Flight Fare Prediction

Contents

Document Version Control

Date Issued	Version	Description	Author
16/10/202	1	Initial LLD — V1.0	Ankit Zanzmera

1. Introduction:

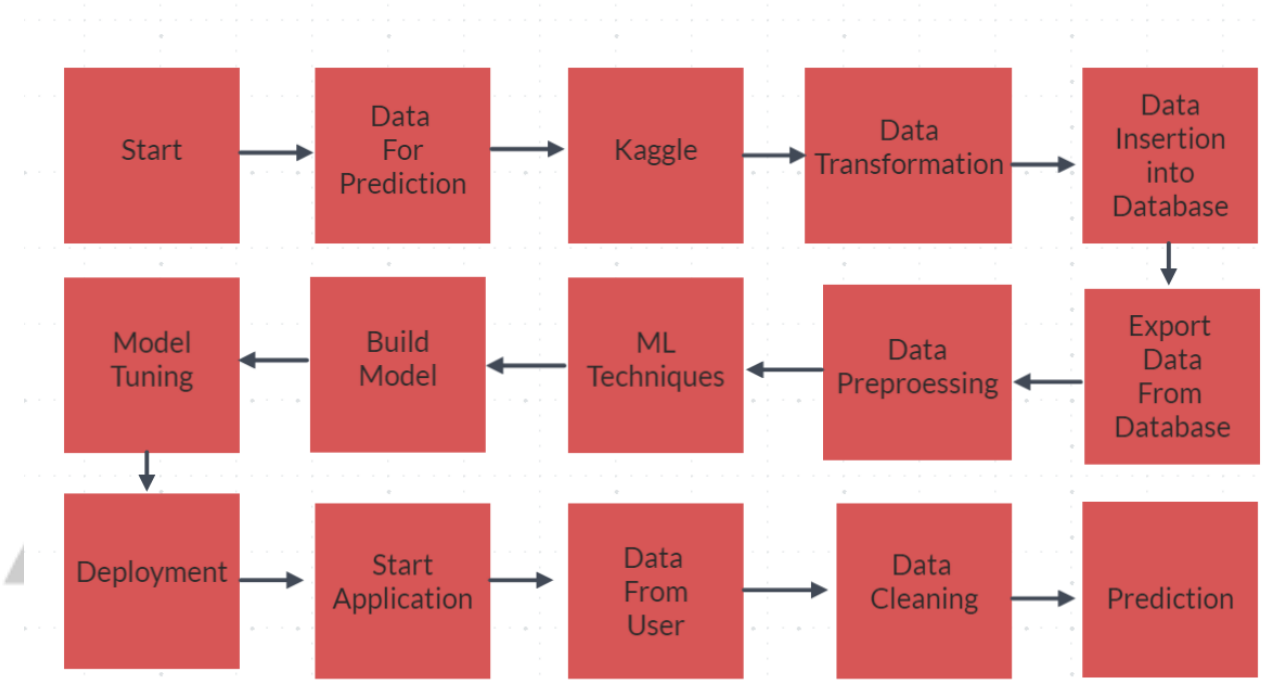
1.1. What is Low-Level design document?

A Low-Level Design (LLD) document is a detailed blueprint or specification that outlines how a specific component, module, or feature of a software system will be implemented. It is a critical step in the software development process and serves as a bridge between the high-level architectural design and the actual coding and implementation phase. The LLD document provides intricate technical details, including algorithms, data structures, class designs, and specific coding guidelines, to guide developers in building the software component.

1.2. Scope:

Low-level design (LLD) is a component-level design process that follows a step-by-step 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:

3.1. Data Description:

Dataset contains information about flight booking options from the website Easemytrip for flight travel between India's top 6 metro cities. There are 300261 datapoints and 11 features in the cleaned dataset.

3.2. Data Insertion Into Database:

Data insertion into a database is the process of adding, updating, or appending information into a structured storage system. This involves defining the data format, specifying the table or collection where the data should be stored, and using SQL statements or APIs to insert the data records.

3.3. Export from a Database:

Data Export from Database - The data in a stored database is exported as a CSV file to be used for Data Pre-processing and Model Training.

3.4. Data Preprocessing:

Data preprocessing is a crucial step in data preparation where raw data is transformed, cleaned, and organized to make it suitable for analysis or machine learning. This involves tasks like handling missing values, removing duplicates, scaling, and normalizing data, and converting categorical variables into numerical formats. Data preprocessing aims to enhance data quality, reduce noise, and prepare the dataset for more accurate and efficient analysis or model training. It is a foundational step in data science and plays a significant role in extracting valuable insights and patterns from complex datasets.

3.5. Model Building:

Model building is the process of creating a mathematical or computational representation that captures patterns and relationships within a dataset. This involves selecting an appropriate algorithm, defining model parameters, and training the model using the prepared dataset. The goal is to develop a predictive or descriptive model that can make accurate predictions or provide valuable insights based on the input data. Model building is a core component of machine learning and statistical analysis and is used to solve a wide range of problems, from image recognition to financial forecasting. It's a critical step in leveraging data for decision-making and automation in various domains.

3.6. Hyperparameter Tuning:

Hyperparameter tuning is the process of optimizing the settings and parameters of a machine learning model to improve its performance. This involves systematically adjusting hyperparameters, such as learning rates, regularization strengths, and tree depths, through techniques like grid search, random search, or Bayesian optimization. The goal is to fine-tune the model to achieve the best possible accuracy or predictive power on a given dataset. Hyperparameter tuning plays a pivotal role in optimizing the performance of machine learning models, ensuring they generalize well to new data and perform at their best in real-world applications. It is a key component of the model development process, contributing to better model accuracy and reliability.

3.7. Deployment:

Deployment in the context of software development refers to the process of making a software application or system available for use in a production environment. This typically involves taking the code that has been developed, tested, and approved and making it accessible to end-users or clients**.**

3.8. Data From User:

In My project, data entry by users is a fundamental process that allows individuals to input information or interact with the system. This data can encompass a wide range of inputs, depending on the project's specific requirements. For instance, in the context of our Flight Fare Prediction Project, users might provide details like departure and arrival locations, travel dates, class preferences, and passenger information.

3.9. Prediction:

The prediction aspect of our project is at the heart of its functionality, and it plays a pivotal role in assisting users and stakeholders. Predictions are the outcomes generated by our model based on the input data provided. This process involves a series of steps, including data preprocessing and feature extraction, culminating in the application of machine learning algorithms to produce estimations or forecasts. In the context of our Flight Fare Prediction Project, these predictions translate into estimated flight fares for specific routes and dates, which serve as invaluable insights for travelers.

4. Conclusion:

In conclusion, the Low-Level Design (LLD) document provides a comprehensive and detailed blueprint for the implementation of our project. It meticulously outlines the technical aspects, algorithms, data structures, and class designs that will underpin the development of our software component. This document is the critical bridge between high-level architectural concepts and the actual coding and implementation phase.

With the LLD in hand, our development team is well-equipped to proceed with confidence, having a clear roadmap for building the specific component. This includes a deep understanding of the architectural layout, the flow of data, the interactions between different modules, and the expected behavior of the software.