Flight Ticket Price Prediction

Karan Singh Kushwah (B20CS024)

Abstract – ***This report outlines my findings and analysis while developing a regression model for predicting flight ticket prices. The dataset contains flight ticket prices for several airlines and between numerous cities throughout the months of March and June 2019. In this report, I compared the results of multiple regression models and hyperparameter tuning.***

I. INTRODUCTION

Flight ticket prices can be something hard to guess, today we might see a price, check out the price of the same flight tomorrow, it will be a different story. We might have often heard travelers saying that flight ticket prices are so unpredictable. Here you will be provided with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities.

Datasets

The file Dataset.xlsx is used as the dataset with 80% as training data and the rest 20% as testing data. The train dataset contains 10683 rows where each row represents a data consisting of following features: -

1 . Airline: The name of the airline
2 . Date_of_Journey: The date of the journey
3 . Source: The source from which the service begins
4 . Destination: The destination where the service ends
5 . Dep_Time: The time when the journey starts from the source.
6. Route: The route of the flight
7 . Arrival_Time: Time of arrival at the destination.
8 . Duration: Total duration of the flight.
9 . Total_Stops: Total stops between the source and destination.
10 . Additional_Info: Additional information about the flight

Target :- *11. Price : The price of the ticket*

II. METHODOLOGY

OVERVIEW

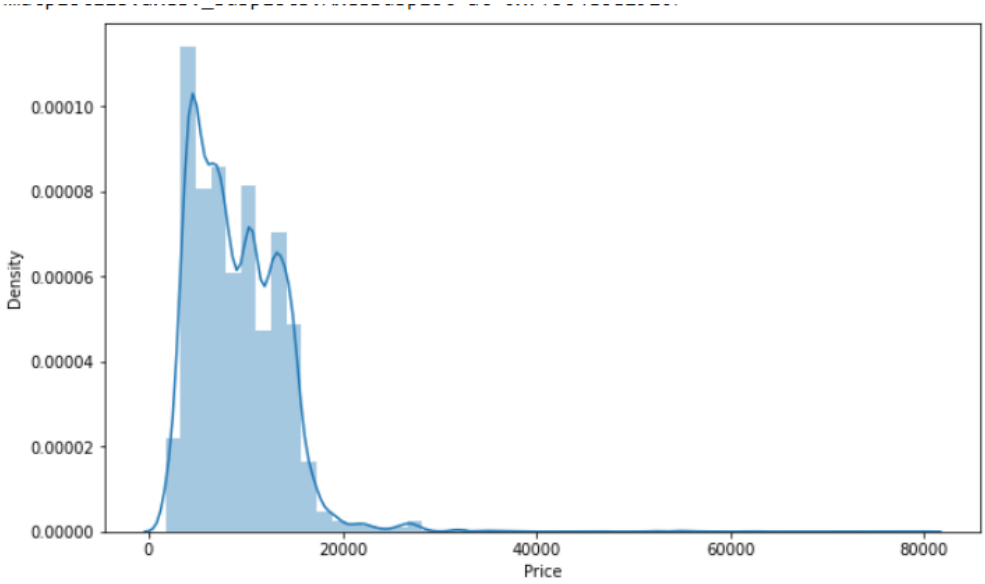
Following are the regression models that I used for the given problems :-

- *Random Forest Regressor*
    - *A. Without Hyper-parameters*
    - *B. With Hyper-parameters*
- *Linear Regression*
- *Decision Tree Regressor*
- *KNeighbour Regression*
- *Gradient Boosting Regressor*

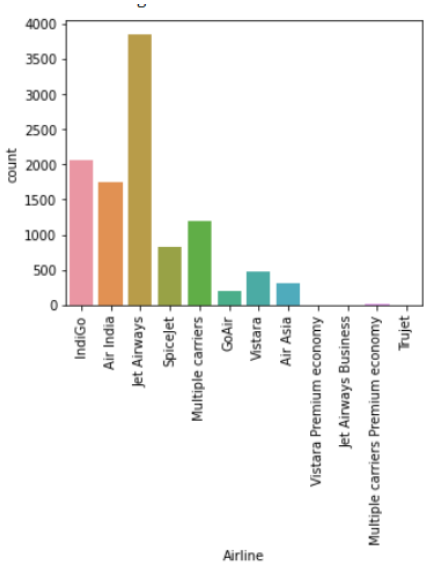
Exploring the dataset and pre-processing

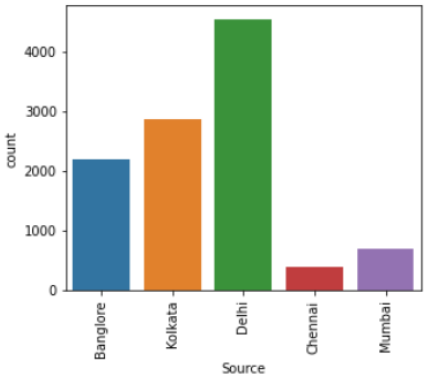
On checking for NaN values in the dataset, no such values were found.After this, we got to know that there is only one feature with a datatype of Integer. So, we have to do the preprocessing on the other features whose data type is object. It started by extracting the day, month, and year of the journey from the Date_of_Journey column, extracting the hour and minute of departure and arrival of the flight from the Dep_Time and Arrival_Time columns, respectively, and also extracting the hour and minutes of flight from the Duration column. The route feature is extracted using the same method. After this, we took care of missing values in the dataset by either dropping the column or replacing the missing value with another suitable number. The next preprocessing step involves encoding categorical columns either using LabelEncoder or OneHotEncoder based on the number of unique entities present in each column.

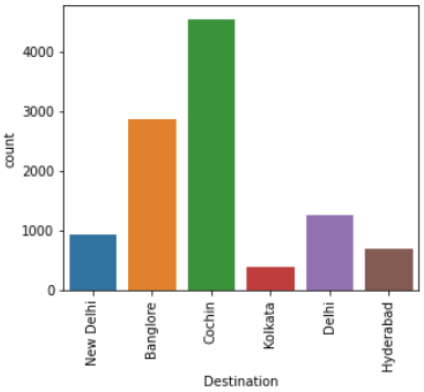
First , we visualize the Target feature of the dataset -

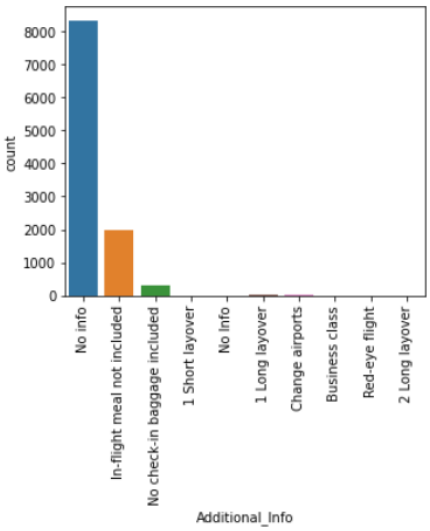


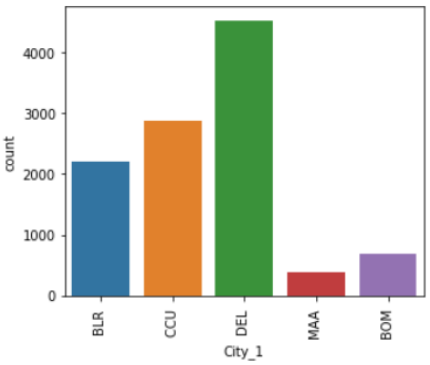
Now, we visuilise the categorical data by Countplot -

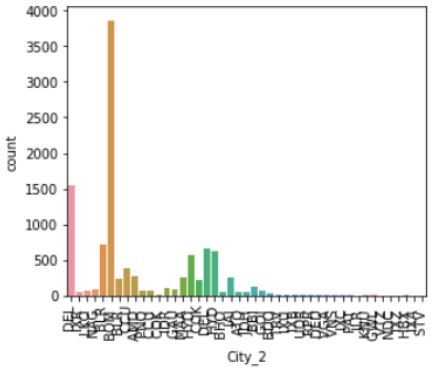

count
Airline
IndiGo
Air India
Jet Airways
SpiceJet
Multiple carriers
GoAir
Vistara
Air Asia
Vistara Premium economy
Jet Airways Business
Multiple carriers Premium economy
Trujet

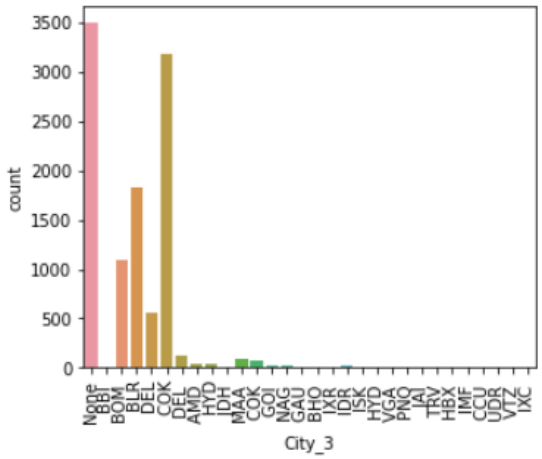

count
Source
Banglore
Kolkata
Delhi
Chennai
Mumbai


count
Destination
New Delhi
Banglore
Cochin
Kolkata
Delhi
Hyderabad


count
Additional_Info
No info
In-flight meal not included
No check-in baggage included
1 Short layover
No Info
1 Long layover
Change airports
Business class
Red-eye flight
2 Long layover


count
City_1
BLR
CCU
DEL
MAA
BOM


count
City_2

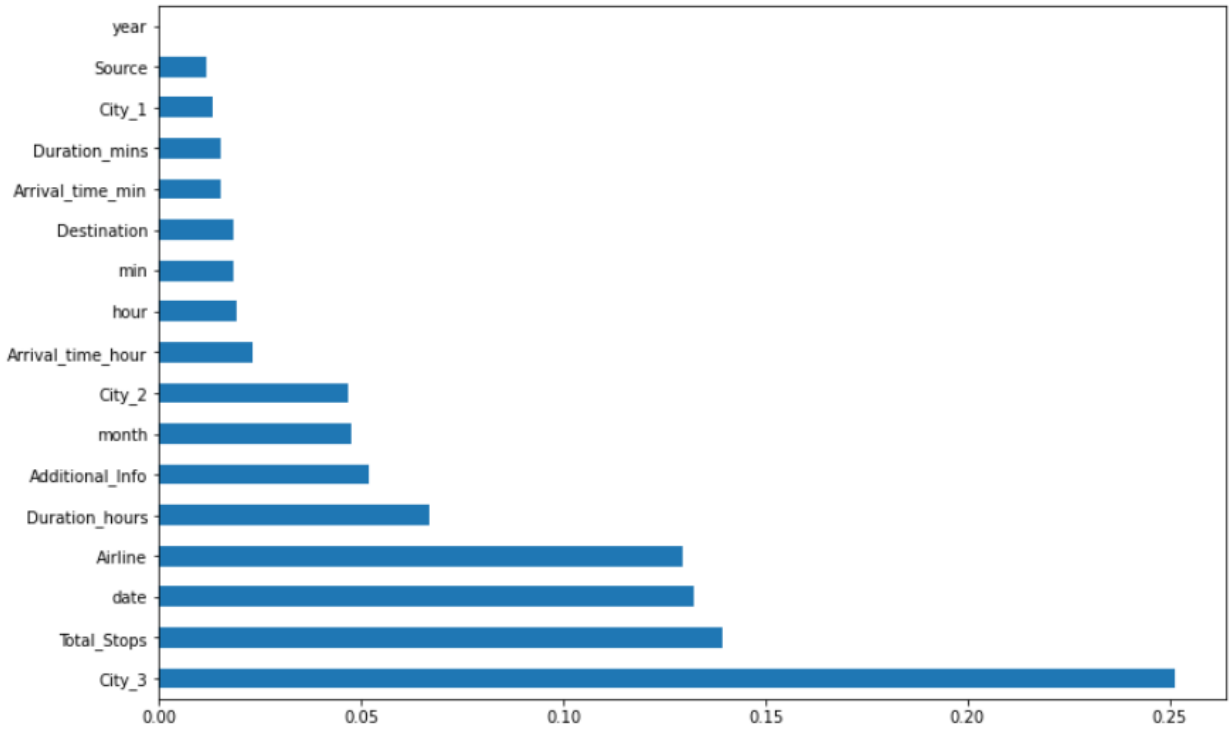


Feature Selection and Reduction

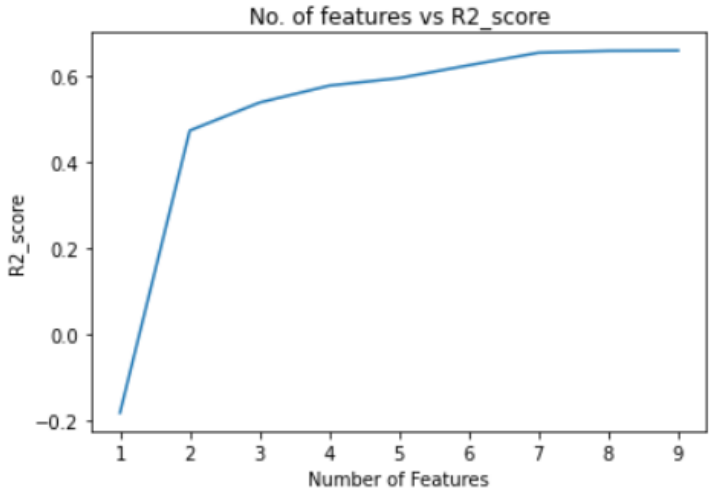
Finding out the best feature which will contribute and have a good relation with the target variable. Following are some of the feature selection methods,

1.Heatmap
2.Feature importance
3.SelectKBest

The Feature Importance method is used for the given problem. Imported Extratreeregressor to get important features from the dataset -



After this , we apply PCA for feature reduction and calculate the r2 score for the number of features selected between the range of 1 to 10 -



From the above graph , it's clear that feature reduction will not be beneficial for this problem. So , i dropped this idea

Implementation of Regression algorithms

- Random Forest Regressor : Random Forest regressors use boosting ensemble methods to train upon various decision trees and produce aggregated results. It is one of the most used machine learning algorithms. Below are the results that we got from this model -

```
R2 Score for Random Forest Regressor model(For Training Data) :-  0.9799358001157015
R2 Score for Random Forest Regressor model(For Testing Data) :-  0.8774599023395623
Mean absolute error for Random Forest Regressor model :-  637.2354845005661
Mean squared error for Random Forest Regressor model :-  2601045.484837802
Root mean squared error for Random Forest Regressor model :-  1612.775708162112
```

- Linear Regression : Linear regression is a linear approach for modeling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables).Below are the results that we got from this model -

```
R2 Score for Linear Regression model(For Training Data) :-  0.49050820612987833
R2 Score for Linear Regression model(For Testing Data) :-  0.5142388318738056
Mean absolute error for Linear Regressor model :-  2243.5973725799204
Mean squared error for Linear Regressor model :-  10310803.705782369
Root mean squared error for Linear Regressor model :-  3211.044021152991
```

- Decision Tree Regressor :Decision tree learning is one of the predictive modeling approaches used in statistics, data mining and machine learning. Algorithms for constructing decision trees usually work top-down, by

choosing a variable at each step that best splits the set of items.Below are the results that we got from this model -

```
R2 Score for Decision Tree Regression model(For Training Data) :-  0.996140951067578
R2 Score for Decision Tree Regression model(For Testing Data) :-  0.759841096395939
Mean absolute error for Decision Tree Regression model :-  726.2498179927196
Mean squared error for Decision Tree Regression model :-  5097631.25531288
Root mean squared error for Decision Tree Regression model :-  2257.7934483280087
```

- KNeighborsRegressor :Regression based on k-nearest neighbors. The target is predicted by local interpolation of the targets associated with the nearest neighbors in the training set.Below are the results that we got from this model -

```
R2 Score for KNeighbours Regressor model(For Training Data) :-  0.7637245801547251
R2 Score for KNeighbours Regressor model(For Testing Data) :-  0.6452544578402343
Mean absolute error for KNeighbours Regressor model :-  1660.8936037441497
Mean squared error for KNeighbours Regressor model :-  7529856.008911077
Root mean squared error for KNeighbours Regressor model :-  2744.058310042095
```

- Gradient Boosting Regressor :Gradient Boosting is a method for optimising arbitrary differentiable loss functions by building an additive model in a forward stage-wise manner. A regression tree is fitted on the negative gradient of the supplied loss function at each level.Below are the results that we got from this model -

```
R2 Score for Gradient Boosting Regressor model(For Training Data) :-  0.8343761239599471
R2 Score for Gradient Boosting Regressor model(For Testing Data) :-  0.8573844722032504
Mean absolute error for Gradient Boosting Regressor model :-  1197.558048464638
Mean squared error for Gradient Boosting Regressor model :-  3027168.100285082
Root mean squared error for Gradient Boosting Regressor model :-  1739.875886460032
```

Analysis ;- From the above results , we are able to see that Random Forest Regression performs better than any other model and gives accuracy equal to 87.74 on test data.Now, we perform some operation on Random Forest Regression model to get best hyper parameters.

Hyper-parameter tuning

Random Forest regressor model is selected for hyperparameter tuning because it perform better then other models.Two types of approach is used to get hyer-parameters -
1.GridSearchCV : It helps in fitting your estimator (model) to your training data by looping through predefined hyperparameters. Finally, you can choose the best parameters from the hyperparameters listed.n_estimators, max_depth, max_samples, and min_sample_split are used in RandomForestRegressor and report the outcomes of the operation. We vary n_estimators from 10 to 90 with a 20-point interval, max_depth from 0 to 4, and min_samples_split from 2.4 to 10.

Best Parameters we got -

```
{'max_depth': None,
 'max_samples': 1000,
 'min_samples_split': 2,
 'n_estimators': 90}
```

Now, below are the results that we got after applying this hyper-parameters -

```
By GridSearchCV
R2 Score for Random Forest Regressor model with hyper-parameters(For Training Data):-  0.8540847949940024
R2 Score for Random Forest Regressor model with hyper-parameters(For Testing Data):-  0.8388892503177395
Mean absolute error for Random Forest Regressor model with hyper-parameters :-  960.8868973330364
Mean squared error for Random Forest Regressor model with hyper-parameters :-  3419749.0945461364
Root mean squared error for Random Forest Regressor model with hyper-parameters :-  1849.2563625809528
```

We can see that this decreases the r2 score instead of increasing it . We varied the parameters but it didn't give much difference . So , we use another approach.

2.RandomizedSearchCV :RandomizedSearchCV is very useful when we have many parameters to try and the training time is very long. The parameters of the estimator used to apply these methods are optimized by cross-validated search over parameter settings. Finally, you can choose the best parameters from the hyperparameters listed.n_estimators, max_depth, max_features, min_samples_leaf, and min_samples_split for RandomForestRegressor, and report the results of operations We vary n_estimators from 100 to 1200 with an interval of 12. max_depth is varied between 5 and 30 with an interval of 5. Two values for max_features (auto, sqrt) are selected. min_samples_leaf is varied between 1 to 10 with an interval of 2, and last, min_samples_split is varied between 2 to 100 in multiples of 5.

Best Parameters we got -

```
{'max_depth': 25,
 'max_features': 'sqrt',
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'n_estimators': 1000}
```

Now, below are the results that we got after applying this hyper-parameters -

```
By RandomizedSearchCV
R2 Score for Random Forest Regressor model with hyper-parameters(For Training Data):-  0.9785878382642988
R2 Score for Random Forest Regressor model with hyper-parameters(For Testing Data):-  0.9096612819417819
Mean absolute error for Random Forest Regressor model with hyper-parameters :-  716.6492359147827
Mean squared error for Random Forest Regressor model with hyper-parameters :-  1917536.538631515
Root mean squared error for Random Forest Regressor model with hyper-parameters :-  1384.7514356849445
```

Finally , we got the max r2 score with 90.96 % by applying the hyper-parameters.

III. RESULTS AND ANALYSIS

Table 1. Evaluation Metrics over primitive models

Model Name	R2 Score for training data	R2 Score for testing data	Mean absolute error
Random Forest Regressor	**0.97993**	**0.87745**	**637.23**
Linear Regression	0.49050	0.51423	2243.59
Decision tree regressor	0.99614	0.75984	726.24
KNeighborsRegressor	0.76372	0.64525	1660.89
Gradient boosting regressor	0.83437	0.85738	1197.55

Table 2. Evaluation Metrics over hyperparameter tuned on Random Forest Regressor model

Name	R2 Score for training data	R2 Score for testing data	Mean absolute error
GridSearchCV	0.85408	0.83888	960.88
RandomizedSearchCV	**0.97858**	**0.90966**	**716.64**

Conclusion -

From the above table , we are able to see Random Forest regressor performed better than any other model. Also, Gradient Boosting regressor is performing better compared to remaining models.WE can see that , hyper-parameter tuning help to improve efficiency for the model.But , it gives relatively more mean absolute error then the primitive model