LOW LEVEL DESIGN

Adult Census Income Prediction

Saakar Sengar

Shreyansh Jain

Contents

1. Introduction……

1.1 What is Low Level Design documentation?...

1.2 Scope

2. Architecture

3. Description

3.1 Data Description

3.2 Data

3.3 Pre-processing

3.4 Model Building

3.5 Cloud Setup

3.6 Deployment

3.7 User Data

3.8 User Data insertion into Database

3.9 Data Pre-processing

3.10 Loading Model

3.11 Model Prediction

3.12 Saving Output to Database

1. Introduction

1.1 What is Low Level Design Documentation?

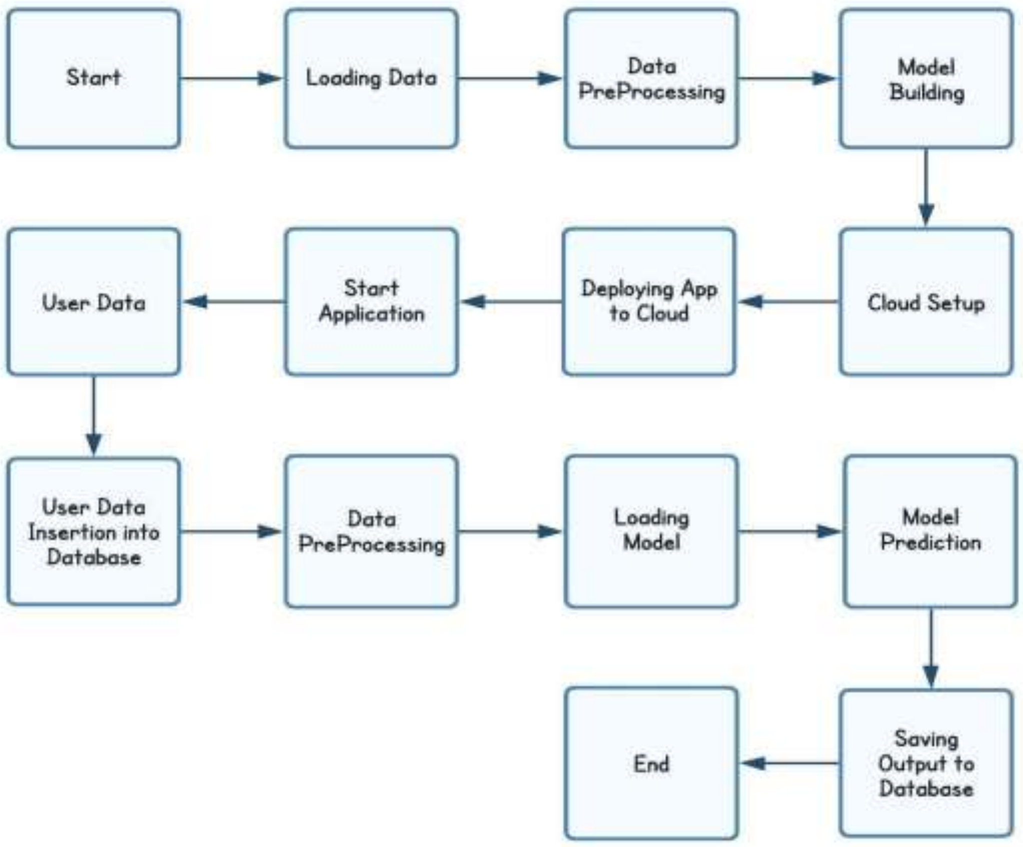
The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Adult Census Income Prediction. It explains the purpose and features of the system, the interface of the system. LLD describes the class diagrams with the methods and relations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

The goal is to predict whether a person has an income of more than 50K a year or not. This is basically a binary classification problem where a person is classified into the >50K group or <=50K group.

1.2 Scope

This software system will be a Web application. This system will be designed to predict whether an individual will have annual income of more than 50K or less than 50K based on the individual’s educational history, previous or current working experience, demographic locations and many other features related to the individual.

2. Architecture



3.Architecture Description

3.1 Data Description

The dataset used is the Adult Census Dataset. The dataset contains 15 feature columns and 32561 rows.

3.2 Data Pre-processing

Pre-processing is done to handle null values, punctuations, categorizing the categorical features to correct format, encoding the categorical features and scaling of the numerical features of the dataset. Finally, the most relevant features are used for building the machine learning model.

3.3 Model Building

After the pre-processing of the data is completed. The dataset is used to train various models and for each model a grid of parameters is passed using GridSearchCV so that the models find its most suitable parameters. The best model is chosen based on F1 score and the AUC score of the models.

3.4 Cloud Setup

During this step we build the web api using Flask web frame work, HTML, CSS and do the basic setup for the database and the logger.

3.5 Deployment

The web api is deployed to Heroku. Before deploying to Heroku some basic requirements have to be fulfilled so that the model works properly on the server.

3.6 Data from User

After the web api is deployed to Heroku, the web api is accessible to the user. We collect the data from the users.

3.7 Data Insertion into Database

The data collected from the user is stored into the database. The database used is Astra DB.

3.8 Pre-processing of User Data

Once the user data is retrieved and stored into the database. The data is pre-processed so it becomes suitable for the model to predict the output.

3.9 Loading Model

Once the pre-processing of the user data is completed. The model is loaded.

3.10 Model Prediction

The model predicts on the input data after it has been pre-processed.

3.11 Saving Output to Database

The model prediction is stored into the database.

4.Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. 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	1. Application URL is accessible 2. Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether the User is able to sign up in the application	1. Application is accessible	The User should be able to sign up in the application
Verify whether user is able to edit all input fields	1. Application is accessible	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	1. Application is accessible	User should get Submit button to submit the inputs

Verify whether user is presented with Predictions results on clicking submit	1. Application is accessible	User should be presented with Predicted results on clicking submit
Verify whether the predicted results are in accordance to the selections user made	1. Application is accessible	The predicted results should be in accordance to the selections user made