Low Level Design (LLD)
FLIGHTFARE PREDICTION

Document Version: 0.3

Last Revised Date: 14-06-2023

Rajesh Singh Gurjar

Document Version Control:

Version	Date	Author	Description
0.1	10-06-2023	Rajesh	Introduction Architecture, Document Content, Version Control and Unit Test cases to be added
0.2	11-06-2023	Rajesh	Data Pre-processing, Model Building
0.3	14-06-2023	Rajesh	Deployment

Contents

1. Introduction

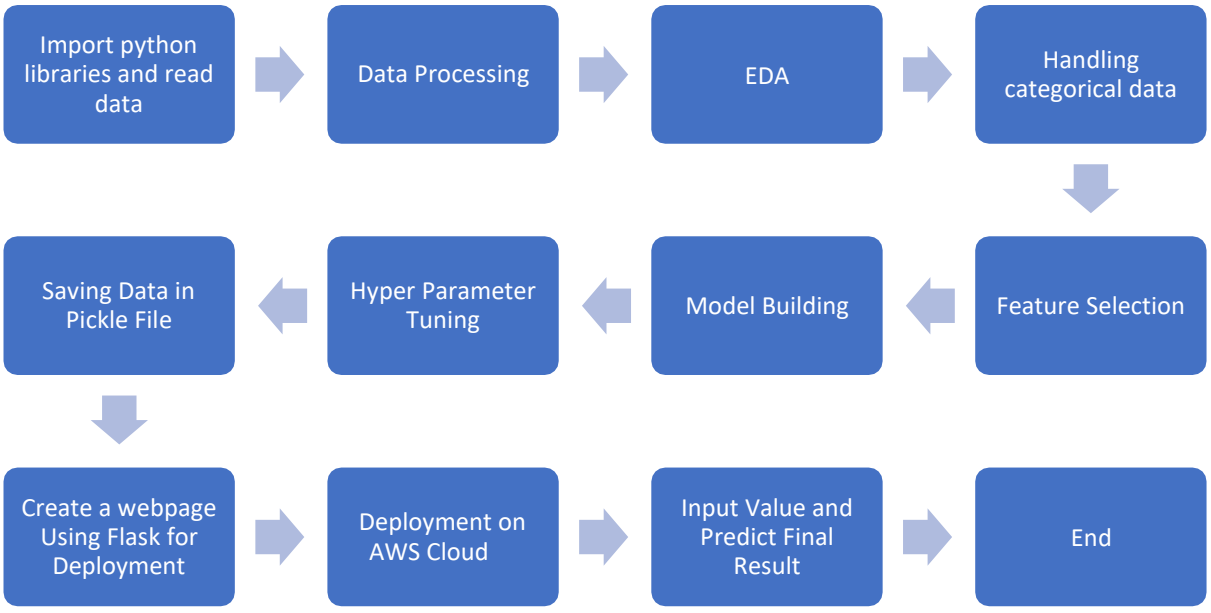
1.1. What is Low-Level design document?

The main purpose of the LLD document is to feature the required detailed description of the project and provide the outline of the different stages of machine learning end-to-end project along with its deployment.

1.2. Scope

Low-level design (LLD) is a component-level design process that follows a step-bystep refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work

2. Architecture



3. Architecture Description

This project aims to create a user interface that provides users with approximate flight ticket prices. Additionally, to gain real-time project experience, we have extracted the data, processed it, build model by inferring the data, did hyperparameter tuning to optimize the model, built batch prediction pipeline, instance prediction API and then deployed it on AWS EC2 instance.

3.1. 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

3.2. 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.

	Airline	e_of_Jour	Source	Destinatior	Route	Dep_Time	rrival_Tim	Duration	otal_Stop	ditional_I	Price
2	IndiGo	24/03/201	Banglore	New Delhi	BLR → DEI	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 Ju	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 → BO	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 → BO	14:00	12:35 13 Ju	22h 35m	1 stop	In-flight m	10262

3.3. Tool Used

- Python 3.8 is used as the programming language, along with frameworks like numpy, pandas, scikit-learn, and other modules for building the model.

- VS Code is employed as IDE.

- For visualization, seaborn and components of matplotlib are used.

- GitHub is used as a repository for the source code. It is used for the version management. GitHub Actions is used, which is a powerful workflow automation and CI/CD (Continuous Integration/Continuous Deployment) platform provided by GitHub. It allows you to automate tasks and workflows directly within your GitHub repository.

- AWS EC2 is used for deployment. At AWS, ECR is used to maintain the Docker image of the project.

3.4. Data Pre-processing

- Initiating the pr-processing by removing the missing rows from the data.
- Removing the duplicate records from the data.
- Splitting date, time and duration features and converting them into integers.
- Encoding the categorical data into integers using respective dictionaries.
- Saving the dictionaries to encode the input values during prediction.

3.4. Model Building

The pre-processed data is visualized, and relevant insights are extracted. Although the insights are randomly distributed, we perform modeling using various machine learning algorithms to cover all possibilities. Finally, random forest regression proves to e the best performing algorithm, and hyperparameter tuning is conducted to enhance the model's accuracy.

3.5. Data from User

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

3.6. 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.

3.7. Rendering Result

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

3.8. Deployment

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