High Level Design Document

Store Sales Prediction

Shubham Gantayat

Version 1.0

01/12/2021

Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Description | Responsible Party | Date |
| 1.0 | Initial version | Shubham Gantayat | 01-12-2021 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Abstract

The demand for a product or service keeps changing from time to time. No business can improve its financial performance without estimating customer demand and future sales of products/services accurately. Sales forecasting refers to the process of estimating demand for or sales of a particular product over a specific period of time. In this project, we will use machine learning to predict sales on a BigMart dataset.

# Introduction

## Why this is a High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding andcan be used as a reference manualfor how the modules interact at a high level.

* + 1. Present all the design aspects and define them in detail
    2. Describe the user interface being implemented
    3. Describe the hardware and software interfaces
    4. Describe the performance requirements
    5. Include design features and the architecture of the project
    6. List and describe the non-functional attributes like
       - Security
       - Reliability
       - Maintainability
       - Portability
       - Reusability
       - Application compatibility
       - Resource utilization
       - Serviceability

## Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

## Definition

|  |  |
| --- | --- |
| Term | Description |
| SSP | Store Sales Prediction Application |
| Database | Collection of all information monitored by this application |
| AWS | Amazon Web Services |
| IDE | Integrated Development Environment |

# General Description

## Product perspective

This SSP is a Web Application used to predict the store sales of BigMart for different outlets and different items.

## Problem Statement

Nowadays, shopping malls and Big Marts keep track of individual item sales data in order to forecast future client demand and adjust inventory management. In a data warehouse, these data stores hold a significant amount of consumer information and particular item details. By mining the data store from the data warehouse, more anomalies and common patterns can be discovered.

## Proposed Solution

The solution proposed here is a SSP Web Application that allows user to predict store sales based on Item and Outlet Data using machine learning.

The model is updated as per the admin so that it remains updated as per the latest store sales data.

## Further Improvements

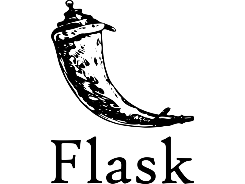
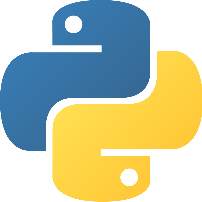
We can also use other data such as population, holidays, etc. to increase the model’s performance.

## Technical Requirements

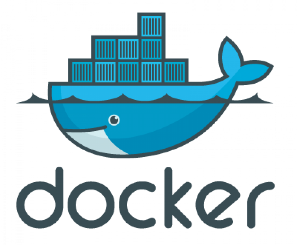
* + 1. Proper computing power to process requests and fetch data from server.

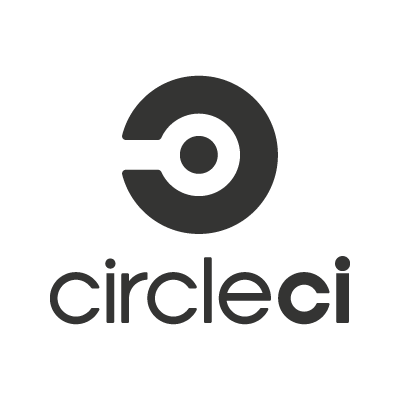
## Tools Used

* PyCharm and DataSpell is used as IDE.
* Python Flask as backend.
* HTML and CSS for frontend.
* MongoDB as database for insertion and retrieval of records.
* GitHub is used as version control system.
* XGBoost is used for model building.
* Optuna is used for hyperparameter tuning.
* Docker is used to containerize the application.
* CircleCI is used as CI/CD tool.
* AWS ECR is used to store docker containers.
* AWS S3 is used to store “Dockerrun.aws.json” file.
* AWS Elastic Beanstalk is used to deploy the container.

## Constraints

The SSP web application must be user friendly, as automated as possible and users should not be required to know any of the workings.

# Design Details

## Process Flow

We have used a HTML web application as frontend and flask as backend along with MongoDB as database. For more information on flow diagrams refer to the software architecture document.

## Event Handling

1. The System identifies at what step logging required

2. The System can log each system flow.

3. We have used database logging in MongoDB.

4. System does not hang even after using so many loggings. Logging is mandatory just because we can easily debug issues.

## Error Handling

Should errors be encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

# Performance

## Reusability

The code written, and the components used should have the ability to be reused with no problems

## Application Compatibility

The different components for this project will be using Python as an interface between them. Each component will have its own task to perform, and it is the job of the Python to ensure proper transfer of information.

## Resource Utilization

When any task is performed, it will likely use all the processing power available until that function is finished.

## Deployment



# Key Performance Indicators

* Correct prediction of sales data helps in estimating customer demand.
* Helps improve companies financial performance.
* Instant availability of products attracts customers and helps in business growth.