Store Sales Prediction

Architectural Design

Shruti Balan
18 February, 2023

Contents

1. Introduction

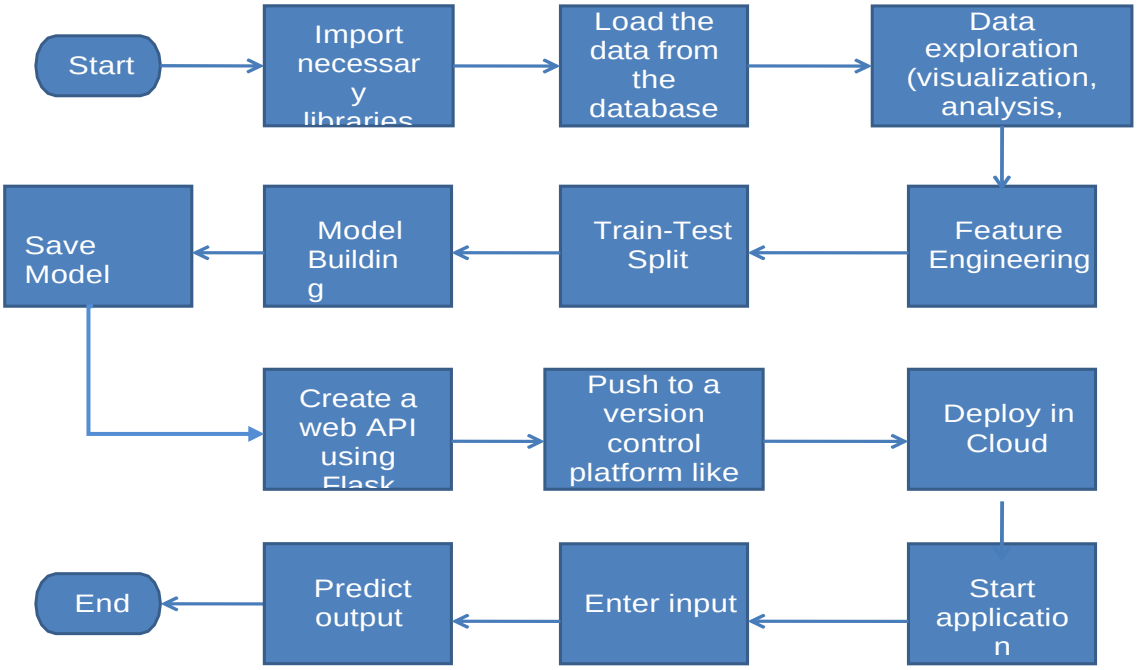
1.1. What is Architectural design document?

The purpose of this document is to provide a comprehensive architectural overview of the Sales Prediction application. It will explain the architecture and features, the modules and components of the application. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for its approval.

1.2. Scope

The architectural design is a map of the software. It allows the users to see the structure of the software at a glance.

2. Architecture



3. Architecture Description

3.1. Data Description

This dataset is taken from the UCI Machine Learning Repository (url: https://archive.ics.uci.edu/ml/datasets/South+German+Credit). It contains information on defaults, demographic factors, credit data etc. of customers. There are 21 variables:

Content There are 21 variables:
This dataset is taken from the UCI Machine Learning Repository (url: https://www.kaggle.com/datasets/brijbhushannanda1979/bigmart-sales-data).
There are 21 variables:

- **Item_Identifier :** Unique product ID.
- **Item_Weight** weight of the product (quantitative)
- **Item_Fat_Content** : Whether the fat is low fat or not (categorical)
    - **Regular**
    - **Low Fat**

- **Item_Visibility** : The % of total display area of all products in a store allocated to the particular product.
- **Item_Type** : The category to which the product belongs.
- **Item_MRP** : Maximum retail price (list price) of the product.
- **Outlet_Identifier** : Unique store ID.
- **Outlet_Establishment_Year** : The year in which the store was established.
- **Outlet_Type :** Whether the outlet is just a grocery store or some sort of supermarket.
- **Oulet_Size :** The size of the store in terms of ground area covered.
- **Outlet_Location_Type :** The type of city in which the store is located.
- **Item_Outlet_Sales :** Sales of a product in a particular store. This is the target variable.

3.2. Data Exploration

We divide the data into two types: numerical and categorical. We explore through each type one by one. Within each type, we explore, visualize and analyze each variable one by one and note down our observations.

3.3. Feature Engineering

Encoded categorical variables.

3.4. Train/Test Split

Split the data into 70% train set and 30% test set.

3.5. Model Building

Built models and trained and tested the data on the models.
Compared the performance of each model and selected the best one.
Feature importance and/or hyper-parameter tuning performed to improve the performance of the selected model.

3.6. Save the model

Saved the model by converting into a pickle file.

3.7. Create a web API

The model is used to create a web API using which the users can interact with the application. In this project, Flask has been used for the purpose.

3.8. Cloud Setup & Pushing the App to the Cloud

Selected Railway for deployment.
Used the model to develop a flask application which can predict sales for unseen data, pushed to Github using and from there deployed the application files from Github to Railway app.

3.9. Application Start and Input Data by the User

Start the application and enter the inputs.

3.10. Prediction

After the inputs are submitted the application runs the model and makes predictions. The out is displayed as a message indicating the sale price at which the product will be sold.

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 URL is accessible 2. Application is deployed	User should be able to see input fields on logging in
Verify whether user is able to edit all input fields	1. Application URL is accessible 2. Application is deployed	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	1. Application URL is accessible 2. Application is deployed	User should get Submit button to submit the inputs
Verify whether user is presented with recommended results on clicking submit	1. Application URL is accessible 2. Application is deployed	User should be presented with recommended results on clicking submit