Low Level Design

Thyroid Disease Detection System

Written By	Umasangavi G
Document Version	1.0
Last Revised Date	

Document Version Control

Change Record:

Version	Date	Author	Comments
1.0	11-11-2021	Umasangavi G	LLD

Reviews:

Version	Date	Reviewer	Comments

Approval Status:

Version	Review Date	Reviewed By	Approved By	Comments

Contents

1. Introduction

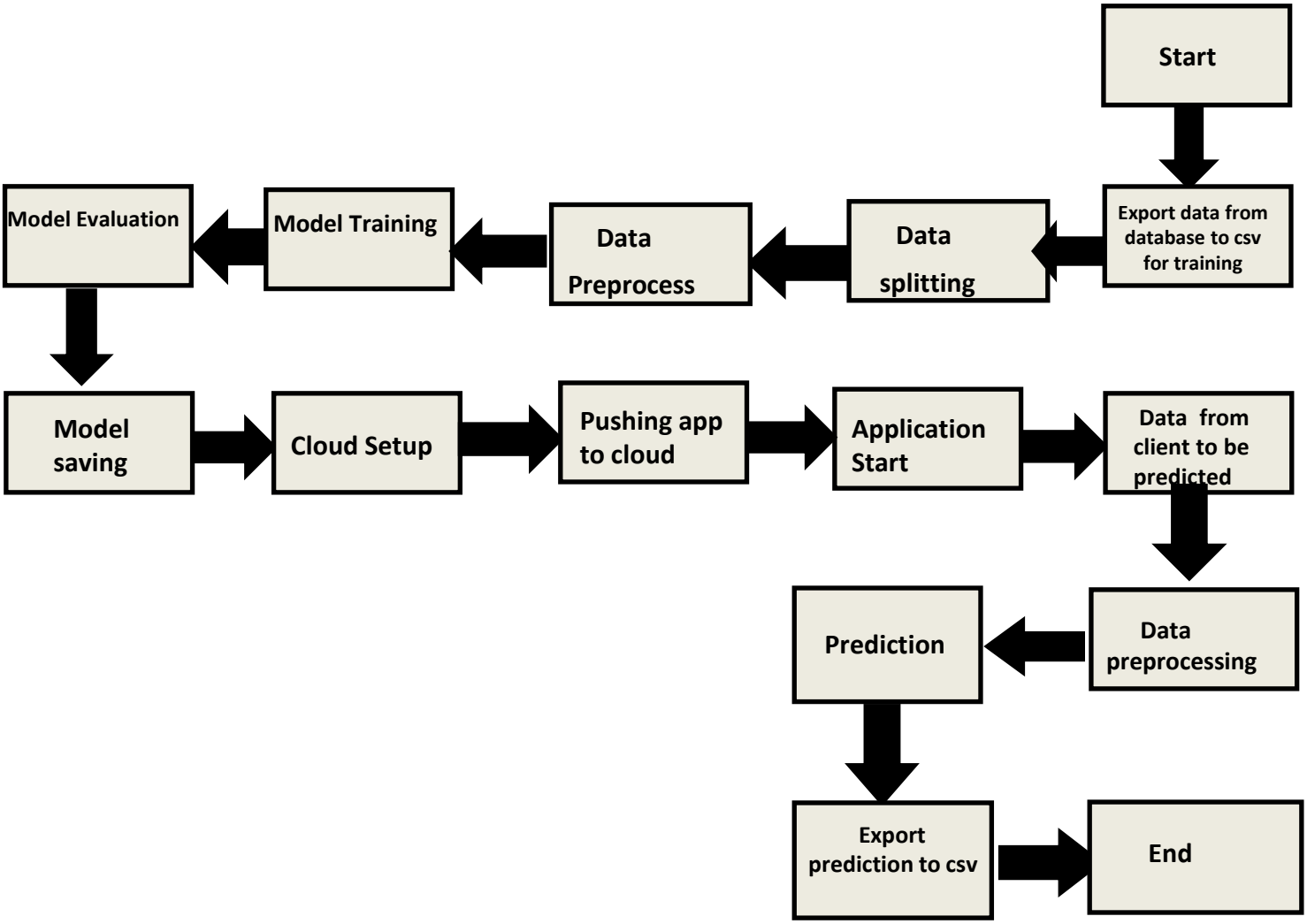
1.1 What is Low-Level design document?

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

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

We will be using Thyroid Disease Data Set present in UCI Machine Learning Repository. This Data set is satisfying our data requirement. Total 3772 instances present in different batches of data.

3.2 Export Data from database to CSV for Training

Here we will be exporting all batches of data from database into one csv file for training.

3.3 Data Splitting

We filter the columns for splitting the data for train and test for further uses

3.4 Data Preprocessing

We will be exploring our data set here and do EDA if required and perform data preprocessing depending on the data set. We first explore our data set in Jupyter Notebook and decide what pre-processing and Validation we have to do such as imputation of null values, etc and then we have to write separate modules according to our analysis, so that we can implement that for training as well as prediction data.

3.5 Data Training

We trained a RandomForestClassifier model in our notebook and was good on it. We trained with our processed data.

3.6 Model Evaluation

Model evaluation done by classification and report was saved to .pkl file

3.7 Model Saving

we will save our models so that we can use them for prediction purpose.

3.8 Cloud Setup

Here We will do cloud setup for model deployment. Here we also create our flask app and user interface and integrate our model with flask app and UI

3.9 Push app to cloud

After doing cloud setup and checking app locally, we will push our app to cloud to start the application.

3.10 Data from client side for prediction purpose

Now our application on cloud is ready for doing prediction. The prediction data which we receive from client side .

3.11 Data processing and Prediction

Client data will also go along the same process **Data pre-processing** and according to that we will predict those data.

3.12 Export Prediction to CSV

Finally when we get all the prediction for client data, then our final task is to export prediction to csv file and hand over it to client.

Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL isaccessible to the user	1. Application URL should be defined	Application URL should beaccessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	1.Application URLis 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 successfully login to the application	1. Application isaccessible 2.User is signed upto the application	User should be able to successfully login to the application
Verify whether user is able to see input fields on logging in	1. Application isaccessible 2.User is signed upto the application 3.User is logged into the application	User should be able to see input fields on logging in
Verify whether user is able to edit all input fields	1.Application isaccessible 2.User is signed upto the application 3.User is logged into the application	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	1. Application isaccessible 2.User is signed upto the application 3.User is logged into the application	User should get Submit button to submit the inputs
Verify whether user get prediction/output back after submitting the inputs	1. Application is accessible 2.User is signed up to the application 3.User is logged in to the application	User should get their output after submitting the inputs.
Verify whether the output which user get Is accordance to inputs user made.	1. Application is accessible 2.User is signed up to the application 3.User is logged in to the application	The output should be in accordance with the inputs user made.

Verify whether user have option to download their result or not.	1. Application is accessible 2.User is signed up to the application 3.User is logged in to the application	User should have option to download their output result.