FLIP ROBO

Flight-Price-Project

Submitted by:

HARPAL SINGH

ACKNOWLEDGMENT

The project is based on Flight-Price machine building model. Various techniques used in the pipeline of Machine Learning model building is used from various sources like geeksforgeeks, seaborn documentation, pandas library, NumPy library. Scikit-learn machine learning models have been used for Regression Analysis of target feature. Jupyter notebook has been used throughout the project and its various libraries have been called for various operations.

INTRODUCTION

Business Problem Framing

In this Flight-Price-problem we have to predict the Flight-Price of a different flights from a source to different destination. There is variation between features and label since flight timing, stop, duration of flight, flight departure, and arrival time is different for different flights and prices of flight vary accordingly. These types of problems with a lot of record can be solved with the help of Machine Learning model easily since Machine Learning models are able to give accurate results for the prediction of output with accuracy.

Conceptual Background of the Domain Problem

These types of problems are faced by potential passengers since these prices vary unexpectedly. The prices of flight increase overtime and flight becomes more expensive when departure date comes near. This usually happens to maximize the revenue based on Time of purchase patterns and keeping the flight as full as they want it. So, in this problem we have to build a model to predict the fares of flight based on the type of destination, source and other factors.

Review of Literature

Various techniques for data cleaning, EDA analysis, Data Pre-processing, Feature-Engineering and Machine Learning Models selection from sklearn library of machine learning have been used. Insights of the data are found by using various data visualization techniques like countplot and distplot for univariate analysis and scatterplot for bivariate analysis and heatmap for multivariate analysis. geeksforgeeks website, seaborn, pandas and NumPy documentation and sklearn for for the technical reference have been used.

Motivation for the Problem Undertaken

Objective to build this machine learning model is to have a hands on the model building techniques along with new facing new challenges while solving various anomalies in the dataset. And solving issues encountered during the machine learning building model. Since the number of records is quite high so computation takes a lot of time along with careful selection of features during feature engineering also posed some challenge.

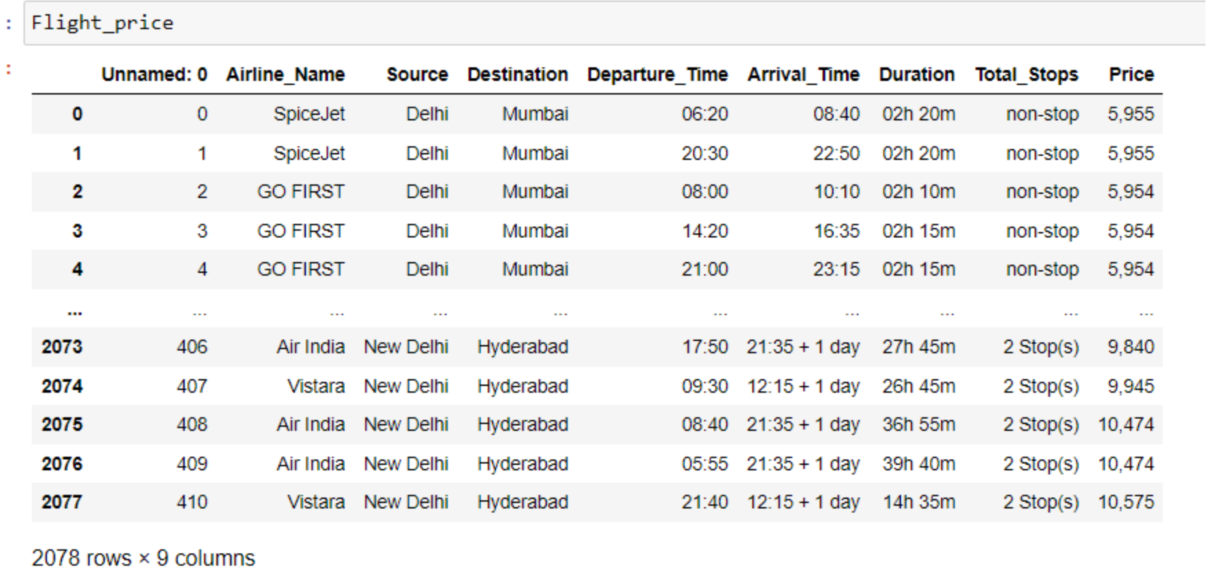
ANALYTICAL PROBLEM FRAMING

Mathematical/ Analytical Modeling of the Problem

Linear Algebra and Calculus concepts are used in the machine learning models. Since various Regressor Models have been used in the Machine Learning Algorithms, the mathematics working behind them. Calculation of evaluation metrics also involved mathematical concepts like algebraic summation. Use of Linear algebra in calculating the Euclidean distance in So, Log usage in DecisionTreeRegressor for calculating Information gain. Usage of exponential function in adaboost algorithm. In the statistical part descriptive statistics have been used to describe the data and to calculate various statistical parameters like mean, minimum value, maximum value, median value, count, standard deviation etc. And correlation coefficient has been calculated for each feature to analyse the correlation between the columns. And in the analytics modelling various techniques have been used like for analysing the data visualization distplot, countplot and scatterplots have been used.

Data Sources and their formats

DataSet have been web-scrapped from different flight websites like yatra, Googleflight using selenium python script and the data have been concatenated from various websites and features are extracted and used for prediction of Flight Price. The DataSet contains features like ‘Airline_Name’ which contains the Airline_Name of the Airline. ‘Source’ contains the source station of the flight.’Destination’ contains the destination of the flight. ‘Departure_Time’ contains the departure time of the flight.’Arrival_Time’ contains the arrival time of the flight to destination. ‘Duration’ contains the duration of the flight. ‘Total_Stops’ contains the total number of stops like non-stop,1-stop,2-stops and more.And finally ‘Price’ of the flight contains the price of the flight.



```
: Flight_price
```

	Unnamed: 0	Airline_Name	Source	Destination	Departure_Time	Arrival_Time	Duration	Total_Stops	Price
0	0	SpiceJet	Delhi	Mumbai	06:20	08:40	02h 20m	non-stop	5,955
1	1	SpiceJet	Delhi	Mumbai	20:30	22:50	02h 20m	non-stop	5,955
2	2	GO FIRST	Delhi	Mumbai	08:00	10:10	02h 10m	non-stop	5,954
3	3	GO FIRST	Delhi	Mumbai	14:20	16:35	02h 15m	non-stop	5,954
4	4	GO FIRST	Delhi	Mumbai	21:00	23:15	02h 15m	non-stop	5,954
...	...	...	...	...	...	...	...	...	...
2073	406	Air India	New Delhi	Hyderabad	17:50	21:35 + 1 day	27h 45m	2 Stop(s)	9,840
2074	407	Vistara	New Delhi	Hyderabad	09:30	12:15 + 1 day	26h 45m	2 Stop(s)	9,945
2075	408	Air India	New Delhi	Hyderabad	08:40	21:35 + 1 day	36h 55m	2 Stop(s)	10,474
2076	409	Air India	New Delhi	Hyderabad	05:55	21:35 + 1 day	39h 40m	2 Stop(s)	10,474
2077	410	Vistara	New Delhi	Hyderabad	21:40	12:15 + 1 day	14h 35m	2 Stop(s)	10,575

2078 rows × 9 columns

Data Pre-processing Done

In the data pre-processing and cleaning. Data have been checked for datatype matching values. After that the dataset has been checked for the empty values. Data have been checked for null values. In this dataset there were no null values. After that data is checked for the type of data like categorical or continuous. Some of the features having categorical and some having continuous data. After that for data visualization countplot is used for the categorical data and distplot have been used for continuous data.

Data Inputs- Logic- Output Relationships

No Inputs-Logic-Output Relationships have been found to exist between the features.

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

No assumption taken during model building.

Hardware and Software Requirements and Tools Used

In the hardware a laptop has been used along with an optical mouse.

In software excel, Python Jupyter notebook, have been used. Here is the table with list of libraries, import method and application of that method/function.

Library	Import method/function	Application
NumPy	array	Creating an array
pandas	DataFrame	importing DataFrame and other DataFrame related operations
matplolib, seaborn	countplot, distplot, scatterplots, pairplots	for data visualization
sklearn.preprocessing	OrdinalEncoder	used for encoding the categorical and ordinal string data
	power_transform	for removing skewness of both types positive and negative
	StandardScaler	for scaling of normalised data respectively
statsmodels.stats.outliers_influence	variance_inflation_factor	for calculating the variance inflation factor of every feature
sklearn.metrics	r2_score, mean_squred_error, mean_absolute_error	for calculating r2_score various regression model.
	joblib	for saving the final model
sklearn.ensemble	RandomForestRegressor,	For importing various classifiers for machine learning algorithms
sklearn.neighbors	KNeighborsRegressor	For importing the Regressor.
scipy.stats	zscore	used for outlier removal
sklearn.model_selection	train_test_split	for splitting dataset into training and testing data
	cross_val_score	for importing cross_val_score
	GridSearchCV	used for hyperparameter tuning

sklearn.tree	DecisionTreeRegressor	For importing DecisionTreeRegressor
sklearn.svm	SVR	For importing SVR
sklearn.kernel_ridge	KernelRidge	For importing the Regressor
sklearn.linear_model	ElasticNet	For importing the elasticnet

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

In the statistical approach various statistical techniques have been used to analyse the data. For descriptive statistics describe() function in python have been used to find out the mean, median, count, and percentiles for various features. And also, corr() function have been used to find out the correlation between the features. And to visualize the correlation coefficient data heatmap have been used. And for skewness detection in the features skew() function have been used and power_transform method has been used to remove the skewness from the features. For detecting the multicollinearity between the columns VIF method have been used and features having higher VIF factor have been removed to reduce the multicollinearity.

In the Analytic approach multiple algorithms have been used to check for the accuracy score for measuring the performances of the algorithms on the dataset.

Testing of Identified Approaches (Algorithms)

Listing down all the algorithms used for the training and testing.

Algorithms used in the training and testing are:

RandomForestRegressor

SupportVectorRegressor

KNeighborsRegressor

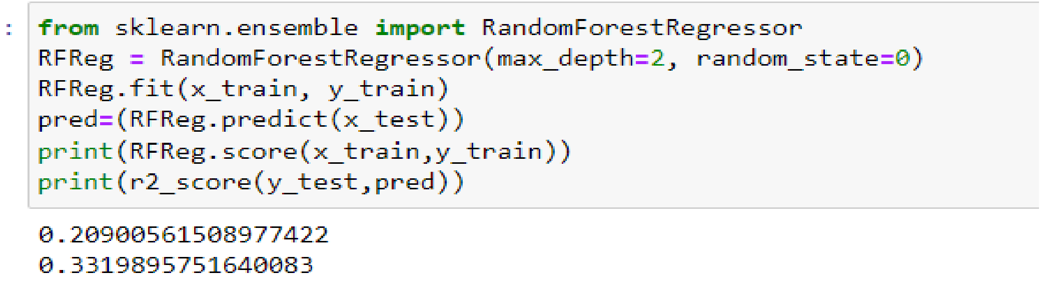
ElasticNet

DecisionTreeRegressor

Run and Evaluate selected models

RandomForestRegressor is the first algorithm used for the Regression which is a type of ensemble technique which is generated using a random selection of attributes at each node to determine the split. Every DecisionTree has high variance but when we combine all of them together in parallel then the resultant variance is low as each decision tree gets perfectly trained on that particular sample data and hence the output doesn’t depend on one decision tree but multiple decision trees. In Regression the final output is the mean of all outputs. This part is Aggregation. Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

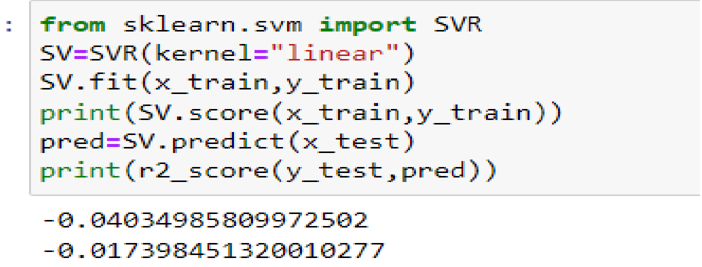
Evaluation metrics and classification report for the algorithm is shown below.



```
:  from sklearn.ensemble import RandomForestRegressor
   RFReg = RandomForestRegressor(max_depth=2, random_state=0)
   RFReg.fit(x_train, y_train)
   pred=(RFReg.predict(x_test))
   print(RFReg.score(x_train,y_train))
   print(r2_score(y_test,pred))

   0.20900561508977422
   0.3319895751640083
```

Second Algorithm is SupportVectorRegressor that have been used for the calculation is Support Vector Regressor. This algorithm is used for both categorical and continuous type of data. It constructs a hyperplane in multidimensional manner in an iterative manner, which is used to minimize the error. The core idea of the SVM is to create a hyperplane that best divides the dataset into classes. A hyperplane is a decision plane which separates between a set of objects having different class memberships. The main objective of the SVM is to segregate the given dataset in the best possible way. The distance between the either nearest points is known as the margin. SVM searches for the maximum marginal hyperplane in following steps:
Generate hyperplanes which segregates the classes in the best way.
Select the right hyperplane with the maximum segregation from the nearest data points.
SVM is implemented in practice using a kernel. A kernel transforms an input data space into the required form. It converts non-separable problem to separable problems by adding more dimension to either nearest point is more useful in non-linear separation.
Evaluation metrics and classification report for the algorithm is shown below.

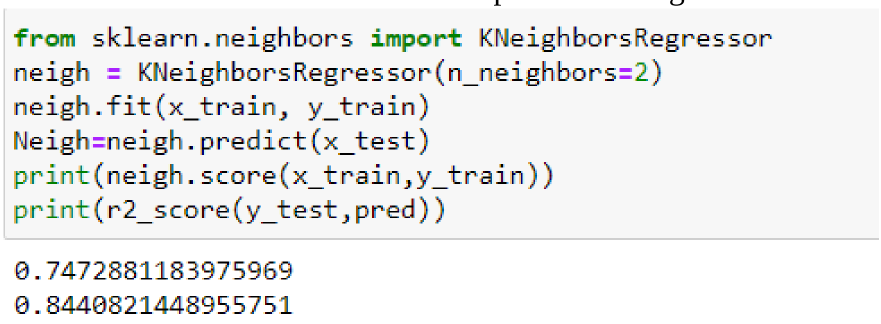


```
:  from sklearn.svm import SVR
   SV=SVR(kernel="linear")
   SV.fit(x_train,y_train)
   print(SV.score(x_train,y_train))
   pred=SV.predict(x_test)
   print(r2_score(y_test,pred))

   -0.040349858099725002
   -0.017398451320010277
```

The third algorithm used is the KNeighborsRegression, it is very simple to understand versatile and one of the topmost machine learning algorithms. In KNeighborsRegression is the number of nearest neighbors. The number of neighbors is the core deciding factor. It uses ‘feature similarity’ to predict the values of the new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set. K is generally an odd number if the number of classes is 2. When k=1 then algorithm is known as the nearest neighbour algorithm. Suppose P1 is the point, for which label needs to predict. First, you find the k-closest point to P1 and then classify points by majority vote of its K-neighbors. Each object

votes for their class and the class with the most votes are taken as the prediction. For finding the closest similar points, we need to find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance. KNeighborsRegressor has the following basic steps:
Calculate the distance
Find the closest neighbors
Vote for labels.
KNeighborsRegressor performs best with the lower number of features than large number of features.
Evaluation metrics and classification report for the algorithm is shown below.



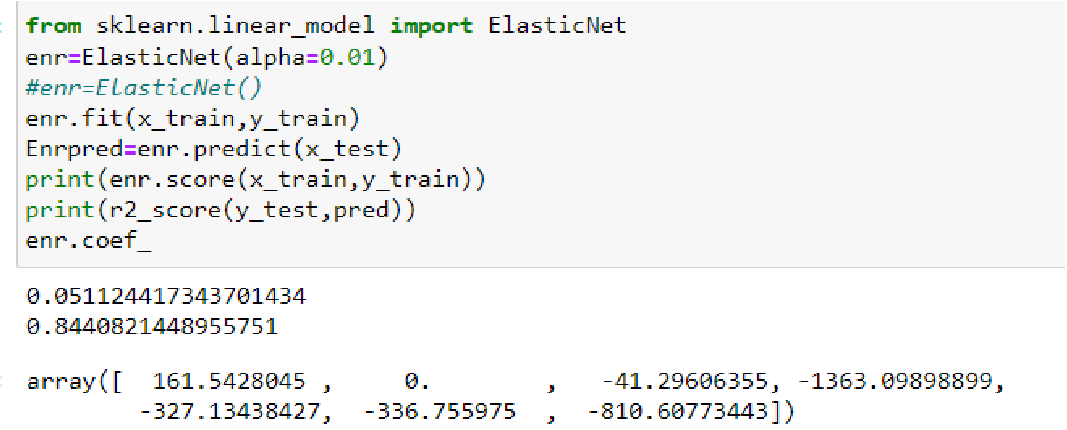
```
from sklearn.neighbors import KNeighborsRegressor
neigh = KNeighborsRegressor(n_neighbors=2)
neigh.fit(x_train, y_train)
Neigh=neigh.predict(x_test)
print(neigh.score(x_train,y_train))
print(r2_score(y_test,pred))
```

```
0.7472881183975969
0.8440821448955751
```

The Fourth Algorithm is elasticnet which is the modification of Linear Regression which shares the same hypothetical function for prediction. The Linear Function suffers from overfitting and can’t deal with collinear data. This makes the model more complex with inaccurate prediction on the test set. Such a model with high variance does not generalize on the new data. So, to deal with these issues, we include both L-1 and L-2 regularization to get the benefits of both Ridge and Lasso at the same time. It performs feature selection and also makes hypothesis simpler. The modified cost function for ElasticNet Regression is given below:

$$\frac{1}{m}\left[\sum_{l=1}^{m}\left(y^{(i)}-h\left(x^{i}\right)\right)^{2}+\lambda_1\sum_{j=1}^{n}w_j+\lambda_2\sum_{j=1}^{n}w_j^2\right]$$

Here wj represents the weight for jth feature.
n is the number of features in the dataset.
Lambda1 is the regularization strength for L-1 norm.
Lambda2 is the regularization strength for L-2 norm.

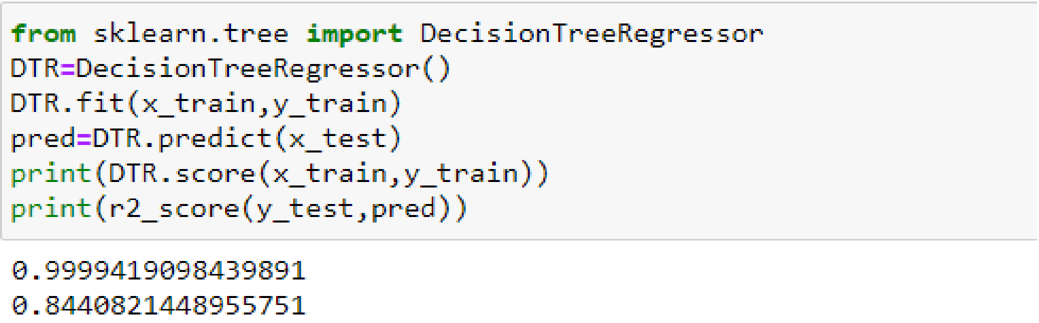


```
from sklearn.linear_model import ElasticNet
enr=ElasticNet(alpha=0.01)
#enr=ElasticNet()
enr.fit(x_train,y_train)
Enrpred=enr.predict(x_test)
print(enr.score(x_train,y_train))
print(r2_score(y_test,pred))
enr.coef_
```

```
0.051124417343701434
0.8440821448955751

array([  161.5428045 ,     0.        ,   -41.29606355, -1363.09898899,
        -327.13438427,  -336.755975  ,  -810.60773443])
```

The fifth algorithm is DecisionTreeRegressor which is uses flowchart-like tree structure or is a model of decisions and all of their possible results, including outcomes, input costs, and utility. DecisionTree Algorithm falls under the category of supervised learning algorithms. Its branches/edges represent the truth/falsity of the statement and take makes a decision based on Decision Tree observes the features of an object and trains a model in the structure of tree in order to produce meaningful continuous output. Continuous output means that output/result is not discrete. Evaluation metrics and r2_score is given below:



```
from sklearn.tree import DecisionTreeRegressor
DTR=DecisionTreeRegressor()
DTR.fit(x_train,y_train)
pred=DTR.predict(x_test)
print(DTR.score(x_train,y_train))
print(r2_score(y_test,pred))
```

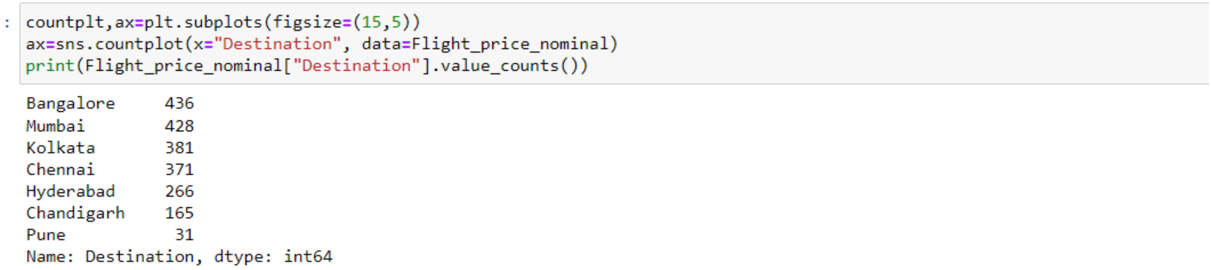
```
0.9999419098439891
0.8440821448955751
```

Key Metrics for success in solving problem under consideration

Key Metrics for success in solving problem under consideration is r2_score which is used for all regression algorithms used and are having more than 0.8 value.

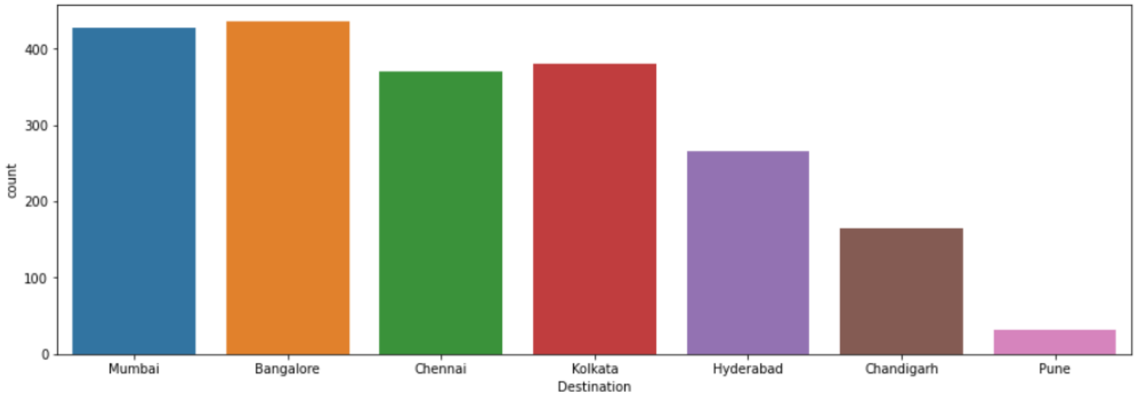
Visualizations

Visualization plots that have been used in the project includes distplots, countplots, scatterplots and boxplots also. Distplots and countplots are used in Univariate analysis whereas Scatterplots are used in Bivariate Analysis. And boxplots are also used in univariate analysis for finding out the outliers. Distplots are used for continuous data whereas countplot is used for nominal data and scatterplots can be used with both continuous as well as categorical data. Below is the countplot shown that are used in the project:

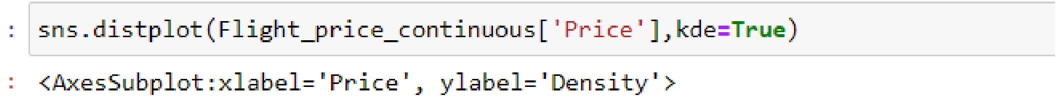


```
countplt,ax=plt.subplots(figsize=(15,5))
ax=sns.countplot(x="Destination", data=Flight_price_nominal)
print(Flight_price_nominal["Destination"].value_counts())
```

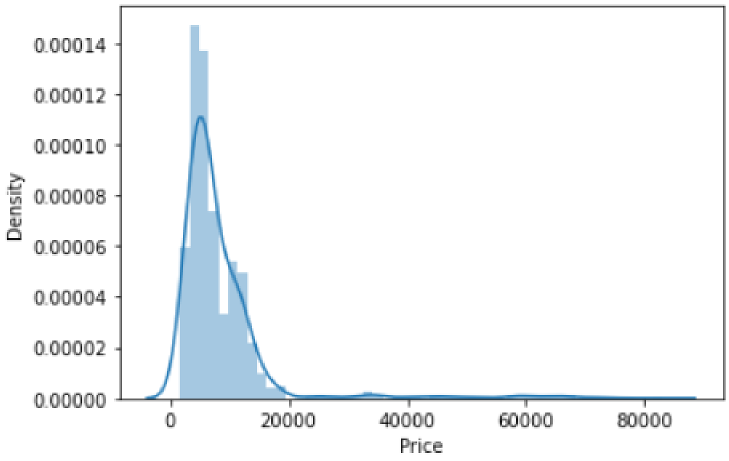
```
Bangalore     436
Mumbai        428
Kolkata       381
Chennai       371
Hyderabad     266
Chandigarh    165
Pune           31
Name: Destination, dtype: int64
```



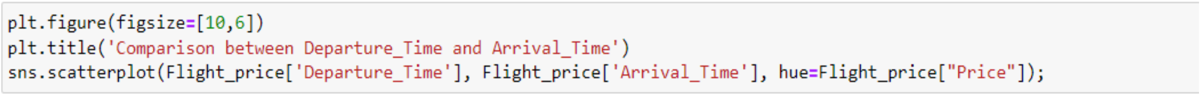
This countplot shows the relationship between feature Destination and its count with respect to Price.



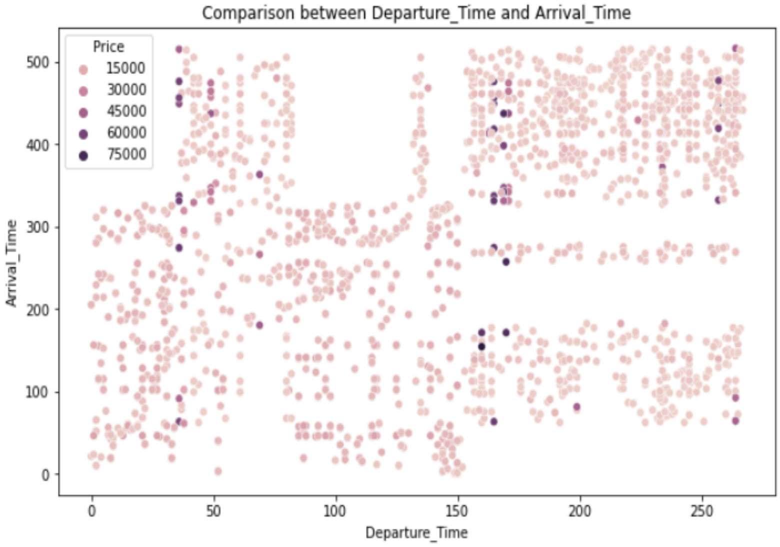
```
: sns.distplot(Flight_price_continuous['Price'],kde=True)
: <AxesSubplot:xlabel='Price', ylabel='Density'>
```



This plot shows the relationship between feature Price and its frequency. The plot here is positively skewed.



```
plt.figure(figsize=[10,6])
plt.title('Comparison between Departure_Time and Arrival_Time')
sns.scatterplot(Flight_price['Departure_Time'], Flight_price['Arrival_Time'], hue=Flight_price["Price"]);
```



This is scatterplot which is showing relationship between two variables that is ‘Departure_Time’ and ‘Arrival_Time’ Here the data is scattered randomly, so there is no correlation between the variables.

Interpretation of the Results

From the Data Visualization part, we can say that most of the features having skewed data that is negatively skewed and correlation between the features is negligible some distributions are also multimodal. The outliers can be found in some of the plots like ‘Duration’,’Total_Stops’,’Price’. From the scatterplot data we can say that the correlation can not be found among features. From plot we can say that 'Vistara' is having highest number of flights for any 'Destination' followed by 'Air india' and followed by others. From the plots we can say that 'Vistara' is having the highest Price for any destination followed by 'Air india' and followed by others.

From the preprocessing part we found that the dataset has some missing, null and empty values. And all the datatype are matching with the datatypes of the features in the dataset. Some datatype in the dataset were having string and object datatype which needs to be converted into numeric type, so encoding was also done to some features. From the feature scaling we found that none of the features is having high VIF which means that no removal of column is required. In the train_test_split method DecisionTreeRegressor have been used for finding out the best random state to do further r2_score checking using other algorithms. Using cross-validation score has been selected as the best model for the Hyperparameter tuning. After Hyperparameter tuning was done, best model was selected and it was called as Final Model which is further used for prediction of test data.

CONCLUSION

Key Findings and Conclusions of the Study

From the complete project we came to know that starting from importing the dataset and to the end of the model building and checking its r2_score there come a lot of challenges. Challenges include starting from data cleaning which includes checking for null and empty values and then passing appropriate values to the missing data using Imputation techniques. Then other pre-processing techniques like EDA analysis and Data Visualization. In data visualization we have to find out whether the data is normally distributed or skewed (positively or negatively). In the project the positive skewness is found in lot of features like ‘Price’ when the data is positively skewed mean is greater than median and mode. From the boxplot visualization we can find the presence of outliers in the dataset. In our dataset there are some outliers present in various features like ‘Total_Stops’,’ Duration’,’ Price’. Outliers presence makes the model learn dataset with decrease in accuracy and all other metrics, which will decrease the overall performance of the model. Then comes the scatterplots which shows the relationship between two feature variables whether they are positively correlated or negatively correlated. In our project the data is found to be not correlated in most of the times. Then from the descriptive statistics point of view, the various statistical finding can be analysed like mean, median, mode, count, various quartiles and maximum value for each feature. In the multivariate analysis correlation table and heatmap gives us an idea about the correlation coefficient of various features. Then the outlier removal is done using z-score to find out the dataset without outliers. After that skewness detection and skewness removal using the power_transform function from sklearn.preprocessing library was done to make the data normal and then the StandardScaler was used to make the data scaled, so it can be fed to the machine learning models for training and testing. Then VIF checking was done for checking the multicollinearity issue in the dataset. Some of the features have found to be having high VIF so they were removed from the dataset to make it more easily used by the machine learning model. Now in the train_test_split we used DecisionTreeRegressor for checking the best random state for other algorithms to work upon. After finding the best random state various Regressor algorithms were used to find out the evaluation metrics for different algorithms. And findings in the model selection was DecisionTreeRegressor performance was best for the dataset. And then Hyperparameter tuning was used to find out the for tuning the best parameters in this algorithm and then the r2_score was found out again. And the increase in r2_score was seen in the final model.

Learning Outcomes of the Study in respect of Data Science

Learnings includes deep analysis of various data visualization techniques to get insights of the data and the variation of data in features with respect to label. In univariate analysis we have found that countplots that are used which helps in visualizing the data clearly. From the statistical point of view, various descriptive parameters like mean, median, mode, standard deviation, minimum and maximum value are observed, which can be further used to predict the skewness and outliers in the dataset. StandardScaler technique used in the dataset for the scaling of the data which converts the data to scaled form so that it can be read effectively and easily by the machine learning model. Using various Machine Learning models for checking the r2_score of models to find the best model for Hyperparameter tuning. Hyperparameter tuning is done to choose the best parameters from the list of various parameters of the model. Challenges in this project include feature selection, Exhaustive computation for Hyperparameter tuning since number of rows are very large. Selecting the best algorithm for the Final Model.

Limitations of this work and Scope for Future Work

Limitations of this work is that more robust model can be build using some more advanced techniques in machine learning model building and Exhaustive EDA can also improve the accuracy of the model using some more techniques for the EDA analysis can solve the problem of less accuracy. Future work can be done in the pre-processing and data visualization part for analysing data more effectively.