Low Level Design

Thyroid Disease Prediction Model

Written By	Ravi Yadav
Document Version	1.0
Last Revised Date	4 – Dec -2023

Document Control

Change Record:

Version	Date	Author	Comments
1.0	4 – Dec--2023	Ravi Yadav	Introduction & Architecture defined

Reviews:

Version	Date	Reviewer	Comments
1.0	4 –Dec-2023	Ravi Yadav	Document Content , Version Control and Unit Test Cases to be added

Approval Status:

Version	Review Date	Reviewed By	Approved By	Comments

Contents

1. Introduction

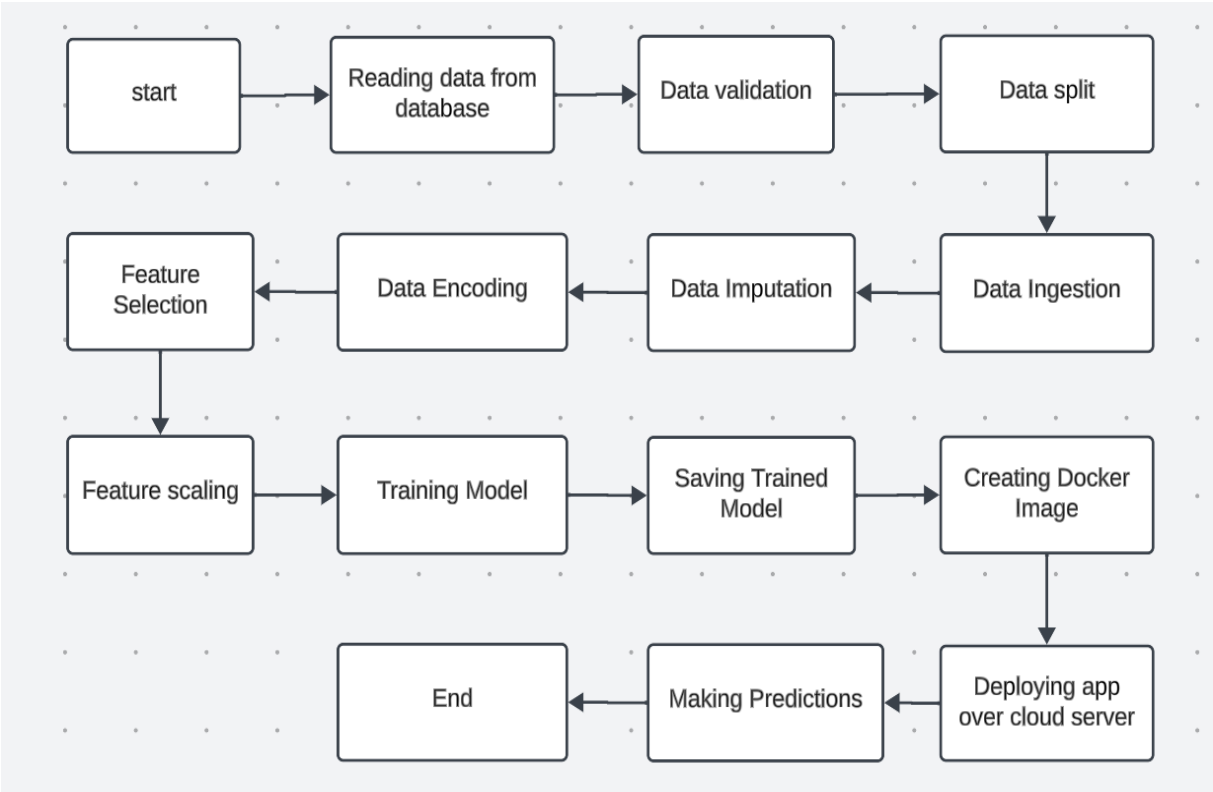
1.1. What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Thyroid Disease Prediction 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. Reading Data from Database

This step involves the reading data from table through python Cassandra driver. This involves the CQL i.e. Cassandra query language to read data from table.

3.2. Data validation

This step involves conversion of datatypes of feature according to their values. This step involves removal of unwanted characters from values.

3.3. Data splitting

After data validation, the raw data will be split into training and test dataset. Training dataset holds 80% data from raw data where test data will contain 20% data.

3.4. Data Ingestion

In this step, the training and testing data will be stored in a file having comma separated format. This kind of files are known as csv file having .csv extension.

3.5. Data Imputation

Data Imputation Technique helps in filling up the missing values of our dataset, various techniques are used to impute the missing value inside the dataset.

3.6. Data Encoding

Data Encoding is an important step which encodes the categorical features of dataset with the numeric value.

3.7. Feature Selection

Feature selection is a step which helps to get rid of curse of dimensionality in dataset. It uses various techniques which helps to get the features having high correlation with target feature.

3.8. Feature Scaling

Feature Scaling helps in scaling down the data in a single unit. This step helpful for the algorithms which are sensitive towards the outlier. Standard scaling and normalization is the technique of scaling down the data.

3.9. Training Model

Training Model is the step where pre-processed data is proceeded towards the various machine learning model and cross-validating the predictions with the data value.

3.10. Saving Training Model

After getting the best model from training model, the model with higher accuracy will stored in a pickle file which helps in avoiding the training of model again and again during deployment.

3.11. Creating Docker Image

A docker image is created in this step of machine learning application.

3.12. Deployment

App deployment is done in this step over a cloud server which can be accessed by a public user.

3.13. Making Predictions

The model created during training will be loaded, and predicts target value according the given inputs by the user.

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 user is able to see input fields on logging in	1. Application is 1. Application is accessible 2. User is 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. User is 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
Verify whether user is presented with recommended results on clicking submit	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be presented with recommended results on clicking submit
Verify whether the recommended results are in accordance to the selections user made	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	The recommended results should be in accordance to the selections user made