Low Level Design(LLD)

Insurance Premium Prediction

Written By	Vijay shinde
Document Version	0.2
Last Revised Date	13/06/2023

Document Version Control

Version	Date	Author	Comments
0.1	10/06/2023	Vijay shinde	Abstract Introduction Architecture
0.1	12/06/2023	Vijay shinde	Architectural Design
0.2	14/06/2023	Vijay shinde	Deployment Unit Test Cases

Contents

1. Introduction

What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Insurance Premium Prediction. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

A Low-Level Design (LLD) document is a detailed technical document that provides a comprehensive description of how a particular system or software component will be implemented. It is a step closer to the actual implementation, providing detailed information on the structure, modules, functions, and interactions of the system.

The purpose of the LLD document is to guide the developers during the coding phase by specifying the internal workings of the system. It serves as a blueprint for the actual implementation and helps ensure that the software is developed according to the desired specifications.

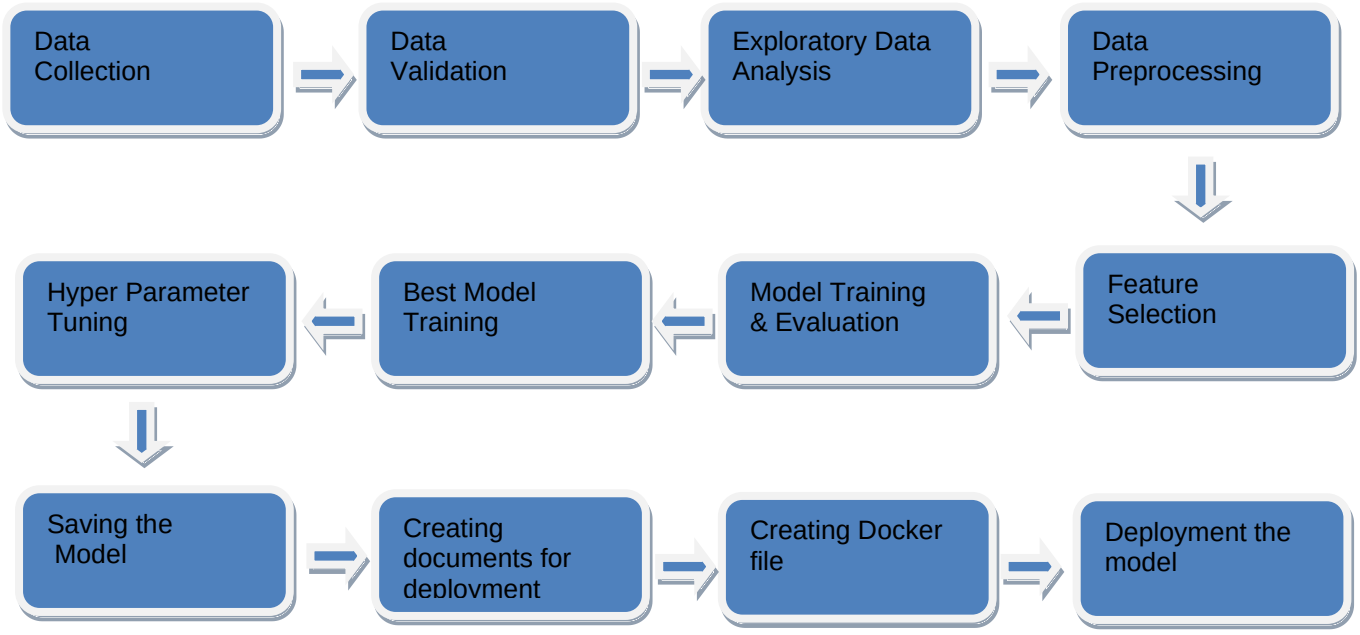
Scope

The scope of a project or document refers to the boundaries and extent of what is included and addressed within that project or document. It defines the specific goals, objectives, deliverables, and tasks that are within the purview of the project or document.

In the context of a Low-Level Design (LLD) document, the scope would typically involve outlining the detailed design and implementation aspects of a specific system or software component. This includes the modules, algorithms, data structures, interfaces, and other relevant details that are necessary for the development of that particular system or component.

The scope of an LLD document can vary based on the project requirements and complexity. It generally focuses on the internal workings and technical aspects of the system, providing a more detailed view compared to higher-level design documents

2. Architecture



3. Architecture Description

Data Collection

In this Data is collected from mongo DB data base

Data Description

The dataset is consists of 1338 records with 6 columns, including age, sex, gender, region, BMI, etc.

Data Ingestion

During the ingestion process, the initial dataset in the .zip format will be converted into the CSV format. Subsequently, the dataset will be divided into the train and test datasets.

Data Validation

In the data validation steps, various techniques can be employed to ensure the quality and reliability of the dataset. These techniques include handling null values, addressing outliers, managing imbalanced datasets, and handling columns with low standard deviation or values below a certain threshold.

Data Transformation

In this step, we will transform our data. For numeric data, we will utilize the standard scalar to standardize the values and make them compatible for analysis. Additionally, for categorical data, we will apply the one-hot encoding technique to convert it into a numeric representation. This conversion allows the machine to effectively interpret and process the categorical variables..

Model Building

We can observe that the problem statement is a supervised machine learning problem. In order to address this, we have implemented various regression models. Moving forward, we will proceed to build the machine learning model using all available regression algorithms.

Model Evaluation

In this stage, model evaluation will be performed on the model that was built in the previous stage. The base accuracy of the model will be defined, and only if the model's accuracy surpasses the base accuracy, it will be accepted. Otherwise, the model will be rejected.

Model Deployment

The model will be deployed on the Streamlit cloud platform.

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	Application URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	Application URL is accessible Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether user is able to see input fields on application	Application is accessible User is logged in to the application	User should be able to see input fields on application
Verify whether user is able to edit all input fields	Application is Accessible User is logged in to the application	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs Verify whether user is getting predicted results on clicking	Application is accessible User is logged in to the application	User should get Submit button to submit the inputs User should be presented with predicted results on clicking