Low Level Design (LLD)

FLIGHT FARE PREDICTION

Revision Number – 1.0

Last Date of Revision – 29/04/2023

Sagar Durgude

Document Version Control

Date	Version	Description	Author
27-04-2023	1.0	Abstract, Introduction Architecture	Sagar
28-04-2023	1.1	Data Preprocessing	Sagar
29-04-2023	1.2	Deployment	Sagar

Contents

Abstract 4

INTRODUCTION 5

Why this LLD documentation? 5

1 Architecture 3

2 Architecture Design 6

2.1 Data Gathering from Main Source 6

2.2 Tools Used 6

2.3 Data Description 7

2.4 Import Data into Database 7

2.5 Export Data from Database 8

2.6 Data Pre-Processing 8

2.7 Modelling 8

2.8 UI Integration 8

2.9 Data From User 8

2.10 Data Validation 8

2.11 Rendering the Results 8

3 Deployment 8

3.1 Unit Test Cases 9

Abstract

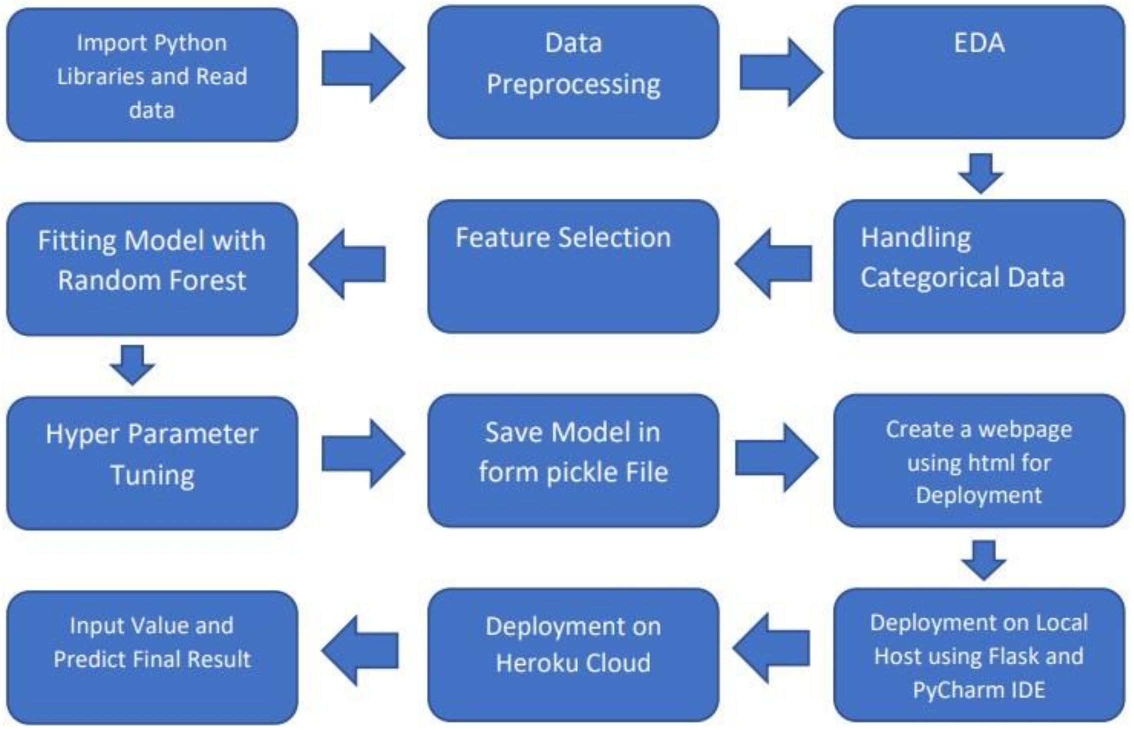
The recent international things had a large impact on the aviation sector because of several reasons. This impact has 2 class folks, the primary is business perspective and therefore the second is that the customers perspective. As safety is that the major reason for such impact on the aviation sector, the governments round the world amended totally different rules to their various airlines firms. These restrictions had created the supply of the flights and their attendant capability less. Taking of these factors in thought the value of the flight tickets has accrued and vary from one place to the opposite. Booking a flight price tag has split into 2, one is that the on-line and therefore the alternative is that the offline bookings. each these have their various criteria for value of the price tag, one such example is that the server load and therefore the range of booking requests. during this machine learning implementation, we are going to see numerous factors that impact worth of the flight ticket price and predict the acceptable price of the ticket.

1 Introduction

1.1 Why this Low-Level Design Document?

The main purpose of this LLD documentation is to feature the required details of the project and supply the outline of the machine learning model and also the written code. This additionally provides the careful description on however the complete project has been designed end-to-end.

1.2 Architecture


Import Python Libraries and Read data
Data Preprocessing
EDA
Handling Categorical Data
Feature Selection
Fitting Model with Random Forest
Hyper Parameter Tuning
Save Model in form pickle File
Create a webpage using html for Deployment
Deployment on Local Host using Flask and PyCharm IDE
Deployment on Heroku Cloud
Input Value and Predict Final Result

2. Architecture Design

This project is to make associate interface for the user to grasp their approximate flight price ticket worth, additionally to the present, in would like of obtaining the important time project expertise we have a tendency to square measure mercantilism the gathered information into our own information then begin the project from the scratch.

2.1. Data Gathering

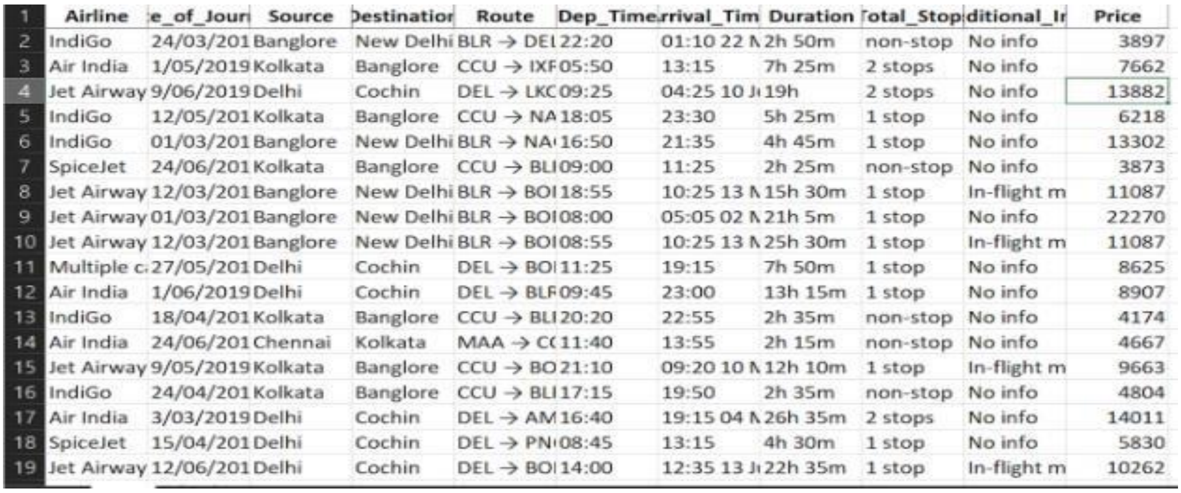
The data for the current project is being gathered from Kaggle dataset, the link to the data is: https://www.kaggle.com/nikhilmittal/flight-fare-prediction-mh

2.2. Tool Used

- Python 3.8 is employed because the programming language and frame works like numpy, pandas, sklearn, matplotlib and alternative modules for building the model.
- Visual Studio Code is employed as IDE.
- For visualizations seaborn and components of matplotlib are getting used.
- For information assortment prophetess info is getting used version management.
- AWS EC2 Instance is employed for deployment

2.3 Data Description

There are about 10k+ records of flight information such as airlines, data of journey, source, destination, departure time, arrival time, duration, total stops, additional information, and price. A glance of the dataset is shown below.



1	Airline	e_of_Jour	Source	Destinatior	Route	Dep_Time	rrival_Tim	Duration	otal_Stop	ditional_Ir	Price
2	IndiGo	24/03/201	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 M	2h 50m	non-stop	No info	3897
3	Air India	1/05/2019	Kolkata	Banglore	CCU → IXF	05:50	13:15	7h 25m	2 stops	No info	7662
4	Jet Airway	9/06/2019	Delhi	Cochin	DEL → LKC	09:25	04:25 10 J	19h	2 stops	No info	13882
5	IndiGo	12/05/201	Kolkata	Banglore	CCU → NA	18:05	23:30	5h 25m	1 stop	No info	6218
6	IndiGo	01/03/201	Banglore	New Delhi	BLR → NA	16:50	21:35	4h 45m	1 stop	No info	13302
7	SpiceJet	24/06/201	Kolkata	Banglore	CCU → BLI	09:00	11:25	2h 25m	non-stop	No info	3873
8	Jet Airway	12/03/201	Banglore	New Delhi	BLR → BOI	18:55	10:25 13 M	15h 30m	1 stop	In-flight m	11087
9	Jet Airway	01/03/201	Banglore	New Delhi	BLR → BOI	08:00	05:05 02 M	21h 5m	1 stop	No info	22270
10	Jet Airway	12/03/201	Banglore	New Delhi	BLR → BOI	08:55	10:25 13 M	25h 30m	1 stop	In-flight m	11087
11	Multiple c	27/05/201	Delhi	Cochin	DEL → BOI	11:25	19:15	7h 50m	1 stop	No info	8625
12	Air India	1/06/2019	Delhi	Cochin	DEL → BLF	09:45	23:00	13h 15m	1 stop	No info	8907
13	IndiGo	18/04/201	Kolkata	Banglore	CCU → BLI	20:20	22:55	2h 35m	non-stop	No info	4174
14	Air India	24/06/201	Chennai	Kolkata	MAA → CC	11:40	13:55	2h 15m	non-stop	No info	4667
15	Jet Airway	9/05/2019	Kolkata	Banglore	CCU → BO	21:10	09:20 10 M	12h 10m	1 stop	In-flight m	9663
16	IndiGo	24/04/201	Kolkata	Banglore	CCU → BLI	17:15	19:50	2h 35m	non-stop	No info	4804
17	Air India	3/03/2019	Delhi	Cochin	DEL → AM	16:40	19:15 04 M	26h 35m	2 stops	No info	14011
18	SpiceJet	15/04/201	Delhi	Cochin	DEL → PN	08:45	13:15	4h 30m	1 stop	No info	5830
19	Jet Airway	12/06/201	Delhi	Cochin	DEL → BOI	14:00	12:35 13 J	22h 35m	1 stop	In-flight m	10262

2.4 Import Data into Database

- Created associate api for the transfer of the info into the Cassandra info, steps performed are:
- Connection is created with the info.
- Created a info with name flightfare.
- Cqlsh command is written for making the info table with needed parameters.
- And finally, a cqlsh command is written for uploading the knowledgeset into data table by bulk insertion.

2.5 Export Data into Database

In the above created api, the download url is also being created, which downloads the data into a csv file format.

2.6 Data Preprocessing

Steps performed in pre-processing are:

- First the info sorts square measure being checked and located solely the value column is of sort number.
- Checked for null values as there square measure few null values, those rows square measure born.
- Converted all the desired column into the date time format.
- Performed one-hot cryptography for the desired columns.
- Scaling is performed for needed information.
- And, the info is prepared for passing to the machine learning formula

2.7 Modelling

The pre-processed information is then envisioned and every one the specified insights are being drawn. though from the drawn insights, the info is at random unfold however still modelling is performed with completely different machine learning algorithms to form positive we tend to cowl all the chances. and eventually, for sure random forest regression performed well and any hyperparameter calibration is finished to extend the model's accuracy.

2.8 UI Integration

Both CSS and HTML files are being created and are being integrated with the created machine learning model. All the required files are then integrated to the app.py file and tested locally

2.3 Data from User

The data from the user is retrieved from the created HTML web page.

2.4 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.

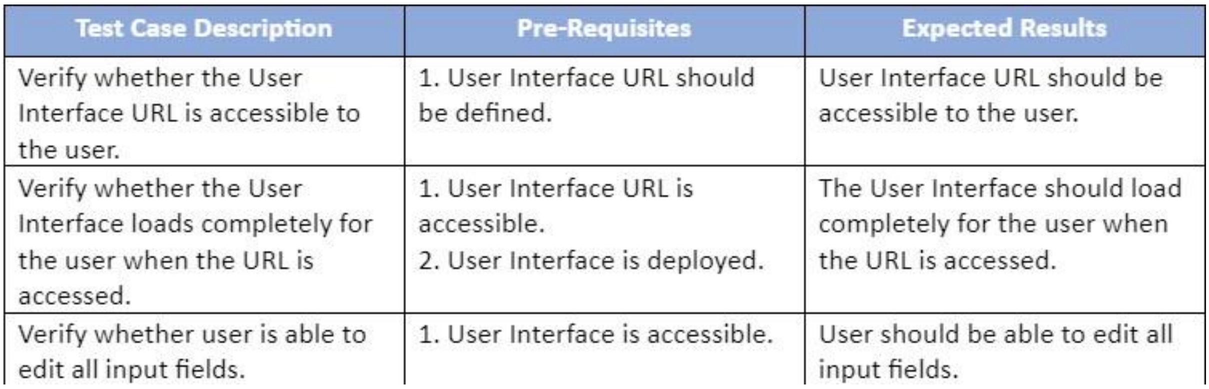
2.11 Rendering Result

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

3. Deployment

The tested model is then deployed on local machine and AWS EC2 Instance. So, users can access the project from any internet devices.

3.1 Unit Test



Test Case Description	Pre-Requisites	Expected Results
Verify whether the User Interface URL is accessible to the user.	1. User Interface URL should be defined.	User Interface URL should be accessible to the user.
Verify whether the User Interface loads completely for the user when the URL is accessed.	1. User Interface URL is accessible. 2. User Interface is deployed.	The User Interface should load completely for the user when the URL is accessed.
Verify whether user is able to edit all input fields.	1. User Interface is accessible.	User should be able to edit all input fields.