

# Architecture Design

## STORES SALES PREDICTION

Architecture Design

**Document Control**

Version	Date	Author	Comments
11	7/7/2023	TRUPTI SHINDE	

## Index

Content	Page No
Abstract	4
1. Introduction	4
1.1 What is Architecture Design?	4
1.2 Scope	4
1.3 Constraints	4
2. Technical Specification	5
2.1 Dataset	5
2.2 Logging	7
2.3 Deployment	7
3. Technology Stack	7
4. Proposed Solution	8
5. Architecture	8
5.1 Architecture Description	9
6. User Input/Output Workflow	10

## **Abstract**

Machine Learning is a category of algorithms that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build models and employ algorithms that can receive input data and use statistical analysis to predict an output while updating outputs as new data becomes available. These models can be applied in different areas and trained to match the expectations of management so that accurate steps can be taken to achieve the organization's target. In this paper, the case of Big Mart, a one-stop-shopping-center, has been discussed to predict the sales of different types of items and for understanding the effects of different factors on the items' sales. Taking various aspects of a dataset collected for Big Mart, and the methodology followed for building a predictive model, results with high levels of accuracy are generated, and these observations can be employed to make decisions to improve sales.

## **1. Introduction**

### **1.1 What is Architecture Design?**

The goal of Architecture Design (AD) or a low-level design document is to give the internal design of the actual program code for the `Store Sales Prediction`. AD describes the class diagrams with the methods and relation between classes and program specification. It describes the modules so that the programmer can directly code the program from the document.

### **1.2 Scope**

Architecture Design (AD) 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. And the complete workflow.

### **1.3 Constraints**

We only predict the expected casual and registered customers based on the weather condition and date information.



The data set consists of various data types from integer to floating to object as shown in Fig.

```
In [9]: # Now we will find more information about the dataset using 'info' attribute
merged_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14204 entries, 0 to 14203
Data columns (total 12 columns):
 #   column                Non-null count  Dtype
---  --
 0   Item_Identifier        14204 non-null  object
 1   Item_Weight            11765 non-null  float64
 2   Item_Fat_Content       14204 non-null  object
 3   Item_Visibility        14204 non-null  float64
 4   Item_Type              14204 non-null  object
 5   Item_MRP               14204 non-null  float64
 6   Outlet_Identifier      14204 non-null  object
 7   Outlet_Establishment_Year 14204 non-null  int64
 8   Outlet_Size            10188 non-null  object
 9   Outlet_Location_Type   14204 non-null  object
10   Outlet_Type            14204 non-null  object
11   Item_Outlet_Sales      14204 non-null  float64
dtypes: float64(4), int64(1), object(7)
memory usage: 1.3+ MB
```

In the raw data, there can be various types of underlying patterns which also gives an in-depth knowledge about the subject of interest and provides insights into the problem. But caution should be observed

with respect to data as it may contain null values, or various types of ambiguity, which also demands pre-processing of data. The dataset should therefore be explored as much as possible.

Various factors important by statistical means like mean, standard deviation, median, count of values and maximum value, etc. are shown below for numerical attributes.

```
In [10]: # let us look at the statistical information about the dataset(min, max, mean, count etc.)
merged_data.describe()

Out[10]:
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	11765.000000	14204.000000	14204.000000	14204.000000	14204.000000
mean	12.752954	0.085953	141.004977	1997.830681	1308.895489
std	4.662502	0.061469	82.088938	8.371664	1899.791423
min	4.585000	0.000000	31.290000	1985.000000	0.000000
25%	9.719000	0.027038	94.012000	1987.000000	0.000000
50%	12.600000	0.054021	142.247000	1996.000000	589.272000
75%	16.750000	0.084037	185.858000	2004.000000	2182.184200
max	21.360000	0.328361	299.868400	2009.000000	13089.964800

Preprocessing of this dataset includes doing analysis on the independent variables like checking for null values in each column and then replacing or filling them with supported appropriate data types so that analysis and model fitting is not hindered from their way to accuracy. Shown above are some of the representations obtained by using Pandas tools which tell about variable count for numerical columns and model values for categorical columns. Maximum and minimum values in numerical columns, along with their percentile values for median, play an important factor in deciding which value to be chosen at priority for further exploration tasks and analysis. Data types of different columns are used further in label processing and a one-hot encoding scheme during the model building.

## 2.2 Logging

We should be able to log every activity done by the user

- The system identifies at which step logging require.
- The system should be able to log each and every system flow.
- Developers can choose logging methods. Also, can choose database logging.
- The system should be not be hung even after using so much logging. Logging just because we can easily debug issuing so logging is mandatory to do.

## 2.3 Deployment

For the hosting of the project, we will use Heroku.



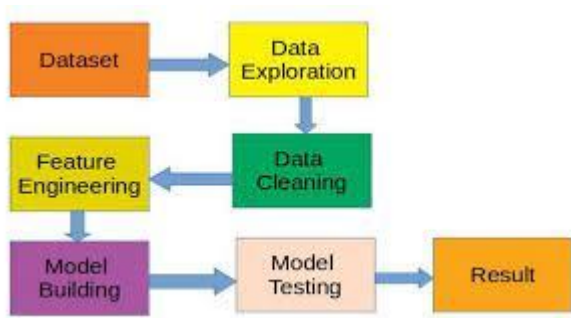
## 3. Technology Stack

<b>Front End</b>	HTML/CSS
<b>Backend</b>	Python/ Flask
<b>Deployment</b>	Heroku

#### 4. Proposed Solution

We will use performed EDA to find the important relation between different attributes and will use a machine-learning algorithm to predict the future sales demand. The client will be filled the required feature as input and will get results through the web application. The system will get features and it will be passed into the backend where the features will be validated and preprocessed and then it will be passed to a hyperparameter tuned machine learning model to predict the final outcome.

#### 5. Architecture





5.data gathering:

Data source: <https://github.com/1998Tss77j/datasets>

Train and Test data are stored in .csv format.

## 5.2 Raw Data Validation

After data is loaded, various types of validation are required before we proceed further with any operation. Validations like checking for zero standard deviation for all the columns, checking for complete missing values in any columns, etc. These are required because The attributes which contain these are of no use. It will not play role in contributing to the sales of an item from respective outlets.

Like if any attribute is having zero standard deviation, it means that's all the values are the same, its mean is zero. This indicates that either the sale is increasing or decrease that attribute will remain the same. Similarly, if any attribute is having full missing values, then there is no use in taking that attribute into an account for operation. It's unnecessary increasing the chances of dimensionality curse.

## 5.3 Data Transformation

Data transformation is required. Here, the 'Item Weight' and "Outlet Type' attributes contain the missing values. So, they are filled in both the train set as well as the test set with supported appropriate data types.

#### 5.4 New Feature Generation

We can derive new item category from item type and Outlet\_Years

#### 5.5 Data Pre-processing

In data pre-processing all the processes required before sending the data for model building are performed. New attributes were added named "Outlet years", where the given establishment year is subtracted from the current year. Then mapping of "Fat content" is done based on 'Low', 'Reg' and 'Non-edible'.

#### 5.6 Feature Engineering

After pre-processing it was found that some of the attributes are not important to the item sales for the particular outlet. So those attributes are removed. Then standard scalar is performed to scale down all the numeric features. Even one hot encoding is also performed to convert the categorical features into numerical features. After that pipeline is created for the scaling numerical features and encoding the categorical features.

#### 5.7 Model Building

After doing all kinds of pre-processing operations mention above and performing scaling and encoding, the data set is passed through a pipeline to all the models, Linear Regression, Decision tree, Random Forest, Gradient boost, Adaboosting, Support vector machine and XGBoost regressor using EvalML. It was found that Random Forest performs best with the smallest RMSE value i.e. 1016 and the highest R2 score equals 0.59. on test data So 'Random Forest' performed well in this problem.

### **5.8 Model Saving**

Model is saved using pickle library in pickle` format.

### **5.9 Flask Setup for Web Application**

After saving the model, the API building process started using Flask. Web application creation was created in Flask for testing purpose. Whatever user will enter the data and then that data will be extracted by the model to predict the prediction of sales, this is performed in this stage.

### **5.10 GitHub**

The whole project directory will be pushed into the GitHub repository.

### **5.11 Deployment**

The project was deployed from GitHub into the Heroku platform.

## **6. User Input / Output Workflow.**

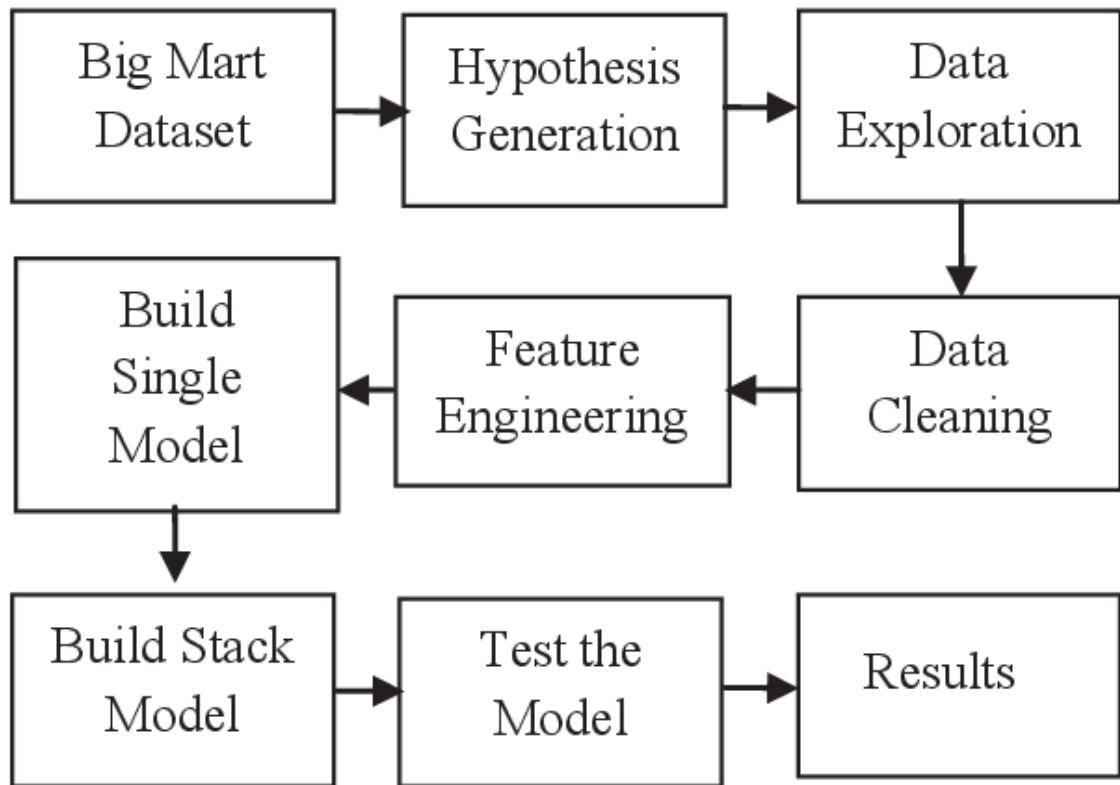


Fig. 1. Flow Diagram of the Proposed System