THYROID DISEASE DETECTION

Low-Level Design (LLD)

Revision Number: 1.0

Last date of revision:01-Sept-2023

Document Version Control

Date Issued	Version	Description	Author
01-Sept-2023	1.0	CompleteLLD	Sahdev Saini

Contents

Document Version control ____________________ 2

1 Introduction ____________________ 4
1.1 What is Low-Level Design Document------------------4
1.2 Scope 4

2 Architecture 5

3 Architecture Description ____________________ 6
3.1 Data Description ____________________ 6
3.2 Export Data from DB to CSV for training ------------------ 6
3.3 Data Preprocessing ____________________ 6
3.4 Data Clustering ____________________ 6
3.5 Get best model of each cluster ------------------6
3.6 Hyperparameter Tuning ____________________ 6
3.7 Model saving ____________________ 7
3.8 Cloud setup 7
3.9 Push App to cloud ____________________ 7
3.10 Data from client side for prediction ------------------ 7
3.11 Export prediction to CSV ____________________ 7

1. Introduction

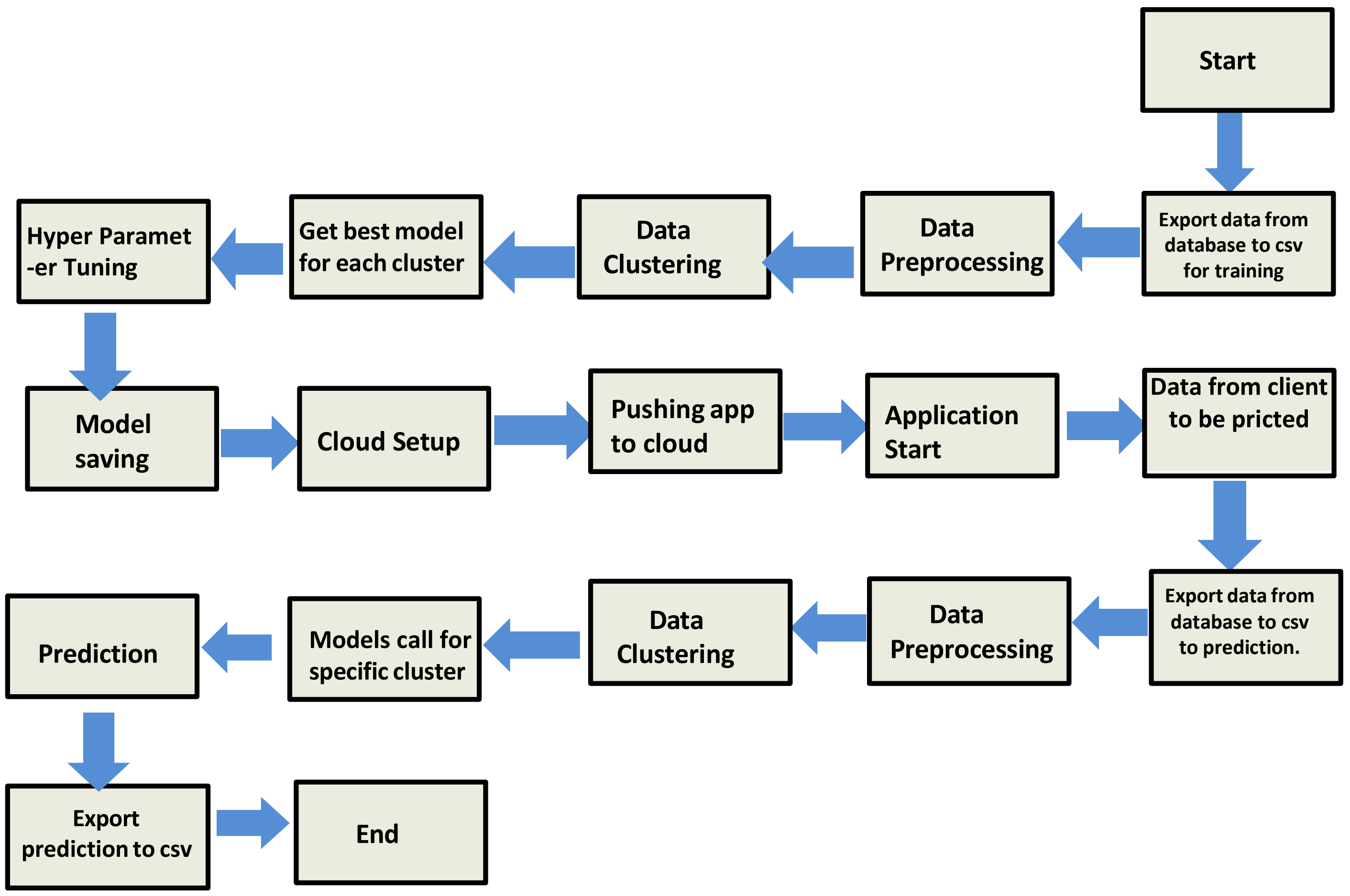
1.1 What is a Low-Level design document?

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 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.

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 7200 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 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 Jupiter Notebook and decide what pre-processing and Validation we must do such as imputation of null values, dropping some column, etc. and then we must write separate modules according to our analysis, so that we can implement that for training as well as prediction data.

3.4 Data Clustering

Random Forest is an ensemble learning method that builds multiple decision trees during training. Each tree is constructed independently using a random subset of the features and a random subset of the training data. The final prediction is made by aggregating the predictions of all the individual trees (e.g., averaging for regression, voting for classification).

3.5 Get the best model of each cluster.

Here we will train various models on each cluster which we will obtain in Data Clustering, and then will try to get the best model of each cluster.

3.6 Hyperparameter Tuning

After selecting the best model for each cluster, we will do hyperparameter tuning for each selected model, and try to increase the performance of the models.

3.7 Model Saving

After performing hyperparameter tuning for models, we will save our models so that we can use them for prediction purposes.

3.8 Data from client side for prediction purpose

Now our application on cloud is ready for prediction. The prediction data which we receive from the client side will be exported from DB and further will do same data cleansing process as we have done for training data using modules, we will write for training data. Client data will also go along the same process of **Exporting data from DB**, **Data pre-processing**, **Data clustering** and according to each cluster number we will use our **saved model** for prediction on that cluster.

3.9 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.

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
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 is accessible. 2. User is signed up to 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 is accessible. 2. Users are signed up to the application. 3. User is logged in to 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 is accessible. 2. Users are signed up to the application. 3. 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	1. Application is accessible. 2. User is signed up to the application. 3. User is logged in to the application	User should get Submit button to submit the inputs