Name: Nagda Yash Narendra

Topic: Predict the Flight Ticket Price

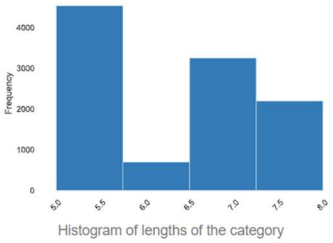
In the first step, dataset was analyzed and it was seen that only 1 entry had NULL values. Hence that entry was dropped. Next, profile report of the dataset was made using the Pandas Profiling. From the report, the following points were noted:

1. Flight source is only from 5 cities: Delhi, Mumbai, Kolkata, Chennai, Bangalore.

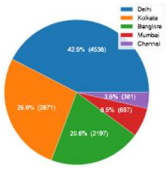
Common Values

Value	Count	Frequency (%)
Delhi	4536	42.5%
Kolkata	2871	26.9%
Banglore	2197	20.6%
Mumbai	697	6.5%
Chennai	381	3.6%

Length



Pie chart

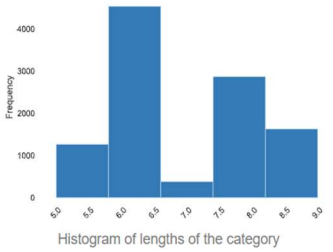


2. Flight Destination is only to 6 cities: Cochin, Bangalore, Delhi, New Delhi, Kolkata, Hyderabad.

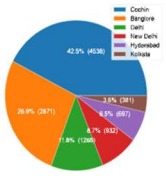
Common Values

Value	Count	Frequency (%)
Cochin	4536	42.5%
Banglore	2871	26.9%
Delhi	1265	11.8%
New Delhi	932	8.7%
Hyderabad	697	6.5%
Kolkata	381	3.6%

Length



Pie chart

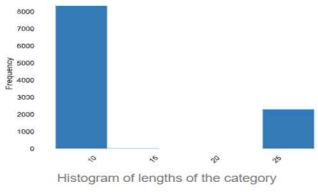


3. Additional Information has maximum values has “No Info”.

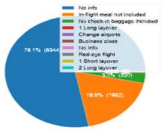
Common Values

Value	Count	Frequency (%)
No info	8344	78.1%
In-flight meal not in…	1982	18.6%
No check-in bagga…	320	3.0%
1 Long layover	19	0.2%
Change airports	7	0.1%
Business class	4	< 0.1%
No Info	3	< 0.1%
Red-eye flight	1	< 0.1%
1 Short layover	1	< 0.1%
2 Long layover	1	< 0.1%

Length

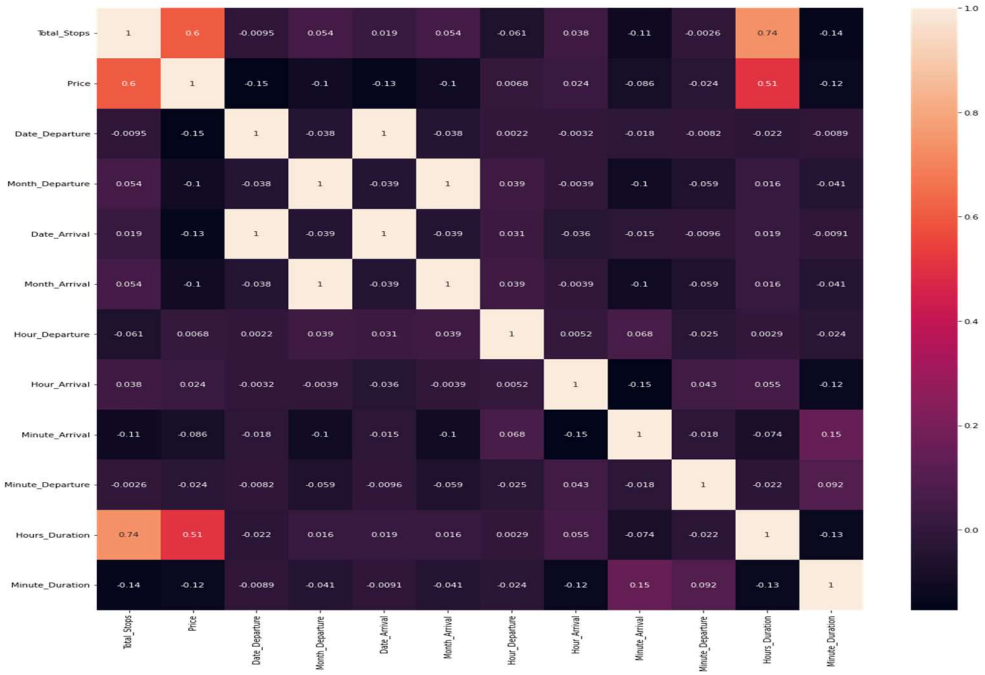


Pie chart

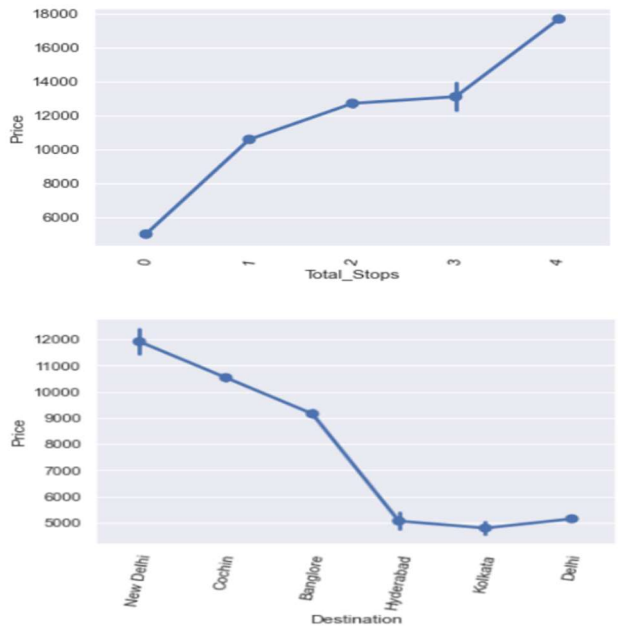


The above information gained is used further in pre-processing. In the next step, the time, date, month and year of journey were separated. Similarly, the time, date, month and Year of Arrival were also separated. Since, there was ambiguity in entries of these features, these were first made into a single format. From the year of journey, we see that all dates correspond to year 2019. Thus, columns “Addition_Info”, “Year_Departure”, and “Date of Journey” were dropped in next step.

In the next step, label encoding was performed from scratch to encode the number of stops. The features “Dep_Time”, “Arrival_Time”, “Duration” were replaced with appropriate minutes and hours column. In the next step, correlation matrix was obtained as follows:



From the matrix, we see that Price Mainly depends upon the Total Number of stops, and Hours of Duration. To get an idea of variation of Price with respect to some of the common dependent variables, following plots were obtained:



From the above plots, we see that the Price varies a lot with Number of Stops, Airline, Source and Destination. Hence One Hot Encoding was performed on Source, Destination and Airline features. Further, we see that flight price increases to some extent with duration in hours of the journey. This can be because higher duration may imply flight covers more distance, hence higher Price. In the next step, we see that there are 52 different airports covered under all Flight Routes. Since the combinations possible between route cities is very large, it may not be possible to cover each combination in our model. Hence, Feature Routes was dropped.

In the next step, train data was split into Train Set(64%), Cross-Validation Set(16%) and Test-Set(20%). The error metrics used for evaluating the model are $R^2\ score$ and RMSE. For each model, firstly GridSearchCV with scoring metric as $R^2\ score$ was used as evaluation metric, along with 5 fold cross-validation. The Optimal Models along with the scores are attached in the report.

$$Root\ Mean\ Squared\ Error\ (RMSE) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_{True}[i] - Y_{pred}[i])^2}$$

$$R^2\ score = 1 - \frac{\sum_{i=1}^{n}(Y_{True}[i] - Y_{pred}[i])^2}{\sum_{i=1}^{n}(Y_{True}[i] - Y_{True}[mean\ value])^2}$$

Firstly, **Random Forest Regressor** was implemented. The optimal parameters were obtained from GridSearch CV, and the model trained on those parameters gave the following Scores:

```
RandomForestRegressor(max_depth=25, min_samples_leaf=2, min_samples_split=10,
                      n_estimators=1000)
Best Score: 0.8112345921720578
                      On Random-Forest Model, RMSE scores are:
                      Train: 1423.197645962446
                      CV: 1961.3531998152441
                      Test: 2023.7773135208254
                      On Random-Forest Model, R2 scores are:
                      Train: 0.9066120310913595
                      CV: 0.8066505349973406
                      Test: 0.8048107888010303
```

Thus, we see that R2 score for CV set is nearly equal to that obtained after Grid-Search CV. Also, the training R2 score is higher and RMSE is lower than that of CV set.

Secondly, **SVM Regressor** was implemented. The optimal parameters were obtained from GridSearch CV, and the model trained on those parameters gave the following Scores:

```
SVR(C=100, kernel='linear')
Best Score: 0.5337493844881488
      On SVR-Model, RMSE scores are:
      Train: 3179.616671077659
      CV: 2978.6248962580707
      Test: 3169.091341587801
      On SVR-Model, R2 scores are:
      Train: 0.5338668096774218
      CV: 0.5540738013260698
      Test: 0.5213700438725375
```

Here, we see that R2 score obtained for all sets is much lower than that of Random Forest Regressor. Thus, it may not be an optimal Regressor for the given dataset.

The Third Model Implemented was **Logistic Regression**. Again, on the optimal Paramters, the R2 Score on CV set is much lower than that of Random Forest. Here, we can also see that training R2 score is a lot higher than that of CV set. Thus on optimal paramters, we can see a overfit in Logistic Regressor.

```
LogisticRegression(C=3, max_iter=1000)
Best Score: 0.6114476555177106
```

```
On lr-Model, RMSE scores are:
Train: 1973.2818845008535
CV: 3225.180188923402
Test: 2982.627143181942
On lr-Model, R2 scores are:
Train: 0.8204692875234907
CV: 0.47719549053040733
Test: 0.5760366703705377
```

The Fourth Model Implemented was **AdaBoost Regressor**. Here, too, R2 score obtained on CV Set is much lower than that of Random Forest Regressor.

```
AdaBoostRegressor(learning_rate=0.1, n_estimators=100)
Best Score: 0.5631231236262297
```

```
On AdaBoost Regressor Model, RMSE scores are:
Train: 2955.5873389980684
CV: 3048.4324288534217
Test: 3124.7088177247283
On AdaBoost Regressor Model, R2 scores are:
Train: 0.5972383683822353
CV: 0.5329272783745121
Test: 0.5346824116415754
```

The Fifth Model Implemented was **LightGBM Regressor**. In this model, we see that the R2 Score is better than the other models implemented and is nearly 0.81 and almost equal to random forest model, on the CV Data.

```
LGBMRegressor(n_estimators=300)
Best Score: 0.8028686063030591
```

```
On LightGBM Regressor Model, RMSE scores are:
Train: 1331.8557013045258
CV: 1917.2363478061718
Test: 1891.0915778770993
On LightGBM Regressor Model, R2 scores are:
Train: 0.9182147764949948
CV: 0.8152507576775812
Test: 0.8295662955839556
```

The Sixth Model Implemented was **XGBoost Regressor**. In this model, again we see that the R2 Score is better than the other models implemented and is nearly 0.81 and almost equal to random forest model, on the CV Data.

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.1, max_delta_step=0,
             max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=100, n_jobs=8,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
Best Score: 0.8172550137586679
```

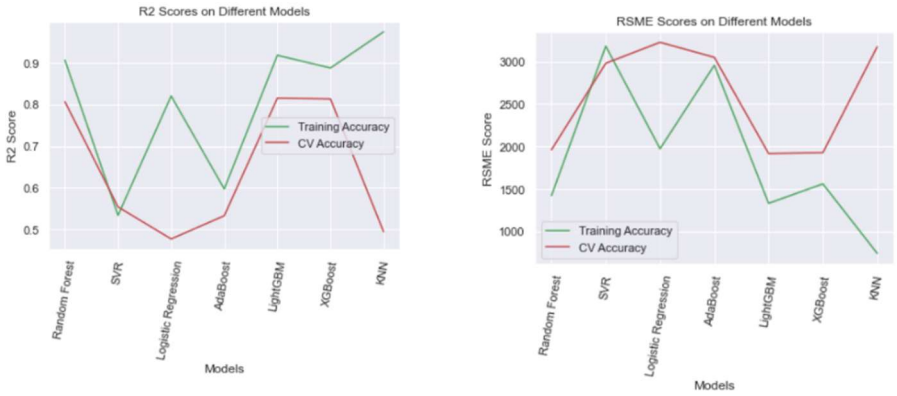
```
On XgBoost Regressor Model, RMSE scores are:
Train: 1560.6786792091902
CV: 1927.26586684682
Test: 1897.2844008466095
On XgBoost Regressor Model, R2 scores are:
Train: 0.8876979998219784
CV: 0.8133127674591785
Test: 0.8284482175839685
```

The Seventh Model Implemented was **KNN Regressor**. Here, we can see that the accuracy on CV data was lowest among all models. Thus, KNN Regressor is not a good option for implementation.

```
KNeighborsRegressor(weights='distance')
Best Score: 0.5300260169085915
```

```
On KNN-Model, RMSE scores are:
Train: 744.6955572488342
CV: 3171.321879752909
Test: 3083.4700520575875
On KNN-Model, R2 scores are:
Train: 0.9744307480591484
CV: 0.4945106548690854
Test: 0.5468835469030955
```

Finally, RMSE and R2 Scores on all models was plotted and following plot were obtained:



From the plot, we see that CV RMSE and R2 score are both higher for Random Forest, LightGBM and XGBoost Models. Further, for Logistic Regression and KNN, we can see that training evaluation metrics are much better than CV evaluation metrics, thus there is a clear overfit in these models on the optimal hyperparameters obtained from GridSearchCV. The Evaluation metrices on SVR and AdaBoost Regressor are also very low obtained as compared to other models. Thus, we can infer that the given dataset works better on Random-Forest, LightGBM and XGBoost Models, each of them giving an R2 score around 0.8, and RMSE around 1900.

Since the above three mentioned models (Random-Forest, LightGBM and XGBoost) work best on the dataset, an ensemble was created comprising of all 3 models. The Price Predicted by each model is calculated and finally, mean Price is stored as output. The evaluation metrices Test Data of the Ensemble thus formed is as follows. We can see that the metrices of the Ensemble is much better than the metrices obtained on individual models.

```
For the ensemble
R2 Score on Test-Data: 0.9132368751706568
RMSE on Test Data: 1348.8339943741935
```

Further, the R2 Score and RMSE on Test Data for each model along with ensemble is plotted as follows:

Thus, from the plot we can confirm that the ensemble of Random-Forest, LightGBM and XGBoost Regressors works the best. This can be because each model of ensemble is trained independently of other model, and since mean in considered, an error in price predicted by 1 model is compensated by the other 2 models.

Thus, from the above Models, we see that the average price predicted of following 3 models along the Hyperparameters predicted from GridSearchCV works the best on the given Dataset.

```
RandomForestRegressor(max_depth=25, min_samples_leaf=2, min_samples_split=10,
                      n_estimators=1000)
Best Score: 0.8112345921720578

XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.1, max_delta_step=0,
             max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=100, n_jobs=8,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
Best Score: 0.8172550137586679

                    LGBMRegressor(n_estimators=300)
                    Best Score: 0.8028686063030591
```

Finally, On the Test Data:

R2 Score = 0.9132368751706568

Root Mean Squared Error (RMSE) = 1348.8339943741935