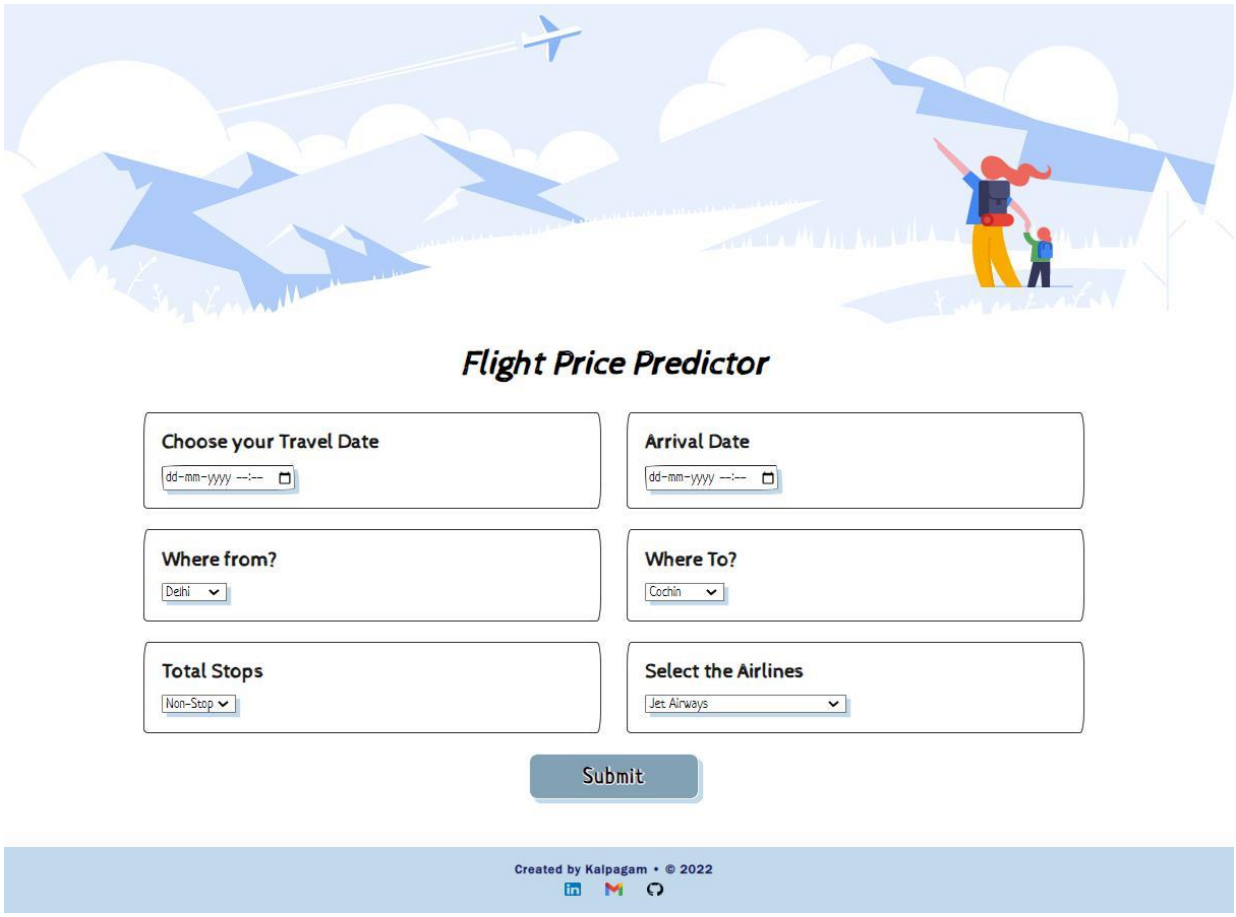
Low Level Design (LLD)

FLIGHT FARE PREDICTION



Contents

ABSTRACT

Traveling through flights has become an integral part of today's lifestyle as more and more people are opting for faster traveling options. The flight ticket prices increase or decrease every now and then depending on various factors like timing of the flights, destination, duration of flights. various occasions such as vacations or festive seasons. Therefore, having some basic idea of the flight fares before planning the trip will surely help many people save money and time. In the proposed system a predictive model will be created by applying machine learning algorithms to the collected historical data of flights. This system will give people the idea about the trends that prices follow and also provide a predicted price value which they can refer to before booking their flight tickets to save money. This kind of system or service can be provided to the customers by flight booking companies which will help the customers to book their tickets accordingly

1. INTRODUCTION

1.1 Purpose of this Low-Level 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 a careful description of the complete project design and end-to-end implementation.

2. Architecture Design

Machine learning life cycle is implemented in this project to create a basic web application which will predict the flight prices by applying machine learning algorithms to historical flight data using python libraries like Pandas, Matplotlib, Seaborn and scikit-learn.

2.1 Data Gathering

Data used in this project is publicly available at Kaggle.

Dataset link:
https://www.kaggle.com/datasets/nikhilmittal/flight-fare-prediction-mh

2.2 Tools used

- Python is used as the programming language and frameworks like pandas, scikit-learn for building the model.
- Jupyter notebook and Visual Studio Code is used as IDE.
- Seaborn and Matplotlib are used for visualization.
- HTML and CSS are used for front end development.
- Flask is used as a backend.

2.3 Data Description

The main data set is a training set and test set. The training set contains all the features, along with flight fares. It contains a (continuous) numerical value. This dataset has prices of flight tickets for various airlines between the months of March and June of 2019 for various cities.

Training Set:

Description: 10683 records, 10 input features and 1 output column 'Price'.

Test Set:

Description: 2671 records and 10 input features. The output 'Price' column needs to be predicted in this set.

We will use Regression techniques here, since the predicted output will be a continuous value.

Description of Features:

1. Airline: The name of the airline.

2. Date_of_Journey: The date of the journey

3. Source: The source from which the service begins.

4. Destination: The destination where the service ends.

5. Route: The route taken by the flight to reach the destination.

6. Dep_Time: The time when the journey starts from the source.

7. Arrival_Time: Time of arrival at the destination

8. Duration: Total duration of the flight.

9. Total_Stops: Total stops between the source and destination.

10. Additional_Info: Additional information about flight

11. Price: The price of the ticket. (Target column)

3. STEPS INVOLVED IN CREATING THIS PROJECT

1. Importing Python libraries and loading the dataset
2. Data Inspection
3. Exploratory Data Analysis (EDA)
4. Data pre-processing
5. Feature selection
6. Building ML models
7. Performance Metrics
8. Hyperparameter tuning
9. Saving the model to a pickle file
10. Creating flask app
11. Creating front end using HTML and CSS
12. UI Integration
13. App Deployment on Render

3.1 Importing Python libraries and loading the dataset

The necessary python libraries such as

- Pandas
- Numpy
- Matplotlib
- Seaborn
- Scikit-learn

are imported and the dataset in the form of CSV file is imported.

3.2 Data Inspection

In this step the size and shape of the dataset is inspected. The appropriate data types for each column in the data frame and summary is checked for numerical features.

3.3 Exploratory Data Analysis (EDA)

Visualizations are done using Matplotlib and Seaborn libraries to get some insights about the dataset.

3.4 Data pre-processing

This is done to transform raw data into features that can be used for creating a predictive model using Machine learning. Steps involved in pre-processing are:

- Converting the string data type into the desired data type.
- Extracting the Date Column separately as Day, Month, Year.
- Hours and minutes are extracted from Departure Time.
- Checking for null values and filling them.
- Performing Label Encoding to convert categorical values into numerical values.

3.5 Feature Selection

This step helps in finding out the best feature which will contribute and have good relation with the target variable (Price). Heat map is used here to find correlation between features.

3.6 Building ML models

After selecting the features which are more correlated to price the next step involves applying machine algorithms and creating a model. As our dataset consists of labeled data, we use Supervised machine learning algorithms. We will apply Regression Algorithms to solve this use case as our dataset contains continuous values in the target column. Regression models are used to describe relationships between dependent and independent variables. The machine learning algorithms used in this project are:

- K neighbors regressor
- Decision Tree regressor
- Random Forest Regressor

3.7 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. The following metrics will be used to check the error measure of each model.

- R2_score
- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)

After training and building the models, accuracies are checked for the 3 models and the model with maximum accuracy is selected. The **Random Forest Regressor model** turned out to be the most accurate one out of the 3 models.

3.8 Hyperparameter tuning

Accuracy is improved by doing Hyperparameter tuning. The random Search method is used to find the best parameters. After, the accuracy is found to increase.

3.9 Saving the model to a pickle file

The model is then saved to a pickle file for future use.

3.10 Creating flask app

Now a web application is designed where the user can input all the attribute values and the data will be given to the model, based on the training given to the model, the model will predict the flight ticket fares.

The back end is created using Flask Framework where API endpoints such as GET and POST will be created to perform operations related to fetching and displaying data on the front end of the application.

3.11 Creating front end using HTML and CSS.

The front-end of the application will be created using HTML, CSS, Bootstrap framework where users will have the functionality of entering their flight data.

3.12 UI Integration

The data from the user is retrieved from the created HTML web page.
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 rendered to the front-end and displayed.

On clicking Submit, the predicted price is displayed on the screen as follows: