FLIGHT FARE PREDICTION SYSTEM

A Project Submitted
In Partial Fulfillment of the Requirements
for the Degree of

BACHELOR OF TECHNOLOGY
IN
Computer Science & Engineering
by

DEEPAK JAISWAL (1701010051)
ABHISHEK KUMAR (1701010005)
AYUSH RAJ SINGH (1701010043)
ABHISHEK KUMAR MISHRA (1701010006)

Under the Supervision of
Mr. Sunil Khare Sir
Professor CS/IT Dept.

United College of Engineering And Research, Prayagraj

to the
Faculty of Computer Science & Engineering

Dr. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY
LUCKNOW
August, 2021

CERTIFICATE

Certified that **Deepak Jaiswal** (1701010051), **Abhishek Kumar** (1701010005), **Ayush Raj Singh** (1701010043), **Abhishek Kumar Mishra** (1701010006) has carried out the project work presented in this project entitled **"Flight Fare Prediction System"** for the award of **Bachelor of Technology (CSE)** from Dr. A.P. J. Abdul Kalam Technical University, Lucknow under our supervision. The project embodies results of original work, and studies are carried out by the student **himself** and the contents of the project do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

HOD CS/IT DEPARTMENT

Mr. Vijay Dwivedi Sir

PROJECT GUIDE

Mr. Sunil Khare Sir

DATE :

CANDIDATE'S DECLARATION

We, Deepak Jaiswal (1701010051), Abhishek Kumar (1701010005), Ayush Raj Singh (1701010043), Abhishek Kumar Mishra (1701010006), students of B.Tech of Computer Science & engineering hereby declare that we own the full responsibility for the information, results etc. provided in this PROJeCT titled **FLIGHT FARE PREDICTION SYSTEM** submitted to Dr. A.P.J Abdul Kalam University, Lucknow for the award of B.Tech (CSe) degree. I have taken care in all respect to honor the intellectual property right and have acknowledged the contribution of others for using them in academic purpose and further declare that in case of any violation of intellectual property right or copyright we, as a candidate, will be fully responsible for the same. My supervisor should not be held responsible for full or partial violation of copyright or intellectual property right.

Deepak Jaiswal (1701010051),
Abhishek Kumar (1701010005),
Ayush Raj Singh (1701010043)
Abhishek Kumar Mishra (1701010006)

DATE:

PLACE: Prayagraj (Allahabad) UP

ACKNOWLEDGEMENT

The completion of this project could not have been possible without the participation and assistance of a lot of individuals contributing to this project. However, we would like to express our deep appreciation and indebtedness to our teachers and supervisors for their endless support, kindness, and understanding during the project duration.

Also, we would like to thank all our relatives, family, and friends who supported us in one way or another.

Above all, we would like to thank the Great almighty for always having his blessing on us.

List of Tables

List of Figures

ABSTRACT

Optimal timing for airline ticket purchasing from the consumer's perspective is challenging principally because buyers have insufficient information for reasoning about future price movements. In this project we majorly targeted to uncover underlying trends of flight prices in India using historical data and also to suggest the best time to buy a flight ticket.

For this project, we have collected data from 18 routes across India while the data of 4 routes were extensively used for the analysis due to the sheer volume of data collected over 4 months resulting in 5.27 lakh data points each across the Mumbai-Delhi and Delhi-Mumbai route and 1.03 lakh data points each across the Delhi-Guwahati and Guwahati-Delhi route. The project implements the validations or contradictions towards myths regarding the airline industry, a comparison study among various models in predicting the optimal time to buy the flight ticket and the amount that can be saved if done so. a customized model which included a combination of ensemble and statistical models have been implemented with a best accuracy of above 90% for a few routes, mostly from Tier 2 to metro cities. These models have led to significant savings and produced average positive savings on each transaction.

Remarkably, the trends of the prices are highly sensitive to the route, month of departure, day of departure, time of departure, whether the day of departure is a holiday and airline carrier. Highly competitive routes like most business routes (tier 1 to tier 1 cities like Mumbai-Delhi) had a non-decreasing trend where prices increased as days to departure decreased, however other routes (tier 1 to tier 2 cities like Delhi - Guwahati) had a specific time frame where the prices are minimum. Moreover, the data also uncovered two basic categories of airline carriers operating in India – the economical group and the luxurious group, and in most cases, the minimum priced flight was a member of the economical group. The data also validated the fact that, there are certain time-periods of the day where the prices are expected to be maximum.

With a high probability (about 20-25%) that a person has to wait to buy a ticket, the scope of the project can be extensively extended across the various routes to make significant savings on the purchase of flight prices across the Indian Domestic airline market.

TABLE OF CONTENTS

CHAPTER 1 : INTRODUCTION

This project aims to develop an application which will predict the flight prices for various flights using machine learning model. The user will get the predicted values and with its reference the user can decide to book their tickets accordingly. In the current day scenario flight companies try to manipulate the flight ticket prices to maximize their profits. There are many people who travel regularly through flights and so they have an idea about the best time to book cheap tickets. But there are also many people who are inexperienced in booking tickets and end up falling in discount traps made by the companies where actually they end up spending more than they should have. The proposed system can help save millions of rupees of customers by proving them the information to book tickets at the right time. The proposed problem statement is “Flight Fare prediction system”.

Anyone who has booked a flight ticket knows how unexpectedly the prices vary. airlines use using sophisticated quasi-academic tactics known as "revenue management" or "yield management". The cheapest available ticket for a given date gets more or 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, if we could inform the travellers with the optimal time to buy their flight tickets based on the historic data and also show them various trends in the airline industry we could help them save money on their travels.This would be a practical implementation of a data analysis, statistics and machine learning techniques to solve a daily problem faced by travellers.

The objectives of the project can broadly be laid down by the following questions -

1. Flight Trends

   Do airfares change frequently? Do they move in small increments or in large jumps? Do they tend to go up or down over time?

2. Best Time To Buy

   What is the best time to buy so that the consumer can save the most by taking the least risk? So should a passenger wait to buy his ticket, or should he buy as early as possible?

3. Verifying Myths

   Does price increase as we get near to departure date? Is Indigo cheaper than Jet airways? Are morning flights expensive?

CHAPTER 2 : ANALYSIS

Automated Script to Collect Historical Data

For any prediction/classification problem, we need historical data to work with. In this project, past flight prices for each route needs to be collected on a daily basis. Manually collecting data daily is not efficient and thus a python script was run on a remote server which collected prices daily at specific time.

Cleaning & Preparing Data

After we have the data, we need to clean & prepare the data according to the model's requirements. In any machine learning problem, this is the step that is the most important and the most time consuming. We used various statistical techniques & logics and implemented them using built-in Python packages.

Analysing & Building Models

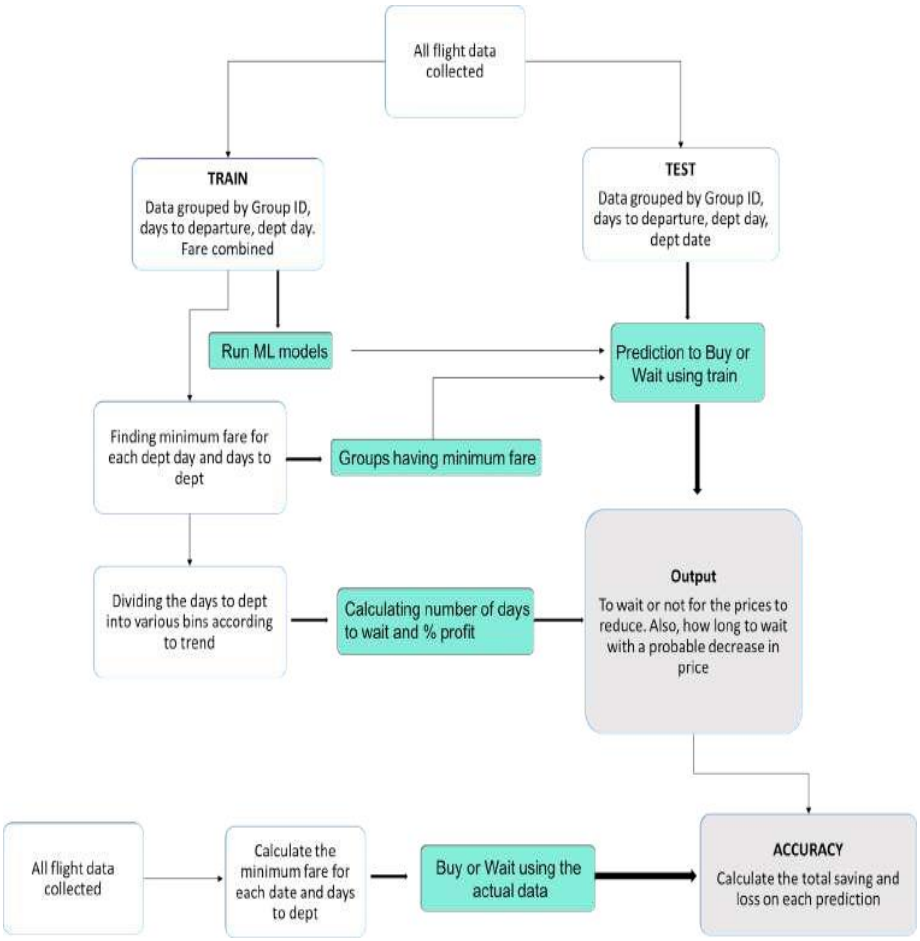
Data preparation is followed by analysing the data, uncovering hidden trends and then applying various predictive & classification models on the training set. These included Random Forest, Logistic Regression, Gradient Boosting and combination of these models to increase the accuracy. Further statistical models and trend analyzer model have been built to increase the accuracy of the ML algorithms for this task.

Merging Models & accuracy Calculation

Having built various models, we have to test the models on our testing set and calculate the savings or loss done on each query put by the user. a statistic of the over Savings, Loss and the mean saving per transaction are the measures used to calculate the accuracy of the model implemented.

Our Model

This section provides a bird-eye view on the whole model we used. The key components of the model are the **training data** and the **testing data**. The way we built these data and the various aspects of the model that uses these datasets is very tricky to understand. This flow chart will provide some basic understanding.



Data Collection

Since the APIs by Indian companies like Goibibo returned data in a complex format resulting in a lot of time to clean the data before analysing, therefore we decided to build a web spider that extracts the required values from a website and stores it as a CSV file. We decided to scrape travel service providers website using a manual spider made in Python. Further we also developed a Python script to run the aPI provided by Google flights which is more reliable, but it allows only 50 queries each day.

Such scrapping returns numerous variables for each flight returned and we had to decide the parameters that might be needed for the flight prediction algorithm. Not all are required and thus we selected the following -

1. Origin City
2. Destination City
3. Departure Date
4. Departure Time
5. Arrival Time
6. Total Fare
7. Airway Carrier
8. Duration
9. Class Type - economy/Business
10. Flight Number
11. Taken Date - date on which this data was collected

Data Cleaning

The data was further processed based on the parameters mentioned below and cleaned based on appropriate considerations -

1. Days to Departure
2. Day of Departure
3. Duration
4. Holiday
5. Outliers

Further, the data was analysed and tests on the distribution were performed. Conclusions of the tests revealed that our data followed Log-Normal distribution and the same has been positively confirmed through statistical methods.

Based on previous history, the trend in the flight prices were modelled and the same was used to provide the user with an approximation of the number of days to wait from the current day, and if at all he waits, the amount he can say on the ticket.

In order to predict if the customer has to wait or not, we used a combination of statistical models and machine learning models. The statistical model provided with a probability corresponding to each airline having the least cost while the machine learning model further went ahead to predict the specific conditions taking into account the days to departure and the day of departure.

The machine learning algorithms implemented started off with basic Regression models and were extended to Decision Trees followed by Random Forests and Gradient Boosting methods. Later we developed an algorithm which had a combination of Rule based learning, ensemble models and Statistical models to increase the accuracy.

Based on the prediction made by the model and the estimated time to wait, we calculated the savings we could achieve and the losses we incurred based on the predictions.

Data Preparation

Data preparation was a critical part, as we had multiple airlines on a specific day and we had to predict the future prices for all those airlines, or the airline which would have the lowest fare.

Suppose a user makes a query to buy a flight ticket 44 days in advance, then our system should be able to tell the user whether he should wait for the prices to decrease or he should buy the tickets immediately. For this we have two options:

1. Predict the flight prices for all the days between 44 and 1 and check on which day the price is minimum.

2. Classify the data we already have into, "Buy" or "Wait". This then becomes a classification problem and we would need to predict only a binary number. However, this does not give a good insight on the number of days to wait.

For the above example, if we choose the first method we would need to make a total of 44 predictions (i.e. run a machine learning algorithm 44 times) for a single query. This also cascades the error per prediction decreasing the accuracy. Hence, the second method seems to be a better way to predict, wait or buy which is a simple binary classification problem. But, in this method, we would need to predict the days to wait using the historic trends.

For this we again have two options:

1. We do the predictions for each flight id. The problem with this is that, if there is a change in flight id by the airline (which happens frequently) or there is an introduction or a new flight for a specific route then our analysis would fail.

2. We group the flight ids according to the airline and the time of departure and do the analysis on each group. For this we need to combine the prices of the airlines lying in that group such that the basic trend in captured.

Moving ahead with the second option, we created the group according to the airlines and the departure time-slot created earlier (Morning, evening, Night) and calculated the combined flight prices for each group, day of departure and depart day. Since these three are the most influencing factors which determine the flight prices. also, we calculated the average number of flights that operated in a particular group, since competition could also play a role in determining the fare.

	Airline	Source	Destination	Total_Stops	Price	Journey_day	Journey_month	Dep_hour	Dep_min	Arrival_hour	Arrival_min	Duration_hours	Duration_m
0	IndiGo	Banglore	New Delhi	0	3897	24	3	22	20	1	10	2	
1	Air India	Kolkata	Banglore	2	7662	1	5	5	50	13	15	7	
2	Jet Airways	Delhi	Cochin	2	13882	9	6	9	25	4	25	19	
3	IndiGo	Kolkata	Banglore	1	6218	12	5	18	5	23	30	5	
4	IndiGo	Banglore	New Delhi	1	13302	1	3	16	50	21	35	4	

Combining fare for the flights in one group:

1. Mean fare: This is the average of the fare of all the flights in a particular group corresponding to departure day and days to departure. Because of high standard deviation, taking the mean is not a very good option.

2. Minimum fare: This does not give a very good insight of the trend, as a minimum value could occur because of some offer by an airline.

3. First Quartile: This is a good measure as we are focusing on minimizing the fare and we do not want to consider the flights with high fares.

4. Custom Fare: This is the fare giving more weightage to recent price trend.
Total_customFare = w*(First Quartile for entire time period) + (1-w)*(First quartile of last x days)

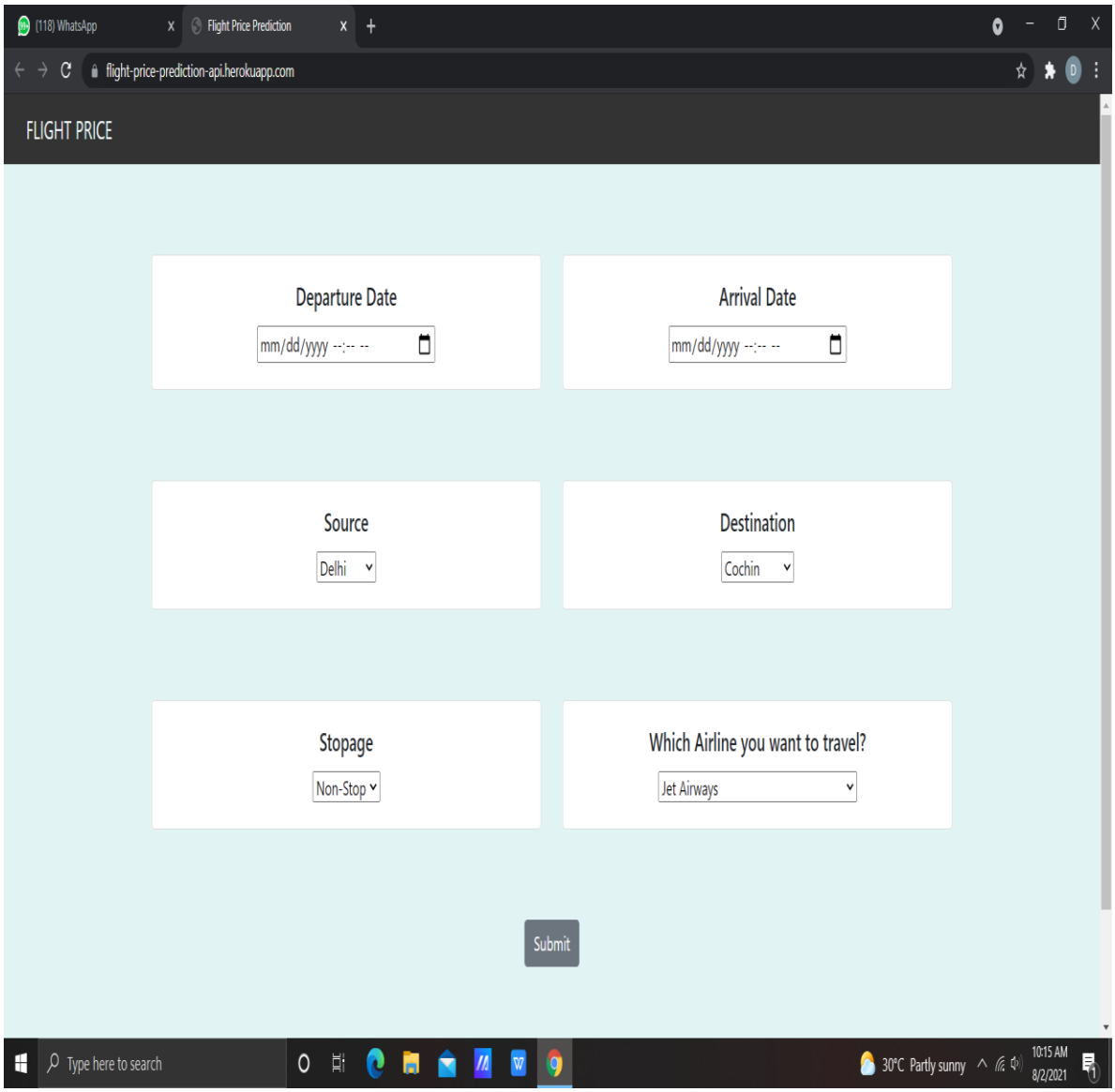
Calculating whether to buy or wait for the this data:

Logical = 1 if for any d < D the Total_customFare is less than the current Total_customFare

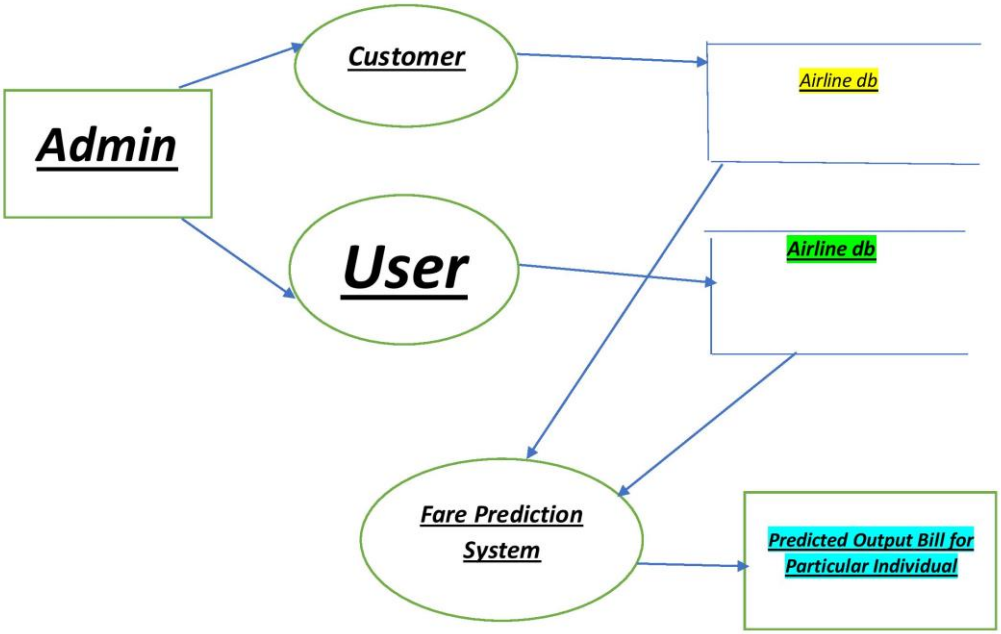
(Here, d is the days to departure and D is the days to departure for the current row.)

CHAPTER 3: DESIGN

This is the home page of our web application.



Level-0 DFD

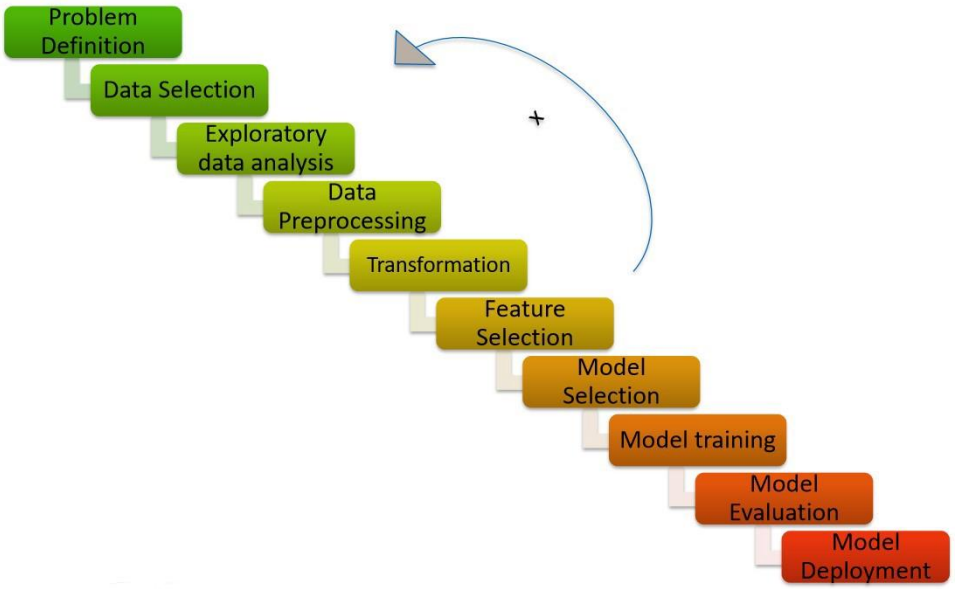

Admin
Customer
User
Airline db
Airline db
Fare Prediction System
Predicted Output Bill for Particular Individual

Level-1 DFD

CHAPTER 4 : CODING / IMPLEMENTATION

IMPLEMENTATION :

For this project, we have implemented the machine learning life cycle to create a basic web application which will predict the flight prices by applying machine learning algorithm to historical flight data using python libraries like Pandas, NumPy, Matplotlib, seaborn and sklearn. Below figure shows the steps that we followed from the life cycle:



Data selection is the first step where historical data of flight is gathered for the model to predict prices. our dataset consists of more than 10,000 records of data related to flights and its prices. Some of the features of the dataset are source, destination, departure date, departure time, number of stops, arrival time, prices and few more. In the exploratory data analysis step, we cleaned the dataset by removing the duplicate values and null values. If these values are not removed it would affect the accuracy of the model. We gained further information such as distribution of data.

Next step is data pre-processing where we observed that most of the data was present in string format. Data from each feature is extracted such as day and month is extracted from date of journey in integer format, hours and minutes is extracted from departure time. Features such as source and destination needed to be converted into values as they were of categorical type. For this one hot-encoding and label encoding techniques are used to convert categorical values to model identifiable values.

Feature selection step is involved in selecting important features that are more correlated to the price. There are some be selected and passed to the group of models.

Random forest basically uses group of decision trees as group of models. Random amount of data is passed to decision trees and each decision tree predicts values according to the dataset given to it. From the predictions made by the decision trees the features such as extra information and route which are unnecessary features which may affect the accuracy of the model and therefore, they need to be removed before getting our model ready for prediction. after selecting the features which are more correlated to price the next step involves applying machine algorithm and creating a model. as our dataset consist of labelled data, we will be using supervised machine learning algorithms also in supervised we will be using regression algorithms as our dataset contains continuous values in the features. Regression models are used to describe relationship between dependent and independent variables. The machine learning algorithms that we will be using in our project are:

Linear Regression

In simple linear regression there is only one independent and dependent feature but as our dataset consists of many independent features on which the price may depend upon, we will be using multiple linear regression which estimates relationship between two or more independent variables and one dependent variable.

The multiple linear regression model is represented by:

$$Y = \beta 0x1 + \ldots. + \beta nxn + Ɛ$$

Y = the predicted value of the dependent variable
Xn = the independent variables
βn = independent variables coefficients
Ɛ = y-intercept when all other parameters are 0

Decision Tree

Decision trees are basically of two types classification and regression tree where classification is used for categorical values and regression is used for continuous values. Decision tree chooses independent variable from dataset as decision nodes for decision making.

It divides the whole dataset in different sub-section and when test data is passed to the model the output is decided by checking the section to which the datapoint belong to. and to whichever section the data point belongs to the decision tree will give output as the average value of all the datapoints in the sub-section Random Forest

Random Forest

We have implemented Random Forest algorithm in our project.
Random Forest is an ensemble learning technique where training model uses multiple learning algorithms and then combine individual results to get a final predicted result. Under ensemble learning random forest falls into bagging category where random number of features and records will average value of the predicted values if considered as the output of the random forest model.

Performance Metrics

Performance metrics are statistical models which will be used to compare the accuracy of the machine learning models trained by different algorithms. The sklearn.metrics module will be used to implement the functions to measure the errors from each model using the regression metrics. Following metrics will be used to check the error measure of each model.

MAE (Mean absolute error)

Mean absolute error is basically the sum of average of the absolute difference between the predicted and actual values.

$$MAE = 1/n[\sum(y-\acute{y})]$$

y = actual output values,
ý = predicted output values
n = Total number of data points

Lesser the value of MAE the better the performance of your model.

MSE (Mean Square error)

Mean Square error squares the difference of actual and predicted output values before summing them all instead of using the absolute value.

$$MSE = 1/n[\sum(y-\acute{y})^2]$$

y=actual output values
ý=predicted output values
n = Total number of data points

MSE punishes big errors as we are squaring the errors. Lower the value of MSE the better the performance of the model.

RMSE (Root Mean Square error)

RMSE is measuredby taking the square root of the average of the squared difference between the prediction and the actual value.

$$RMSE = \sqrt{1/n[\sum(y-\acute{y})^2]}$$

y=actual output values
ý=predicted output values
n = Total number of data points

RMSE is greater than MAE and lesser the value of RMSE between different model the better the performance of that model.

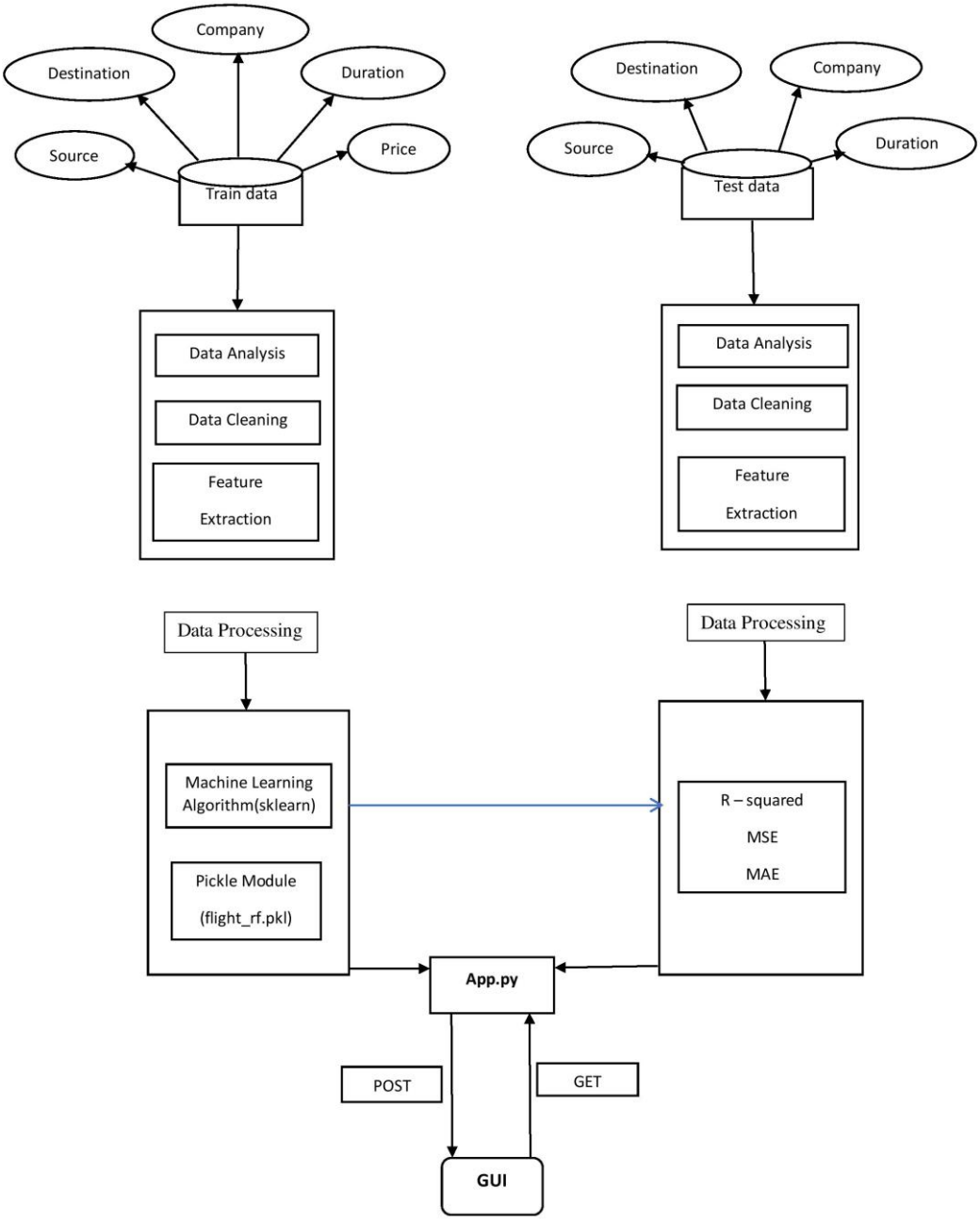
R^2 (Coefficient of determination)

It helps you to understand how well the independent variable adjusted with the variance in your model.

$$R^2 = 1- (\Sigma(\acute{y}-\hat{y}\textbf{Error! No bookmark name given.})^2 / \Sigma(y-\hat{y})^2)$$

The value of R-square lies between 0 to 1. The closer its value to one, the better your model is when comparing with other model values.

System Architecture Diagram



There are also different cross-validation techniques such as gridsearchCV and randomizedsearchCV which will be used for improving the accuracy of the model. Parameters of the models such as number of trees in random forest or max depth of decision tree can be changed using this technique which will help us in further enhancement of the accuracy.

The last three steps of the life cycle model are involved in the deployment of the trained machine learning model. Therefore, after getting the model with the best accuracy we store that model in a file using pickle module. The back-end of the application will be created using Flask Framework where aPI end-points such and GeT and PoST will be created to perform operations related to fetching and displaying data on the front-end of the application.

The front-end of the application will be created using the bootstrap framework where user will have the functionality of entering their flight data. This data will be sent to the back-end service where the model will predict the output according to the provided data. The predicted value is sent to the front-end and displayed

Fitting model using Random Forest

1. Split dataset into train and test set in order to prediction w.r.t X_test
2. If needed do scaling of data
    - Scaling is not done in Random forest
3. Import model
4. Fit the data
5. Predict w.r.t X_test
6. In regression check **RSME** Score
7. Plot graph

CODING:

jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [66]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
```

In [67]:
```
from sklearn.ensemble import RandomForestRegressor
reg_rf = RandomForestRegressor()
reg_rf.fit(X_train, y_train)
```

Out[67]: RandomForestRegressor()

In [68]:
```
y_pred = reg_rf.predict(X_test)
```

In [69]:
```
reg_rf.score(X_train, y_train)
```

Out[69]: 0.952745356858999

In [70]:
```
reg_rf.score(X_test, y_test)
```

Out[70]: 0.7960884857767897

In [71]:
```
sns.distplot(y_test-y_pred)
plt.show()
```

```
C:\Users\win10\Anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

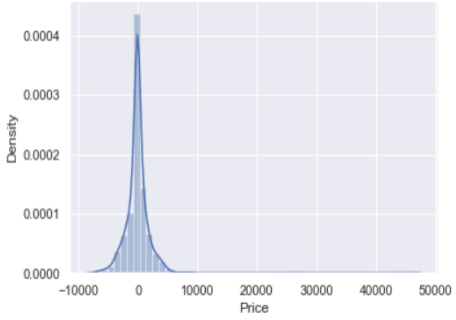
Out[69]: 0.952745356858999

In [70]:
```
reg_rf.score(X_test, y_test)
```

Out[70]: 0.7960884857767897

In [71]:
```
sns.distplot(y_test-y_pred)
plt.show()
```

```
C:\Users\win10\Anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [72]:
```
# Graph is forming a gaussian distribution which is pretty much good and it basically means our results are very good.
```

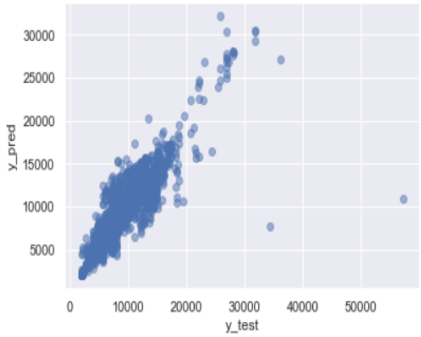
jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [73]:
```
plt.scatter(y_test, y_pred, alpha = 0.5)
plt.xlabel("y_test")
plt.ylabel("y_pred")
plt.show()
```



In [74]:
```
from sklearn import metrics
```

In [75]:
```
print('MAE:', metrics.mean_absolute_error(y_test, y_pred))
print('MSE:', metrics.mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
MAE: 1178.6420429976617
MSE: 4396751.5548259765
RMSE: 2096.8432356344565
```

jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

Hyperparameter Tuning

- Choose following method for hyperparameter tuning
  1. RandomizedSearchCV --> Fast
  2. GridSearchCV
- Assign hyperparameters in form of dictionery
- Fit the model
- Check best paramters and best score

In [77]:
```
from sklearn.model_selection import RandomizedSearchCV
```

In [78]:
```
#Randomized Search CV

# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start=100,stop=1200, num=12)]
# Number of features to consider at every split
max_features = ['auto', 'sqrt']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(5, 30, num = 6)]
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10, 15, 100]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 5, 10]
```

jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [79]:
```
# Create the random grid

random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf}
```

In [80]:
```
# Random search of parameters, using 5 fold cross validation,
# search across 100 different combinations
rf_random = RandomizedSearchCV(estimator = reg_rf, param_distributions = random_grid,scoring='neg_mean_squared_error', n_iter = 1
```

In [81]:
```
rf_random.fit(X_train,y_train)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  10.3s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   10.2s remaining:    0.0s
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  11.1s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  12.0s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  10.8s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  10.4s
[CV] n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15
[CV]  n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15, total=  15.3s
```

jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [81]:
```
rf_random.fit(X_train,y_train)
```


```
[CV]  n_estimators=700, min_samples_split=15, min_samples_leaf=1, max_features=auto, max_depth=20, total=  33.7s
[CV] n_estimators=700, min_samples_split=15, min_samples_leaf=1, max_features=auto, max_depth=20
[CV]  n_estimators=700, min_samples_split=15, min_samples_leaf=1, max_features=auto, max_depth=20, total=  31.8s
[CV] n_estimators=700, min_samples_split=15, min_samples_leaf=1, max_features=auto, max_depth=20
[CV]  n_estimators=700, min_samples_split=15, min_samples_leaf=1, max_features=auto, max_depth=20, total=  33.3s
[Parallel(n_jobs=1)]: Done  50 out of  50 | elapsed: 12.9min finished
```

Out[81]:
```
RandomizedSearchCV(cv=5, estimator=RandomForestRegressor(), n_jobs=1,
                   param_distributions={'max_depth': [5, 10, 15, 20, 25, 30],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [1, 2, 5, 10],
                                        'min_samples_split': [2, 5, 10, 15,
                                                              100],
                                        'n_estimators': [100, 200, 300, 400,
                                                         500, 600, 700, 800,
                                                         900, 1000, 1100,
                                                         1200]},
                   random_state=42, scoring='neg_mean_squared_error',
                   verbose=2)
```

In [82]:
```
rf_random.best_params_
```

Out[82]:
```
{'n_estimators': 700,
 'min_samples_split': 15,
 'min_samples_leaf': 1,
 'max_features': 'auto',
 'max_depth': 20}
```

In [83]:
```
prediction = rf_random.predict(X_test)
```

jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

Logout

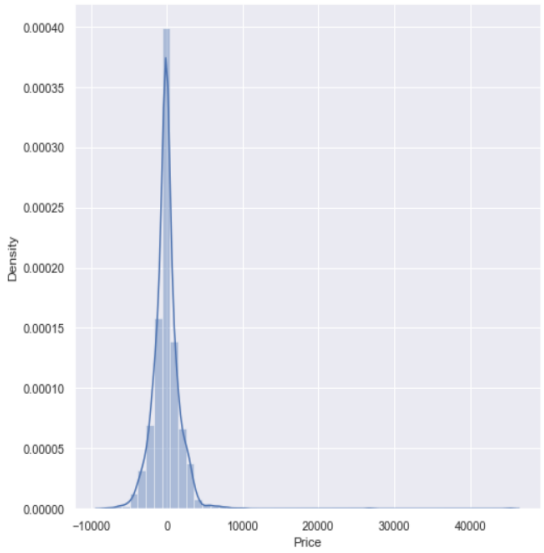
File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [84]:
```
plt.figure(figsize = (8,8))
sns.distplot(y_test-prediction)
plt.show()
```

```
C:\Users\win10\Anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



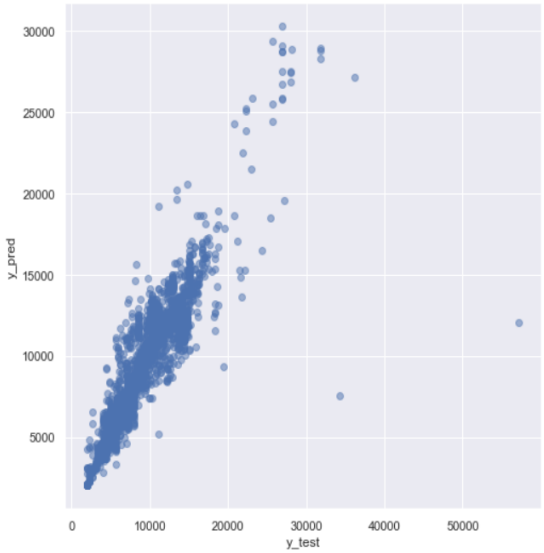
jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [85]:
```
plt.figure(figsize = (8,8))
plt.scatter(y_test, prediction, alpha = 0.5)
plt.xlabel("y_test")
plt.ylabel("y_pred")
plt.show()
```



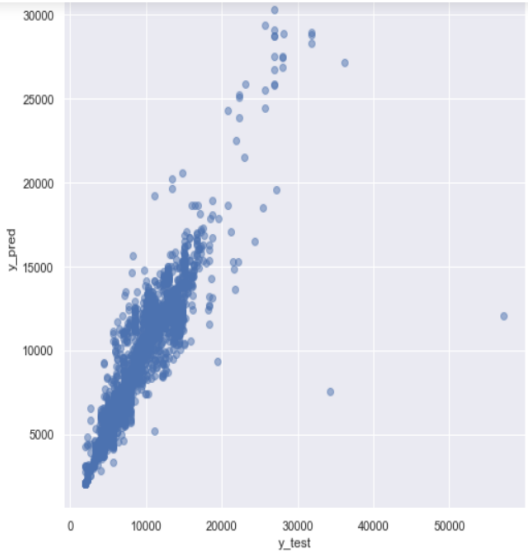
jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code



In [86]:
```
print('MAE:', metrics.mean_absolute_error(y_test, prediction))
print('MSE:', metrics.mean_squared_error(y_test, prediction))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, prediction)))
```

```
MAE: 1165.4642287168228
MSE: 4051871.204272184
RMSE: 2012.926030501912
```

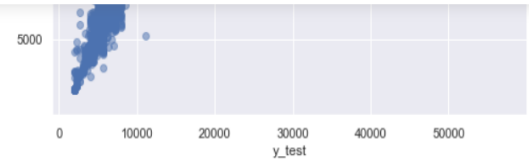
jupyter Flight Fare Prediction System Last Checkpoint: 27/06/2021 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code



In [86]:
```
print('MAE:', metrics.mean_absolute_error(y_test, prediction))
print('MSE:', metrics.mean_squared_error(y_test, prediction))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, prediction)))
```

```
MAE: 1165.4642287168228
MSE: 4051871.204272184
RMSE: 2012.926030501912
```

Save the model to reuse it again

In [87]:
```
import pickle
# open a file, where you ant to store the data
file = open('flight_rf.pkl', 'wb')

# dump information to that file
pickle.dump(reg_rf, file)
```

In [88]:
```
model = open('flight_rf.pkl','rb')
forest = pickle.load(model)
```

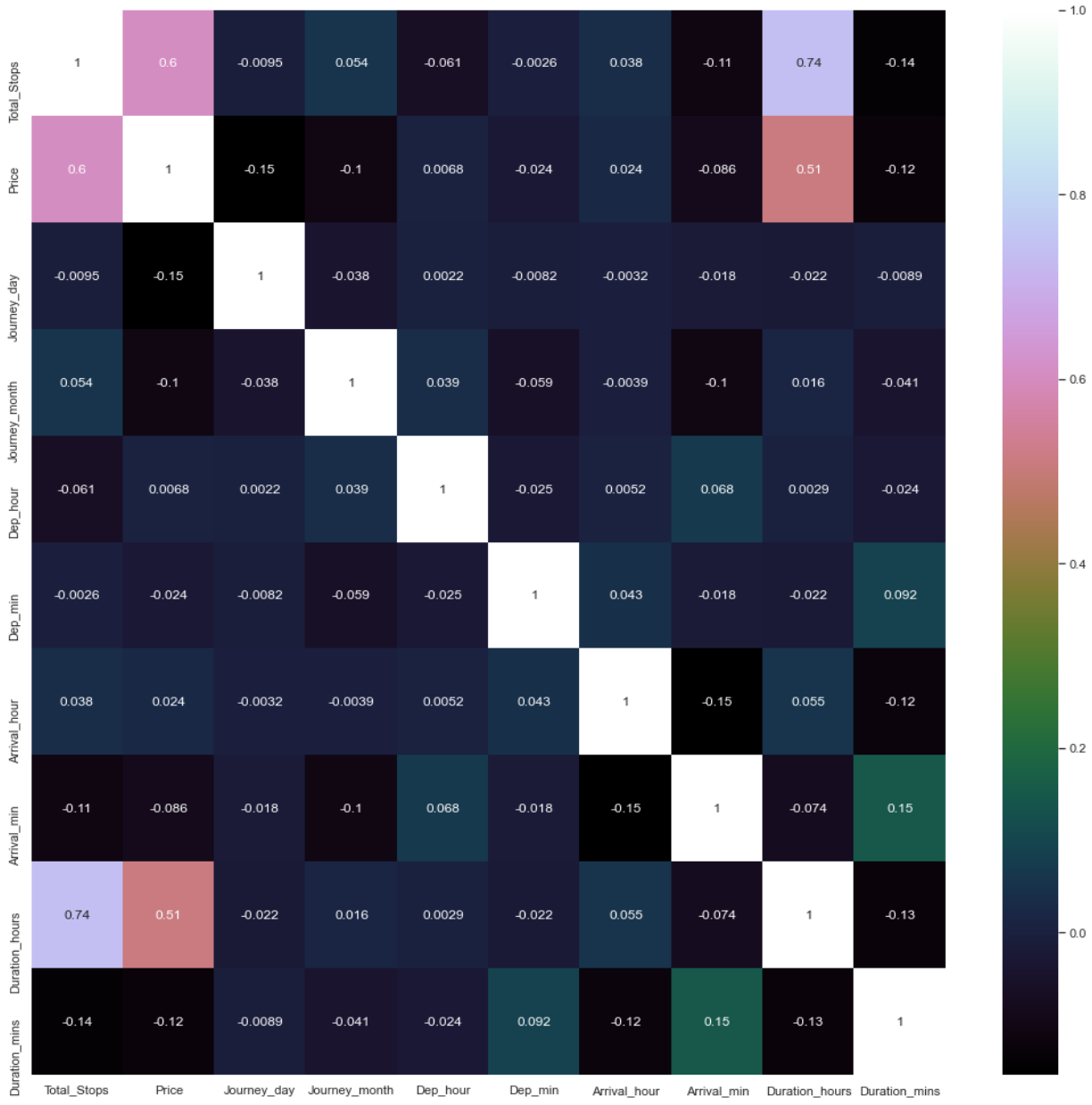
In [89]:
```
y_prediction = forest.predict(X_test)
```

In [90]:
```
metrics.r2_score(y_test, y_prediction)
```

Out[90]: 0.7960884857767897

Correlation between Independent and dependent attributes

```
plt.figure(figsize = (18,18))
sns.heatmap(train_data.corr(), annot = True, cmap = "cubehelix")
plt.show()
```

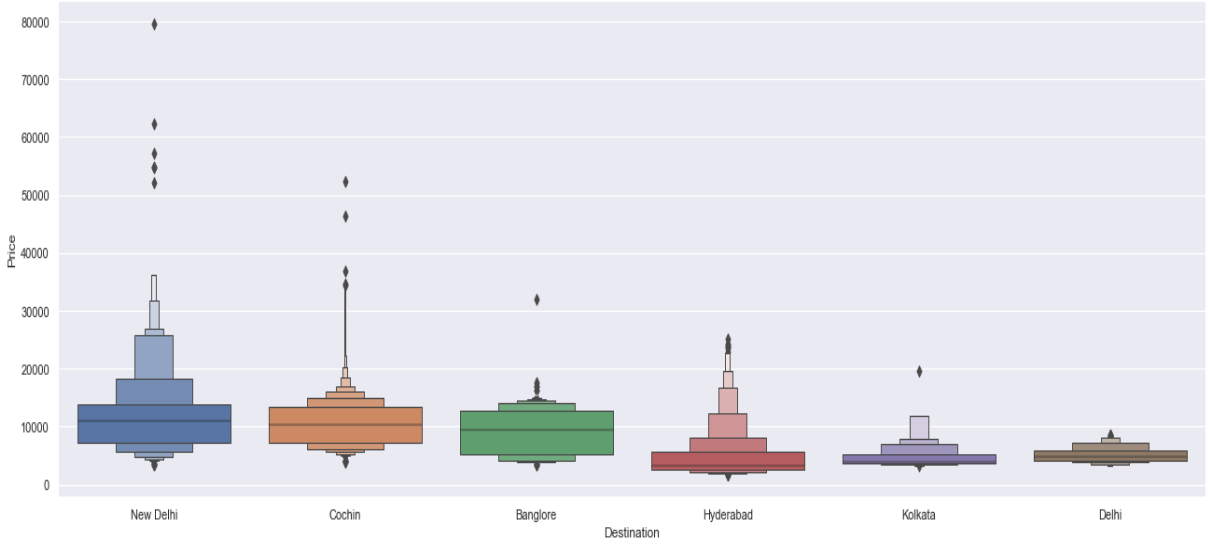


Results

In detailed analysis for the Delhi - Cochin Route

The trends in the data collected for the sector of Delhi to Cochin busted some of the very famous myths assumed by travellers of the aviation industry.

Flight prices do not increase continuously as the Date of Departure approaches closer.



With the validation of the problem statement and with a scope to predict when to buy and when to wait, we begin the analysis of the dataset.

The dataset of the flight prices follows a Lognormal distribution with some outliers which have been ignored as we are only interested with the minimum fare corresponding to a certain route.

Conclusion Remarks from exploratory Data analysis

From the data collected and through exploratory data analysis, we can determine the following:

- The trend of flight prices vary over various months and across the holiday.
- There are two groups of airlines: the economical group and the luxurious group. Spicejet, airasia, IndiGo, Go air are in the economical class, whereas Jet airways and air India in the other. Vistara has a more spread out trend.
- The airfare varies depending on the time of departure, making timeslot used in analysis is an important parameter.
- The airfare increases during a holiday season. In our time period, during Diwali the fare remained high for all the values of days to departure. We

have considered holiday season as a parameter which helped in increasing the accuracy.

- Airfare varies according to the day of the week of travel. It is higher for weekends and Monday and slightly lower for the other days.

- There are a few times when an offer is run by an airline because of which the prices drop suddenly. These are difficult to incorporate in our mathematical models, and hence lead to error.

- Along the business routes, we find that the price of flights increases or remains constant as the days to departure decreases. This is because of the high frequency of the flights, high demand and also could be due to heavy competition.

- Only about 8-10% of the times, a person should wait according to the data collected across the Mumbai-Delhi route, compared to 30-40% in Delhi-Cochin route.

Conclusion

From our detailed analysis of each of the routes, we can determine the following

- Flight prices almost always remain constant or increase between the major cities

- Tourist routes and routes that offer services involving Tier-2 cities of the country have uneven trends related to the increase and decrease of airline ticket prices.

- The model in the worst case almost breaks even with the profits and losses, and most case saves an average of about Rs. 200 per transaction when predicting to wait.

- Routes with data collected over the longer duration of time tend to facilitate with much more accurate predictions in the model and thus lead to higher average savings.

We were successfully able to analyse each route and generalize the entire project based in terms of the sector to which the route belonged, and classified them into three major subsections - Business Routes, Tourist Routes and Tier-2 Routes.
We have also successfully busted some of the typical myths and misconceptions related to the airline industry and backed them up with data and analysis.

Finally, we have created a User Interface for the entire process of buying an airline ticket and given a proof of our predictions based on the previous trends with our prediction. Thus leaving it as a battle between '**The risk appetite of the user**' vs ' **our understanding of the airline industry'.**

Future Work

● More routes can be added and the same analysis can be expanded to major airports and travel routes in India.

● The analysis can be done by increasing the data points and increasing the historical data used. That will train the model better giving better accuracies and more savings.

● More rules can be added in the Rule based learning based on our understanding of the industry, also incorporating the offer periods given by the airlines.

● Developing a more user friendly interface for various routes giving more flexibility to the users.

REFRENCES

1. Manolis Papadakis. Predicting airfare Prices.

2. Groves and Gini, 2011. a Regression Model For Predicting optimal Purchase Timing For airline Tickets.

3. Modeling of United States airline Fares – Using the official airline Guide (oaG) and airline origin and Destination Survey (DB1B), Krishna Rama-Murthy, 2006.

4. Course on Machine Learning by Krish Naik.

5. B. S. everitt: The Cambridge Dictionary of Statistics, Cambridge University Press, Cambridge (3rd edition, 2006). ISBN 0-521-69027-7

PLAGIARISM REPORT

Plagiarism Detector v. 1900 - Originality Report 04-08-2021 17:53:40

Analyzed document: FLIGHT FARE PREDICTION SYSTEM.doc Licensed to: Originality report generated by unregistered Demo version!

Comparison Preset: Rewrite Detected language: En

Check type: Internet Check

Warning: Demo Version - reports are incomplete!

Detect **more Plagiarism** with **Licensed Plagiarism Detector**:

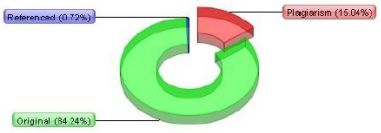
Order your **Lifetime License** packed with features:

1. **Complete** resources processing - with **more results**!
2. **Side-by-side compare** with detailed analysis!
3. **Faster** processing **speed**, **deeper detection!**
4. **Advanced statistics**, Originality Reports management!
5. Many other **cool functions** and **options!**

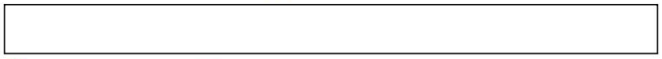
Get your 5% discount:

Detailed document body analysis:

Relation chart:



Distribution graph:



Top sources of plagiarism: 16

12%	666	1. URL will be available only with a License! Order a License
6%	350	2. URL will be available only with a License! Order a License
6%	286	3. URL will be available only with a License! Order a License

Processed resources details: 115 - Ok / 10 - Failed

Important notes:

Wikipedia:	Google Books:	Ghostwriting services:	Anti-cheating:
WIKIPEDIA	Google Book Search		

[not detected]	[not detected]	[not detected]	[not detected]

Active References (Urls Extracted from the Document):

No URLs detected

Excluded Urls:

No URLs detected

Included Urls:

No URLs detected

Detailed document analysis:

FLIGHTTFARETPREDICTIONTSYSTEM

ATProjectTSubmitted

InTPartialTFulfillmentTofTtheTRequirements

forTtheTDegreeTof

BACHELORTOFTTECHNOLOGY

IN

ComputerTScienceT&TEngineering

Tby

DEEPAKTJAISWALT(1701010051)

ABHISHEKTKUMART(1701010005)

AYUSHTRAJTSINGHT(1701010043)

ABHISHEKTKUMARTMISHRAT(1701010006)

UnderTtheTSupervisionTof

Mr.TSunilTKhareTSir

ProfessorTCS/ITTDept.

UnitedTCollegeTofTEngineeringTAndTResearch,TPrayagraj

toTthe

FacultyTofTComputerTScienceT&Tengineering

Dr.TA.P.J.TABDULTKALAMTTECHNICALTUNIVERSITY

LUCKNOW

August,T2021

CERTIFICATE

CertifiedTthatTDeepakTJaiswalT(1701010051)
ThasTcarriedToutTtheTprojectTworkTpresentedTinTthisTprojectTentitledT

Referenced: **0.08%** in: id: **1**

"FlightTFareTPredictionTSystem"

TforTtheTawardTofTBachelorTofTTechnologyT(CSE)
TfromTDr.TA.P.TJ.TAbdulTKalamTTechnicalTUniversity,TLucknowTunderTourTsupervision.TTheTprojectTembodiesTresultsTofTorigi
HODTCS/ITTDEPARTMENTTPROJECTTGUIDE T Mr.TVijayTDwivediTSirTMr.TSunilTKhareTSirT DATET:T
CANDIDATE'STDECLARATION We,TDeepakTJaiswalT(1701010051),TAbhishekTKumarT