BigMart Sales Prediction

Low Level Document

Revision Number : 1.0

Last Updated : 13/09/2021

Document Version Control

Date Issued	Version	Description	Author
14/09/2021	1.0	First Draft - V1.0	Shreyas Kudav

Contents

1. Introduction

1.1 What is a High-Level Design Document?

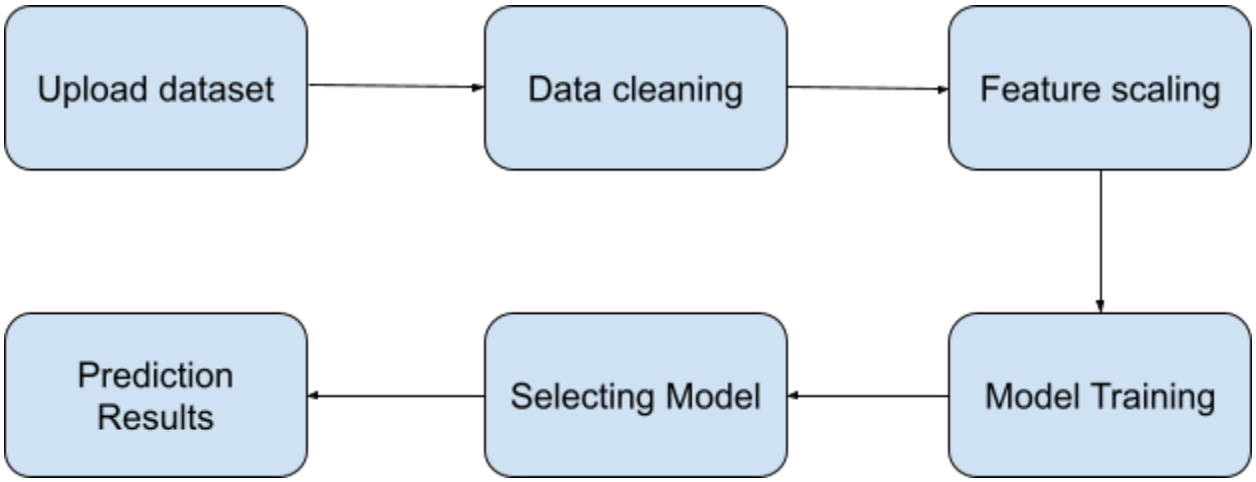
The idea of a Low Level Design Document (LLD) is to provide a complete detailed design of the inner architecture along with actual code for BigMart Sales Prediction. LLD defines the relationship between different classes and module at a modular level. I It provides a blueprint of the entire application so the programmer can use it for directly implementing the concept into an actual executable program.

1.2 Scope

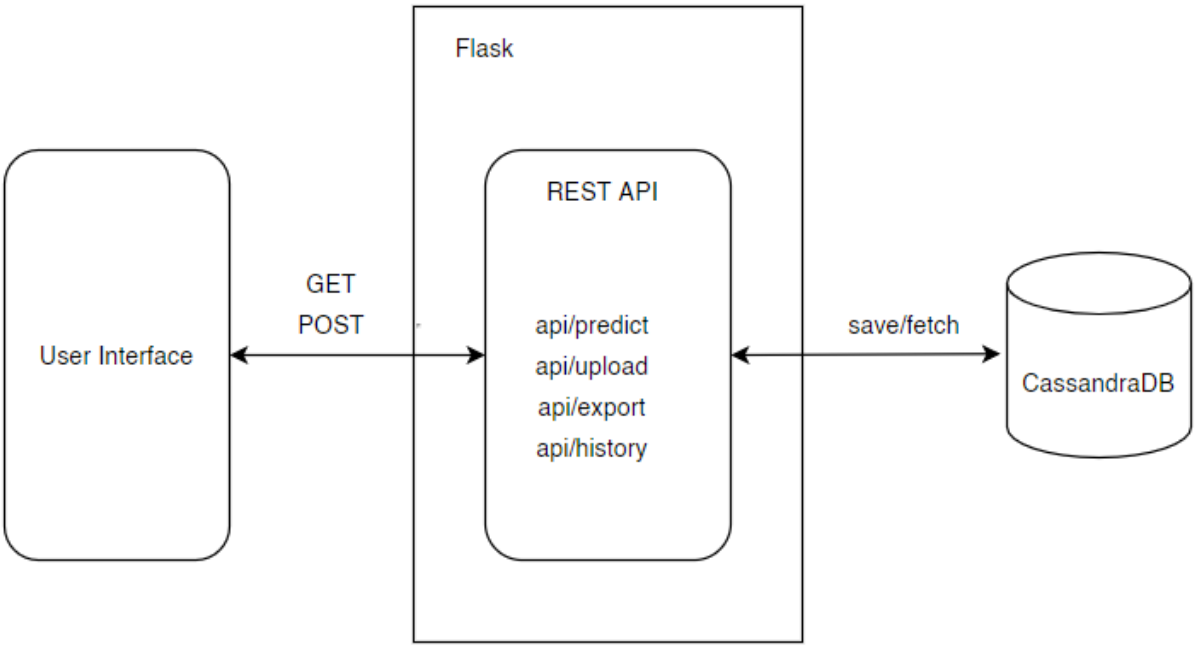
LLD documentation is a design process that follows step by step guidance to refined the development process. With the help of LLD, we can define data structures required for building the entire application.

2. Architecture

Abstract Design



Application Workflow



3. Architecture Description

3.1 Data

BigMart dataset consists of +13,000 rows, both train and test. The information comprises sales history for different stores, product details and outlets across different regions.

3.2 Data Transformation

The data is in CSV format, so no data transformation is required.

3.3 Database Sope

For storing data we are using CassandraDB, which is a NoSQL database.
Operations performed: create cluster, keyspace, table, upload data to table, fetch data from table.

3.4 Data Upload & Export

Sales details can be uploaded, which is then used for making predictions. It will be stored in the database and can be exported on user request from the portal.

3.5 Preprocessing

Data Preprocessing involves handling missing values, imbalanced data set handling, encoder categorical features, handling outliers, checking for skewness in data, cleaning junk data and standardizing dataset.

3.6 Model Building

For this particular use case we will be dealing with two machine learning algorithms, namely, Linear Regression, Random Forest and XGBoost.

3.7 User from Data

For making predictions user needs to upload/insert sales related data e.g. Item type, MRP, Weight, Visibility, Outlet type and establishment year.

3.8 Data validation

Validation steps are added to check if the data provided by the user is valid and in appropriate format.

3.9 Storing Data

After making a prediction, the user has an option to save the prediction in the database for future reference. The database chosen for this project is CassandraDB.

3.10 RESTful API

Every operation is defined as an API which can be consumed in order to access or utilize any service within the system. For every operation, there will be an API with a unique name, dedicated for that purpose only.

3.11 Deployment

We are using Flask for building the entire end to end application, which can be hosted on any cloud based platforms (AWS, Azure) or any free hosting such as Heroku.