Detail Project Report (DPR)

FLIGHT FARE PREDICTION

Revision Number: 1.0

Last Date of Revision: 02/06/2023

Document Version Control

Date Issued	Version	Description	Author
2/06/2023	1.0	Detail project report	Debasish Bhagawati

Detail Project Report (DPR)

1. Objective

Development of a predictive model for flight fare. The model will predict the price of a flight fare based on the user input.

2. DESCRIPTION

2.1 Problem Perspective

Since the privatisation of the airline firm, the pricing of airfares has developed into a sophisticated system of rules, including computer simulations that determine airfare marketing methods. Research has shown that these principles are affected by a variety of events, despite the fact that they are still mostly unknown. While still crucial, traditional factors like proximity no longer determine pricing structure exclusively. Increasingly, sociological, economic, and marketing considerations all have an impact on how much flights cost. The flight fare prediction is a machine learning model that aids in predicting travel costs and aids users in comprehending the cost of their journey.

2.2 Problem Statement

Travelling through flights has become an integral part of today’s lifestyle as more and more people are opting for faster travelling options. The flight ticket prices increase or decrease every now and then depending on various factors like timing of the flights, destination, and duration of flights various occasions such as vacations or festive season. Therefore, having some basic idea of the flight fares before planning the trip will surely help many people save money and time.

2.3 Purposed Solution

The main goal is to predict the fares of the flights based on different factors available in the provided dataset.

2.4 Further Improvements

We can even use the features like weekdays or weekend, season, rating for predicting the price.

2.5 Data Requirement

To build the airline ticket pricing model, we first need information on aircraft business. We now have two datasets, one for training and one for testing. We can access the dataset on kaggle within the type of standout sheet(.xlsx).

2.6 Tool Used

- Jupyter Notebook is used for data pre-processing, model building etc.
- PyCharm is used as IDE.
- Streamlit is used for backend development.
- AWS is used for deployment of the model.
- Github is used as version control system.

2.7. Data Collection

The data is collected from the kaggle website. link : Dataset

2.8. Data Description

We have two dataset one for training and other one is for testing. Training dataset consist of 10683 rows and 11 columns. 11 columns are –

- **airline** – Airline companies name.
- **Date_of_Journey** – This column will inform us of the date the passenger's travel will begin.

- **Source -** This column indicates the names of the location from which the passenger's journey will begin.
- **Destination** – Destination city of the passenger.
- **Route -** names of the location from where the customer's journey would begin.
- **Dep_Time -** The time when the plane leaves the gate of the airport (departure time)
- **Arrival_Time -** the time the airplane arrives at its gate
- **Duration -** The flight's endurance in hours..
- **Total_Stops -** The total number of breaks in the voyage.
- **Additional_Info -** It will indicate whether a meal is included with the journey or not.
- **Price –** Fare of that journey.

The test dataset consist of 2671 rows with 10 columns exactly same as the training data excluding the Price column.

3 Data Pre-processing and Feature Engineering

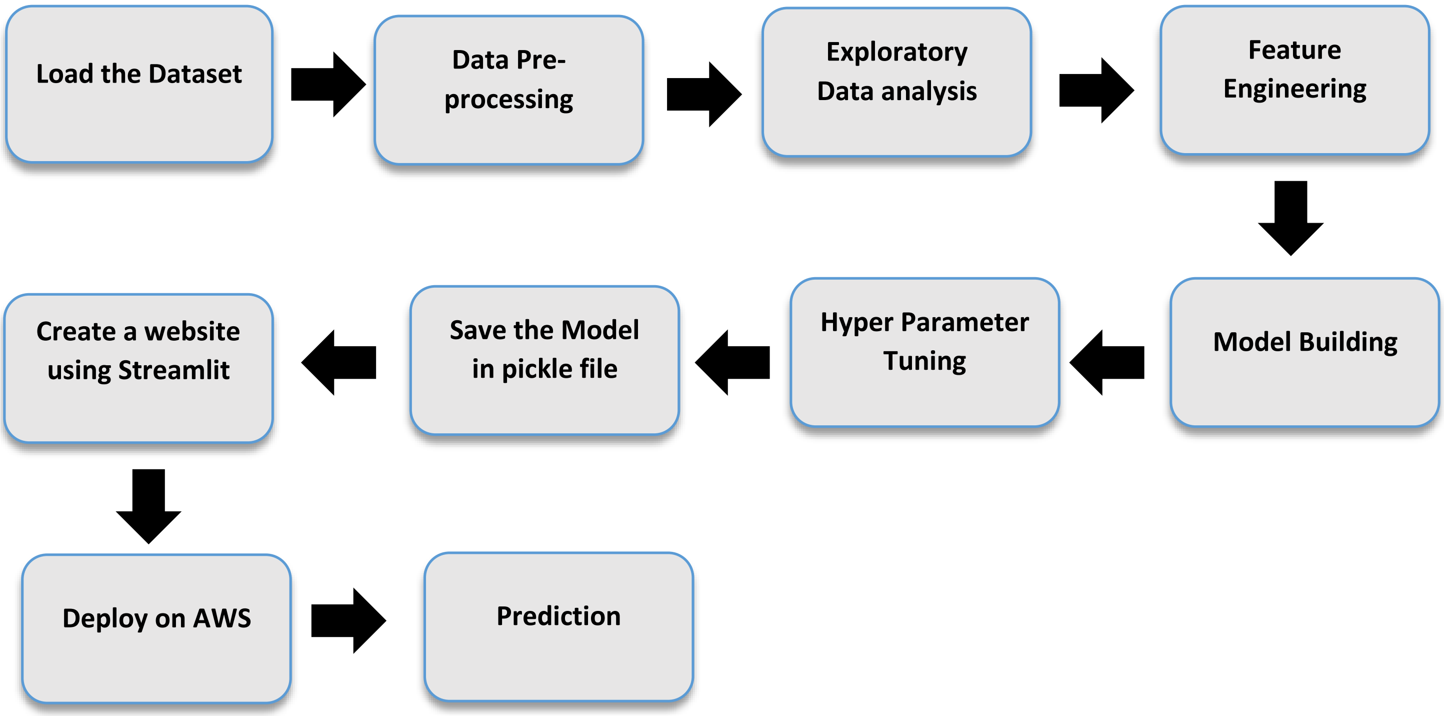
steps :

- Univariate Analysis of each columns.
- Handling the missing value.
- Convert all the desired column into the date-time format.
- Handling the categorical columns.

4.Model Building

We then split the training dataset into two parts using sklearn's train_test_split function and build different regression model using the training data and test it on the test data. Whichever model gives higher accuracy we will do hyperparameter tuning to enhance the performance of the model. In our case, Random forest is the model which gives the better accuracy.

5. Architecture



6. Data from User

Here we collect user data from an HTML web page that has been created.

7. Data Validation

The data provided by the user is then being processed by app.py file and validated. The validated data is then sent for the prediction.

8. Prediction

The data sent for the prediction is then rendered to the web page.

9. Deployment

We will be deploying the model to AWS so that anyone can access.

10. Conclusion

The flight fare prediction uses training information for forecast the value. As a result, the user will be able to estimate the cost of their travel.

6. Q & A

Q1) What’s the source of data?

The data for training is provided by the client.

Q 2) What was the type of data?

The data was the combination of numerical and Categorical values.

Q 3) What’s the complete flow you followed in this Project?

Refer Page no 5 for better Understanding.

Q 6) What techniques were you using for data pre-processing?

- Removing unwanted attributes.
- Visualizing relation of independent variables with each other and output variables.
- Cleaning data and imputing data if there are missing values.
- Converting categorical data into numerical data

- Converting date columns into date-time format.

Q 7) How training was done or what models were used?

- Before dividing the data in training and validation set, we performed pre-processing over the data set and made the final dataset.

- Algorithms like Linear regression, Decision Tree, Random Forest, XGBoost, Gradient Boost, Ada Boost, SVM were used, final model was used on the dataset and we saved that model.

Q 8) How Prediction was done?

The testing files are shared by the client. We Performed the same life cycle on the provided dataset. Then, on the basis of dataset, model is loaded and prediction is performed. In the end we get the accumulated data of predictions.

Q 9) What are the different stages of deployment?

- At first, the files are stored on GitHub.
- The model is first tested in the local environment.
- After successful testing, it is deployed on AWS.