Low Level Document

Stores Sales Prediction

Written By	Derrick T
Document Version	0.1
Last Revised Date	20 – July -2023

Document Control

Change Record:

Version	Date	Author	Comments
0.1	20 – July - 2023	Derrick T	Created

Reviews:

Version	Date	Reviewer	Comments
0.1			

Approval Status:

Version	Review Date	Reviewed By	Approved By	Comments

Contents

1. Introduction

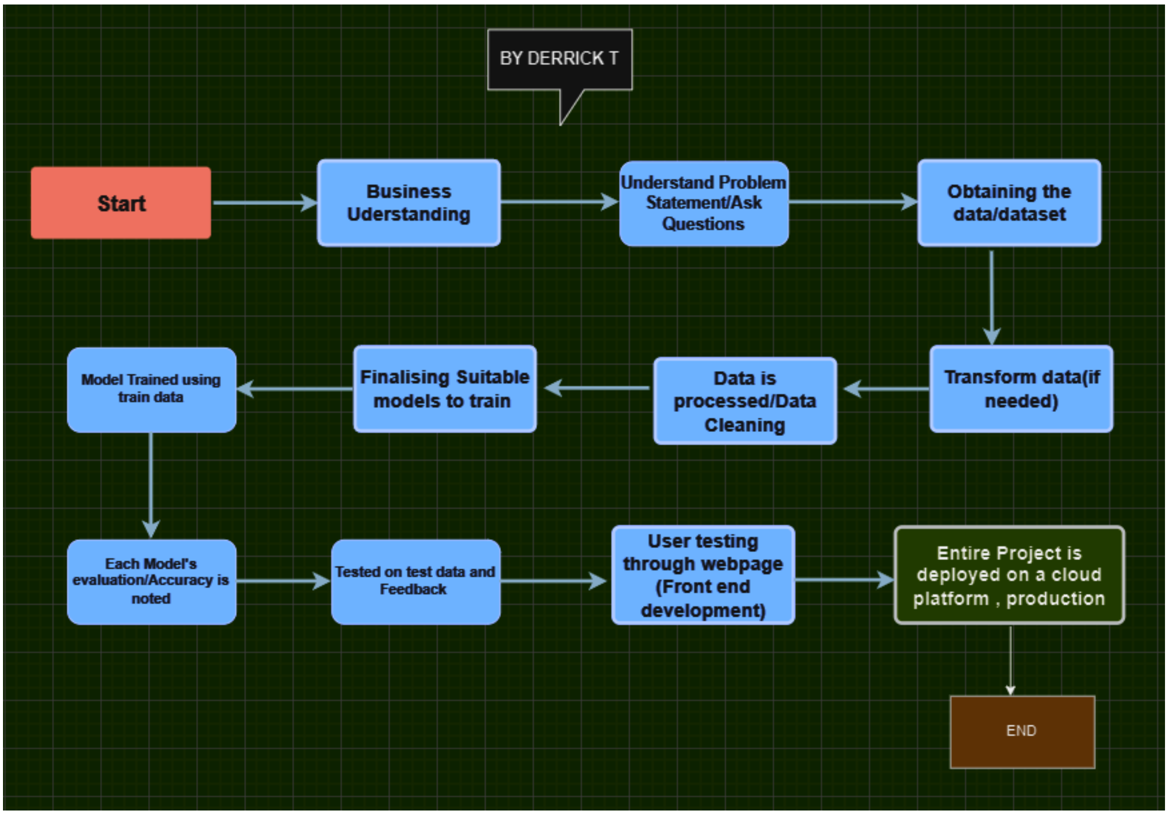
1.1. What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Food Recommendation 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


BY DERRICK T
Start
Business Understanding
Understand Problem Statement/Ask Questions
Obtaining the data/dataset
Transform data(if needed)
Data is processed/Data Cleaning
Finalising Suitable models to train
Model Trained using train data
Each Model's evaluation/Accuracy is noted
Tested on test data and Feedback
User testing through webpage (Front end development)
Entire Project is deployed on a cloud platform , production
END

3. Architecture Description

3.1. Data Description

Stores Sales Dataset, which contains information related to sales transactions across multiple stores. The dataset encompasses a comprehensive record of sales data, including various attributes associated with each transaction, such as date, store location, products sold, and corresponding sales metrics.

3.2. Web Scrapping

Was Not needed as the dataset was available as a .csv file

3.3. Data Transformation

Transformation was not required as the data was in .csv format when obtained.

3.4. Data Pre-processing

Data preprocessing involves cleaning, transforming, and organizing the data for analysis. Steps include handling missing data, addressing outliers, normalizing or standardizing numerical variables, encoding categorical variables, selecting relevant features, and handling skewed data. Additionally, data integration may be necessary when working with multiple datasets. Balancing imbalanced classes and splitting the dataset into training, validation, and test sets are important. Data visualization aids in understanding the pre-processed data. Specific steps may vary based on the dataset, objectives, and algorithms used.

3.5. Model Selection/Model Building

As the task was to predict, regression models and random forest models were chosen and trained.

Random forest model yielded better accuracy

3.6. Model Trained and Tested

Here the random forest model was tested again and was ready for the test data.

3.7. Model Evaluation

Random Forest model trained and tested and ready for deployment and yield results for new data

3.8. Designed a webpage which accepts user data (front-end)

Used HTML and Flask API to create an interface for the model and the user data. (to predict)

3.9. Deployment

The project was deployed on a cloud platform for public sharing/production

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the front-end page is displayed properly	Should run It from the correct browser	Should be able to see the input fields along with the designs
Verify whether the input fields accept the data correctly without any errors	Should enter appropriate values	All the input fields are being able to get filled and drop-down menu works
Verify whether the inputs are processed and output is generated	Appropriate values are entered	The user should be able to click the predict icon and a numerical value should be generated.
When Repeated Again.		
Verify whether user is able to see input fields	User runs it on the browser	User should be able to see input fields on logging in
Verify whether user is able to edit all input fields	Enter appropriate fields	User should be able to edit all input fields
Verify whether user gets Predict button to submit the inputs	Enter appropriate fields	User should get Predict button to submit the inputs and get predictions.