparing above 6 models, these 2 models are good: -

1- Random Forest Regression (R2 & Cross validation scores are higher & RMSE is minimum & other errors are less.)

2- Decision Tree Regression (R2 score is good & MAE is minimum & also other errors are less.

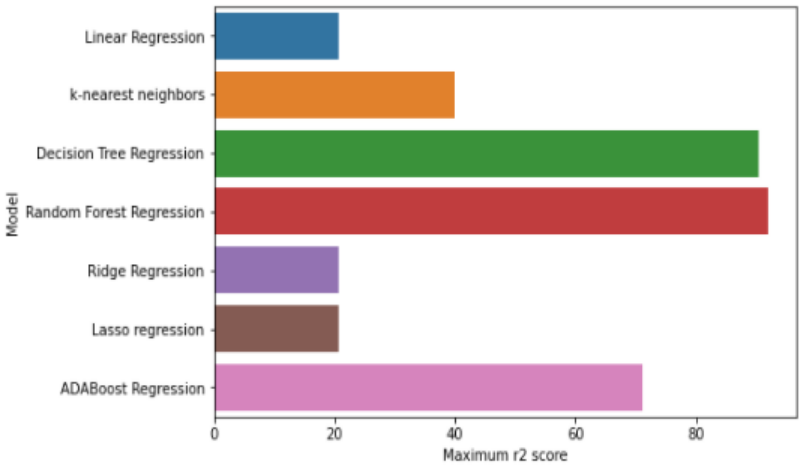
2- **After using ensemble methods**: - AdaBoost is used as ensemble method to combine several machine learning techniques into one predictive model in order to decrease variance, bias & improve predictions. After applying ensemble methods on model and using cross validation we have obtained different coefficient of determination & cross validation scores and after using Error metrics we have calculated MAE, MSE & RMSE for all models. Now we are seeing visualization using Bar Plot of r2 scores: -

	Model	Maximum r2 score	Cross Validation Score	Mean absolute error	Root Mean Squared Error	Mean squared error
0	Linear Regression	20.66	-8.86	2043.37	2647.21	7007713.92
1	k-nearest neighbors	39.93	-9.56	1661.72	2403.29	5775806.56
2	Decision Tree Regression	90.46	-33.79	272.57	968.20	937402.92
3	Random Forest Regression	92.04	-8.97	502.17	872.45	761171.09
4	Ridge Regression	20.66	-8.76	2044.11	2647.24	7007875.48
5	Lasso regression	20.65	-8.86	2043.65	2647.40	7008710.78
6	ADABoost Regression	71.22	-2.40	1076.51	1625.22	2641338.22

```
#Plotting bar plot of Maximum r2 scores of various models
plt.figure(figsize=(17,17))

plt.subplot(3,2,1)
sns.barplot(x = 'Maximum r2 score', y = 'Model', data = model2)
```

```
<AxesSubplot:xlabel='Maximum r2 score', ylabel='Model'>
```



Observations:

1- After comparing above 7 models on basis of scores and errors, & also after using ensemble methods still these 2 models Random Forest regression & Decision Tree Regression are giving good performance.

2- Ada boost Regression is giving good scores but errors are high so we will not select this option.

3- But when we see all the parameters very carefully and making a final decision about selection then Random Forest regression is the best option because all scores are higher, RMSE is less and also other errors are less.

4- Now we are going to do Hyperparameter Tuning for Random Forest regression models using Randomized Search CV approach for best model selection so that we will get best results after model implementation.

- ## Interpretation of the Results: -

- After comparing above 7 models on basis of scores and errors, & after using ensemble methods we have selected **Random Forest regression** for this project. To get best scores, now are using randomized search cv for hyperparameter tuning.

- **Hyperparameter tuning using randomized search cv** – Using Scikit-Learn 's Randomized Search CV method, we can define a grid of hyperparameter ranges, and randomly sample from the grid, performing K-Fold CV with each combination of values.

```
12  rf_random.best_params_
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits

{'n_estimators': 80,
 'min_samples_split': 10,
 'min_samples_leaf': 2,
 'max_features': 'sqrt',
 'max_depth': None}
```

- ## Evaluation & error metrics for final model: -

```
#Using best parameters obtained from RandomizedSearchCV in RandomForestRegressor model
rfc=RandomForestRegressor(n_estimators=80,max_depth=None,min_samples_leaf= 2, max_features= 'sqrt',min_samples_split=10)
print("For RandomForest Regression R2 Score->")
r_state=maximumr2_score(rfc,x1,y)
print('Mean r2 score for Random Forest Regression is:',cross_val_score(rfc,x1,y,cv=5,scoring='r2').mean())
print("\tBest possible r2score is 1.0")
print('Standard deviation in r2 score for Random Forest Regression is',cross_val_score(rfc,x1,y,cv=5,scoring='r2').std())
```

```
For RandomForest Regression R2 Score->
Maximum r2 score for final_r_state 57 is 0.7346436178530239
Mean r2 score for Random Forest Regression is: 0.10304477386862225
        Best possible r2score is 1.0
Standard deviation in r2 score for Random Forest Regression is 0.3824821313011918
```

```
#Score & Error Metrics for RandomForestRegressor after Hyperparameter Tuning
x_train,x_test,y_train,y_test=train_test_split(x1,y,random_state=57,test_size=0.20)
y_pred=rfc.predict(x_test)
r2score=r2_score(y_test,y_pred)
print("r2_score =",r2score*100)
print("Cross validation score =",(cross_val_score(abr,x1,y,cv=5,scoring="r2").mean())*100)
mse=mean_squared_error(y_test,y_pred)
print("Mean_Squared_Error =",mse)
mae=mean_absolute_error(y_test,y_pred)
print('Mean Absolute_Error =',mae)
rmse=np.sqrt(mean_squared_error(y_test,y_pred))
print('Root Mean Squared Error =',rmse)
```

```
r2_score = 81.51869966196254
Cross validation score = -3.764743273956302
Mean_Squared_Error = 1588902.0765823151
Mean Absolute_Error = 895.0677519804589
Root Mean Squared Error = 1260.5165911570998
```

Observation:

Random Forest Regression gives best performance before Hyper Parameter Tuning & also after Hyper Parameter Tuning as compared to other models so we are going to implement Random Forest Regression model in our project.

After implementation of Random Forest regression in our model we are going to check predicted values, true values and relationship between them using visualizations and also Checking Scores and errors after model fitting.

Predictions after Model Fitting: -

```
rfr.fit(x_train,y_train)
y_pred=rfr.predict(x_test)
print(y_pred)
```

```
[ 7457.46800061  8560.58541667  6137.36212527  5313.50155754
  6344.74367514  7571.59032166  6688.05536699  8419.24291171
  5878.11635146  7402.34304563 12614.35045409  5382.71153229
  6586.07230769 12914.00676272 14136.07056863  5499.62318948
  5316.78138393  6637.98881719  6065.92750902  6476.36277226
  7334.79446834  6419.22642316  6192.89057495 10270.3934127
  5348.40011905  6982.84495806  5736.45058938  9788.40562004
  5557.85348558  5492.88278319  7146.54075921 12214.38762085
  8262.97524382  7440.01947917  7745.57764881  7624.35752615
  5429.69016671  7057.01605474 13532.95514881  7754.39920455
 12619.1565882   6276.68548521  7797.7677381   7009.42889159
  5396.62746483 14386.37820572  5890.32118867  6720.97891775
  5762.28331439  7451.91845734  8691.01993409 10026.17884921
  7940.63456894  8656.92335588  5519.59895427  8716.61550189
 11161.11346411  5335.35374369  7110.24758568  7393.33304654
  7770.80634921 14541.35494589 11358.7975983   9863.023966
  7245.00323097  6419.32240076  8347.06130907  7541.04649739
 11996.72072781  9451.86944986 12766.60758013  6156.75610029
  5686.49818744  5703.66917659  9063.1741121   5760.69830177
```

Error Metrics & R2Score after Model Fitting: -

```
#Error Metrics & R2Score for our final model
m1=mean_absolute_error(y_test,y_pred)
m2=mean_squared_error(y_test,y_pred)
print("Mean Absolute Error is: ",m1)
print("Mean Squared Error is: ",m2)
print('Root Mean Square Error after model fitting is:',np.sqrt(mean_squared_error(y_test,y_pred)))
print('Score is:',(rfr.score(x_train,y_train))*100)
print('r2_score after model fitting is:',(r2_score(y_test,y_pred))*100)
```

```
Mean Absolute Error is:  1006.1502122972717
Mean Squared Error is:  2169812.650535245
Root Mean Square Error after model fitting is: 1473.0283943411428
Score is: 86.27967347248958
r2_score after model fitting is: 74.76184350009092
```

Saving Final model: - We have saved final model using job lib in *.pkl format.

Evaluate Predictions: -

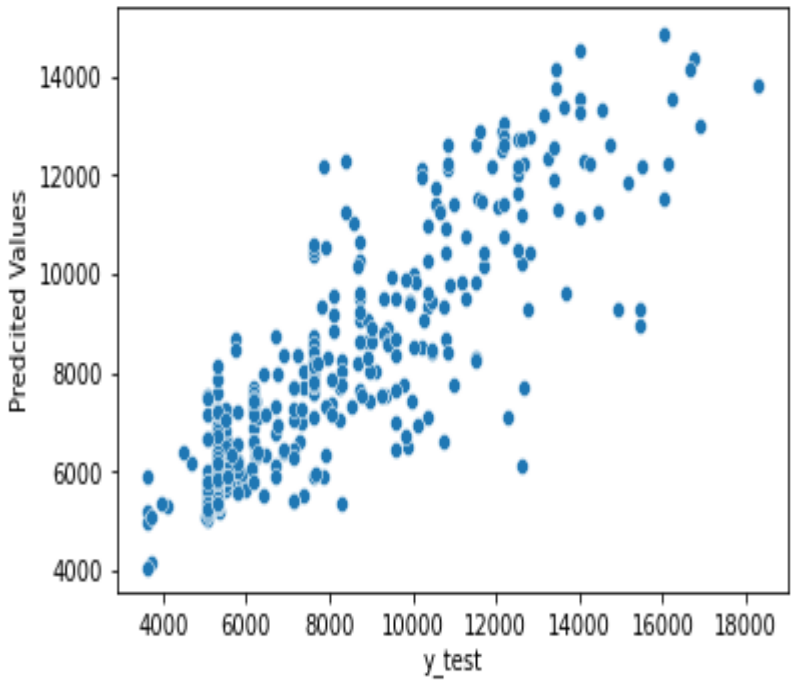
	Predicted Values	Real Values
0	7457.47	9989.0
1	8560.59	10188.0
2	6137.36	5314.0
3	5313.50	5061.0
4	6344.74	5524.0
...	...	...
433	10493.66	12503.0
434	8472.41	5735.0
435	8288.06	8918.0
436	7160.81	6456.0
437	10423.60	11711.0

438 rows × 2 columns

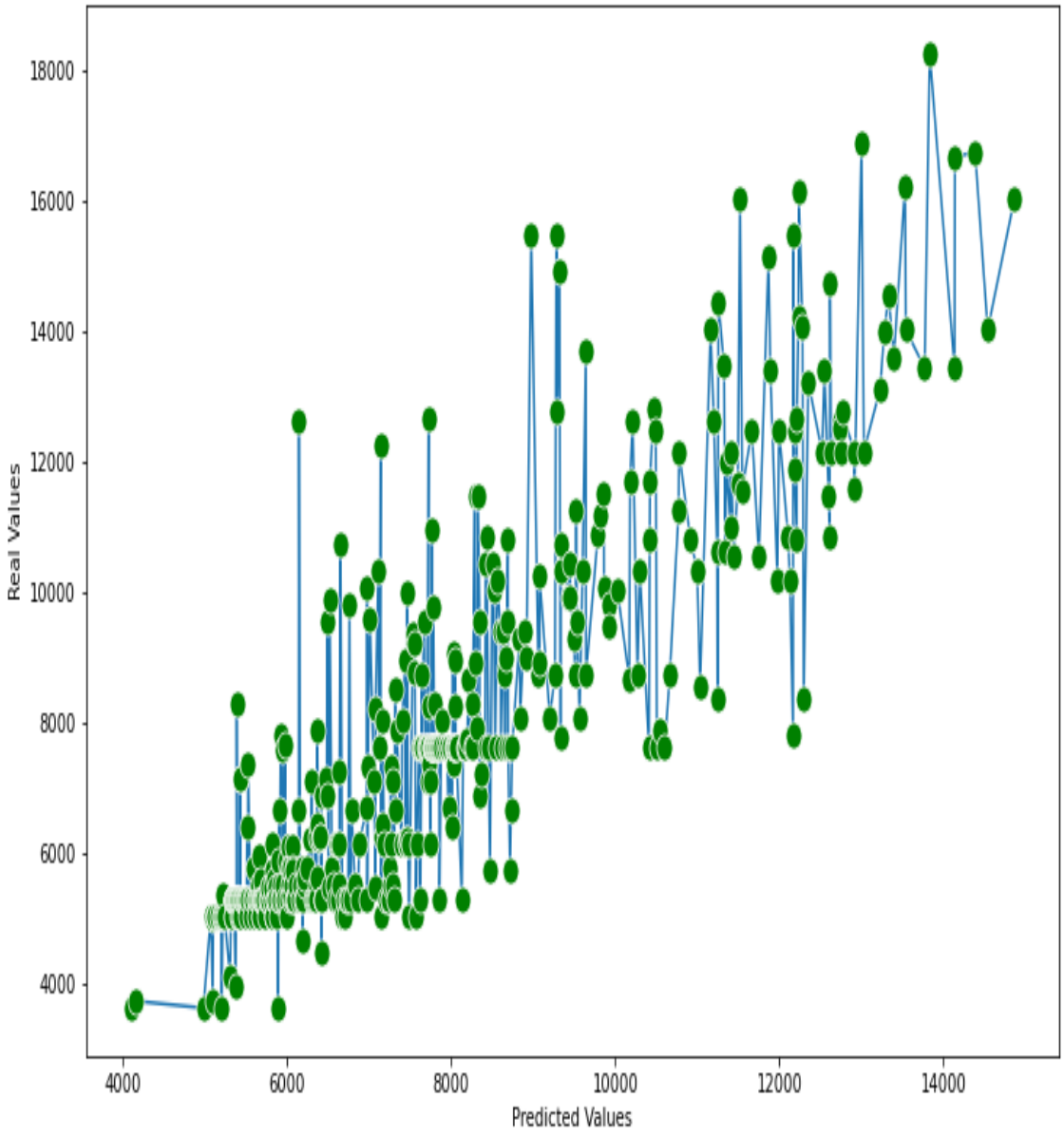
Visualizations to find relation between real and Predicted values: -

1- Scatter Plot: -

```
Text(0, 0.5, 'Predcited Values')
```



2- Line Plot: -



Observation: -

1- Above Plot shows Predicted values are nearly close to real values.

2- Graph is linear except few deviations and it shows good relation between predicted and real values.

3- R2 Score is best and error is minimum for our selected model.

4- Random Forest Regressor is best selection for this project.

CONCLUSION

• Key Findings and Conclusions of the Study

- In this project, we demonstrated the use of machine learning algorithms on a very challenging dataset to predict housing prices. To achieve the best performance, we showed that data pre-processing, a careful selection of techniques of balancing dataset, handling missing values, performed data cleaning, removing skewness, performed regression algorithms are all very important. Random Forest & Decision Tree regressors work quite well on our dataset, and the use of AdaBoost Regression is also effective. In the future, we want to continue exploring more sophisticated learning algorithms and dimension reduction techniques to further improve model performance on this important prediction task.

- Also, we have tested these machine learning algorithms on 2 different PCs of different technical configurations and have also compared their performance using evaluation & error metrics & and then chose the best performing model.

• **Learning Outcomes of the Study in respect of Data Science**

Learning outcomes are as: -

1- Data cleaning is quite tedious task and also very time taking and while working on this project we had encountered multiple issues but after research, study and guidance we neutralized these issues and also cleaned data properly.
2- Using Data visualization, we can easily identify outliers, skewness, missing values & correlation etc. Also, we can identify the relation between target & other features using it. In this project we have used Matplotlib & Seaborn library for data visualization.
3- Random Forest Regression have worked best in terms of r2 score & cross validation and RMSE is minimum and other errors are also less. Even it gave best parameters during hyperparameter tuning and we have used these parameters in our final model.

• Limitations of this work and Scope for Future Work

Following limitations and scope of future work are as: -

1- Selection of best random state to calculate the maximum accuracy of model is very time taking especially in case of Random Forest and AdaBoost Algorithms.
2- Hyperparameter tuning using Grid Search CV is very time consuming specially in prediction of best parameters for AdaBoost, Lasso & k-nearest neighbors. So, we used randomized search cv to get best parameters for random forest regression.
3- Dataset have some extra rows, so removed them carefully.
4- Size of dataset is huge so sometimes it was difficult to handle this dataset but after completion of this project we got enough confidence to handle big datasets.
5- There are some irrelevant columns in this dataset so we have tried our best to check and remove them and also tried not to lose important data.
6- We were trying to remove outliers using Z score and data loss was only 4.5%.
7- We have handled skewness using NumPy methods.
8- In future we will work on more algorithms to make more efficient model.

||Thank you||