

Stores Sales Prediction

Project Architecture

Domain: Machine Learning

Creator: Harshal S. Patil

Date: 02.06.2023

<https://www.linkedin.com/in/harshal-patil-annu/>

<https://github.com/HarshalPatil20/Sale-Market-Analysis_ML-Project>

**Architecture**

Data

Preprocessing

Cloud

Deployment on

Cloud

Desinging a

Server

Design UI On

Flask

Model

Evaluation

Hyper-

Parameter

Tuning

Exploratory Data

Analysis

Exploratory Data

Analysis

Exploratory Data

Analysis

**Deployment Process**

Render App to keep Server Running

Deploying The code on

Render

Integrating Github pipeline code with Flask

Loading Pipeline on Github

**Architecture Description**

**Data Preparation**

Data Description

The given dataset consists of two parts: a training set with 8,523 entries and a test set with 5,681 entries. The training set contains both input and output variables, while the test set only includes input variables. The goal is to predict the sales for the test data set.

The variables in the dataset are as follows:

1. Item\_Identifier: This represents a unique ID assigned to each product.

2. Item\_Weight: It denotes the weight of the product.

3. Item\_Fat\_Content: This indicates whether the product is classified as low fat or not.

4. Item\_Visibility: It represents the percentage of total display area in a store allocated to the specific product.

5. Item\_Type: This categorizes the product into different types or categories.

6. Item\_MRP: It refers to the maximum retail price or list price of the product.

7. Outlet\_Identifier: This represents a unique ID assigned to each store.

8. Outlet\_Establishment\_Year: It indicates the year in which the store was established.

9. Outlet\_Size: This specifies the size of the store in terms of ground area covered.

10. Outlet\_Location\_Type: It categorizes the type of city in which the store is located.

11. Outlet\_Type: This differentiates between a grocery store and other types of supermarkets.

12. Item\_Outlet\_Sales: This is the target variable representing the sales of the product in a particular store.

Using the given information, we can utilize various machine learning algorithms and techniques to analyze the relationship between the input variables and the target variable. By training a predictive model on the provided training data, we can make predictions on the test data set to estimate the sales for the given products.

Data Preprocessing

In data preprocessing step, we check if there missing data, duplicate values, and datatypes of each feature. Some column names are renamed, Non required column are drop.

Exploratory Data Analysis

This step includes bivariate and univariate analysis of features. Checking outliers using boxplots, and outlier treatment is carried out as well. Distribution of numerical values is plotted to see to what extent our data is skewed.

Feature Engineering

For Feature I have use this columns as feature 'Item\_Weight','Item\_Fat\_Content','Item\_Visibility','Item\_Type','Item\_MRP','Outlet\_Size','Outlet\_Location\_Type','Outlet\_Type','Years\_Established' And Item\_Outlet\_Sales this is target columns.

**Model Development**

Model implementation

After train and test splitting, pipeline containing Standard Scaler and Ordinal Encoder was fitted to several models such as AdaBoost Regressor, Gradient Boosting Regressor, RandomForest Regressor, XGB Regressor. Their R2 score were obtained. The highest score is acquired from the XGBoost Model.

Model Evaluation

Test dataset is used to evaluate the model. 20% of dataset was separated for testing. Predicted results of the model are compared with the actual data to check the amount of error.

**Deployment**

Designing UI with Flask

For this project, a user interface is built on Flask. It is a web application that helps us to create applications for projects. It is a free Python-based drag-and-drop web app builder.

Code deployment on cloud

The codes for this machine learning model should be deployed to the cloud, so that when data is entered into the application, our code runs and a user gets the result online.

**Deployment Process**

In this stage, we establish a server using Flask that runs the uplink code (server code) in parallel, where our model is executing, via an uplink. We will post the hole after execution or asynchronous execution. Git and Github are used to code in the Render cloud. Then, we'll configure a cron job to maintain the server and server code in operation indefinitely.