

Low Level Design Document

For

“Market Basket Project on E-Commerce”

Document Version Control

| SL. No | Version No. | Date | Prepared By | Reviewed By | Comments |
|--------|-------------|------------------------------|-------------|-------------|----------|
| 01 | 1.0 | 12 th April, 2022 | Irfan Razi | | |
| | | | | | |
| | | | | | |

Signature : _____

Prepared by : Irfan Razi
Intern – Data Scientist

Signature : _____

Reviewed by :

Signature : _____


Approved by : Sudhanshu Kumar
Founder iNeuron, Chief AI Engineer & CEO,
iNeuron.ai

Table of Contents


| | |
|--|---|
| Table of Contents | 2 |
| List of Figures..... | 2 |
| GENERAL..... | 3 |
| 1.1 Introduction | 4 |
| 1.2 Scope of this Document | 4 |
| ARCHITECTURE..... | 5 |
| 2 Architecture Details | 6 |
| 2.1 Data Description..... | 6 |
| 2.2 Data Ingestion | 6 |
| 2.3 Data Pre-processing | 7 |
| 2.4 Data Splitting | 7 |
| 2.5 Model Training | 7 |
| 2.6 Model Evaluation..... | 7 |
| 2.7 Model Saving | 7 |
| 2.8 Model Push to App..... | 7 |
| 2.9 Data from Client for testing | 7 |
| 2.10 Data Pre-processing | 7 |
| 2.11 Prediction | 7 |
| 3 Unit Test Cases | 8 |

List of Figures

| | |
|--|---|
| Figure 1: Architecture Details for Market Basket Project | 6 |
|--|---|

| | | | |
|--|-------------------------------------|-------------------------------------|-------------|
|  | Proprietary Confidential | Market Basket Project on E-Commerce | Issue: 1.0 |
| | | Low Level Design Document | Page 3 of 8 |

GENERAL

| | | | |
|---|-------------------------------------|-------------------------------------|-------------|
|  | Proprietary Confidential | Market Basket Project on E-Commerce | Issue: 1.0 |
| | | Low Level Design Document | Page 4 of 8 |

1.1 Introduction

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 describes the modules so that the programmer can directly code the program from the document.

1.2 Scope of this Document

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.

ARCHITECTURE

2 Architecture Details

This section describes the architecture details for Market Basket Project on E-Commerce.

Kindly, refer the Figure 1 for the same.

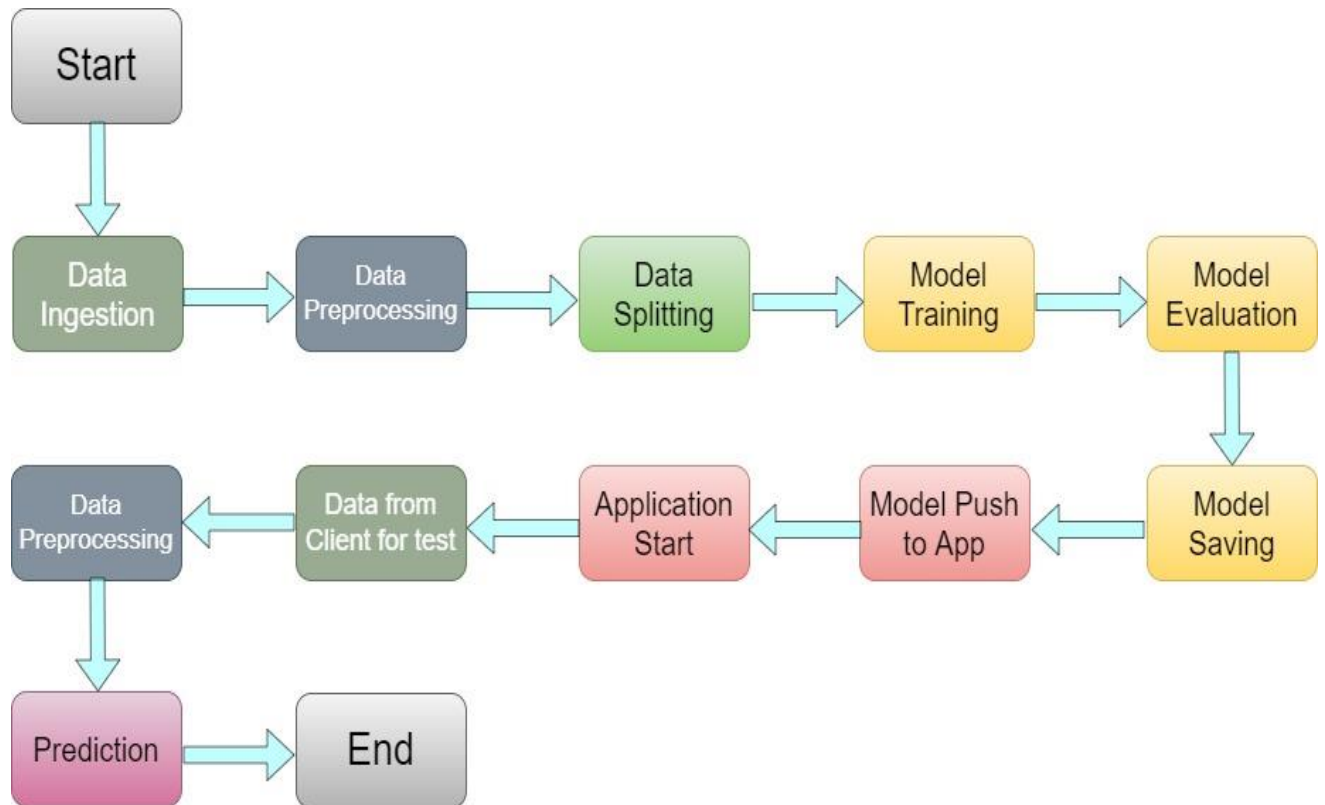


Figure 1: Architecture Details for Market Basket Project

2.1 Data Description

We will be using Brazilian E-Commerce Public Dataset by Olist. This is a Brazilian ecommerce public dataset of orders made at Olist Store. The dataset has information of 100k orders from 2016 to 2018 made at multiple marketplaces in Brazil. Its features allow viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers. We also released a geolocation dataset that relates Brazilian zip codes to lat/long coordinates.

2.2 Data Ingestion

Here, we will be ingesting all the batches of data from Cassandra database to our machine in csv format.

| | | | |
|---------|-----------------------------|-------------------------------------|-------------|
| iNeuron | Proprietary Confidential | Market Basket Project on E-Commerce | Issue: 1.0 |
| | | Low Level Design Document | Page 7 of 8 |

2.3 Data Pre-processing

We will do Exploratory Data Analysis of the data in Jupyter Notebook to get the complete understanding of the data. Based on that we can decide the strategy for Data Processing and validation. We may have to drop insignificant columns, handle missing values, handle imbalanced data, etc so that we can get a clean data for model training. For this, we have to write separate modules as per our need.

2.4 Data Splitting

We split the data for model training and model validation.

2.5 Model Training

We train our data with various ML models. Among those, Random Forest Classifier is the best fit model.

2.6 Model Evaluation

Model evaluation is done by classification report. Since, this is a problem of imbalanced data, we have to analyse and improve Recall score and F1-score, not just Accuracy.

2.7 Model Saving

After model training and evaluation, we will save the model for production.

2.8 Model Push to App

We are going to do the cloud setup for our model deployment. We are going to create Flask App and User Interface. We will integrate our model with it.

2.9 Data from Client for testing

Now, our Web-Application is ready and deployed to clouds. We can get the data from our clients and start testing the model.

2.10 Data Pre-processing

Client-data is also required to go through the same process as our train data has gone for model training.

2.11 Prediction

Finally, when we complete the prediction process with client's data, we convert it into csv format and share it to the client.

3 Unit Test Cases

We are going to give all the test cases with pre-requisites and expected results. Details are available in Table 1.

Table 1: Table for Test Cases and Expected Result

| 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. | 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. | Application is accessible. | User should be able to see input fields. |
| Verify whether user is able to edit all input fields. | Application is accessible. | User should be able to edit all input fields. |
| Verify whether user gets Submit button to submit the inputs. | Application is accessible. | User should get Submit button to submit the inputs. |
| Verify whether user is presented with result on clicking button. | Application is accessible. | User should be presented with clicking on submit button. |
| Verify whether the prediction result is displayed correctly according to the user input. | Application is accessible. | The prediction result should be displayed correctly according to user input. |