

Architecture Design

BIG MART SALES PREDICTION

Document Control

Change Record:

Version	Date	Author	Comments
1.0	07-02-2023	Shaik Imran Fazil	
	07-02-2023	Bandi Lokesh	
	07-02-2023	O Dinesh Kumar	
	07-02-2023	Shaik Chetansha	

Approval Status:

Version	Review date	Reviewed By	Approved By	Comments

Index

Content	Page No
Abstract	3
1. Introduction	3
1.1 What is Architecture Design?	3
1.2 Scope	3
1.3 Constraints	3
2. Technical Specification	4
2.1 Dataset	4
2.2 Logging	5
2.3 DataBase	6
2.4 Deployment	6
3. Technology Stack	6
4. Proposed Solution	6
5. Architecture Detail	7

Abstract

A class of methods known as "machine learning" enables software applications to predict outcomes more accurately without having to be explicitly coded. Building models and using algorithms that can take input data and use statistical analysis to predict an output while updating results as new data becomes available is the fundamental tenet of machine learning. These models can be used in many contexts and taught to meet management expectations so that precise actions can be done to meet the organization's goal. In order to forecast the sales of various things and comprehend the influences of various elements on the sales of the items, the instance of Big Mart, a one-stop shopping mall, has been examined in this paper. Results are produced with high degrees of accuracy using different components of a dataset gathered for Big Mart and the process used to create a prediction model. Using these observations, decisions may be made to increase sales.

1. Introduction

1.1 What is Architecture Design?

A low-level design document, or Architecture Design (AD), aims to provide the internal design of the actual computer code for the "Bike Share Prediction System." With regard to the techniques and connections between classes and programme specifications, AD describes the class diagrams. In order for the programmer to create the programme directly from the document, it describes the modules.

1.2 Scope

Architecture Design (AD) is a component-level design method that incorporates a sequential process of refinement. Data structures, necessary software, architecture, source code, and finally performance algorithms can all be designed using this method. Overall, during requirement analysis, the data organisation may be created, and then refined, during data design work. and the entire process.

1.3 Constraints

Based merely on the weather and date information, we forecast the anticipated number of casual and registered clients.

2. Technical Specification

2.1 Dataset

The data scientists at Big Mart gathered sales information from 10 stores spread across various locales, each of which sold 1559 distinct products. It is deduced what function specific attributes of an item play and how they impact sales using all the observations. The following describes the dataset:

```
data=pd.read_csv("/content/Train.csv")
data.head()
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	Tier 3	Grocery Store	732.
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.

The data collection includes a variety of data kinds, including integer, floating-point, and object, as illustrated in Fig.

```
[ ] data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   Item_Identifier        8523 non-null   object
1   Item_Weight            7060 non-null   float64
2   Item_Fat_Content       8523 non-null   object
3   Item_Visibility        8523 non-null   float64
4   Item_Type              8523 non-null   object
5   Item_MRP               8523 non-null   float64
6   Outlet_Identifier       8523 non-null   object
7   Outlet_Establishment_Year 8523 non-null   int64
8   Outlet_Size            6113 non-null   object
9   Outlet_Location_Type   8523 non-null   object
10  Outlet_Type            8523 non-null   object
11  Item_Outlet_Sales      8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

There may be a variety of underlying patterns in the raw data that can be used to get insight into the issue and an in-depth understanding of the subject of interest. However, as data may include null values, redundant values, or different sorts of ambiguity, caution should be exercised with regard to it. This necessitates pre-processing of the data. Therefore, it is important to learn as much as you can about the dataset.

The following table illustrates various statistically significant factors for numerical properties, including mean, standard deviation, median, count of values, maximum value, etc.

```
[ ] data.describe()
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	7060.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857645	0.066132	140.992782	1997.831867	2181.288914
std	4.643456	0.051598	62.275067	8.371760	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.773750	0.026989	93.826500	1987.000000	834.247400
50%	12.600000	0.053931	143.012800	1999.000000	1794.331000
75%	16.850000	0.094585	185.643700	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

In order to ensure that analysis and model fitting are accurate, preprocessing of this dataset include performing analysis on the independent variables, such as checking for null values in each column and then replacing or filling them with supported relevant data types. Some of the representations created by Pandas tools, which display model values for categorical columns and variable counts for numerical columns, are shown above. Deciding which value to prioritise for further investigation activities and analysis depends on the maximum and minimum values in numerical columns as well as their percentile values for the median. When developing a model, a one-hot encoding technique and label processing both require data types from various columns.

2.2 Logging

We ought to be able to record each action the user does.

- The system determines the step at which logging is necessary.
- The system must be able to record every single system flow.
- Developers have a choice of logging techniques. can also select database logging.
- Even after using so much logging, the system shouldn't hang. Logging is required since it makes it simple to troubleshoot issues.

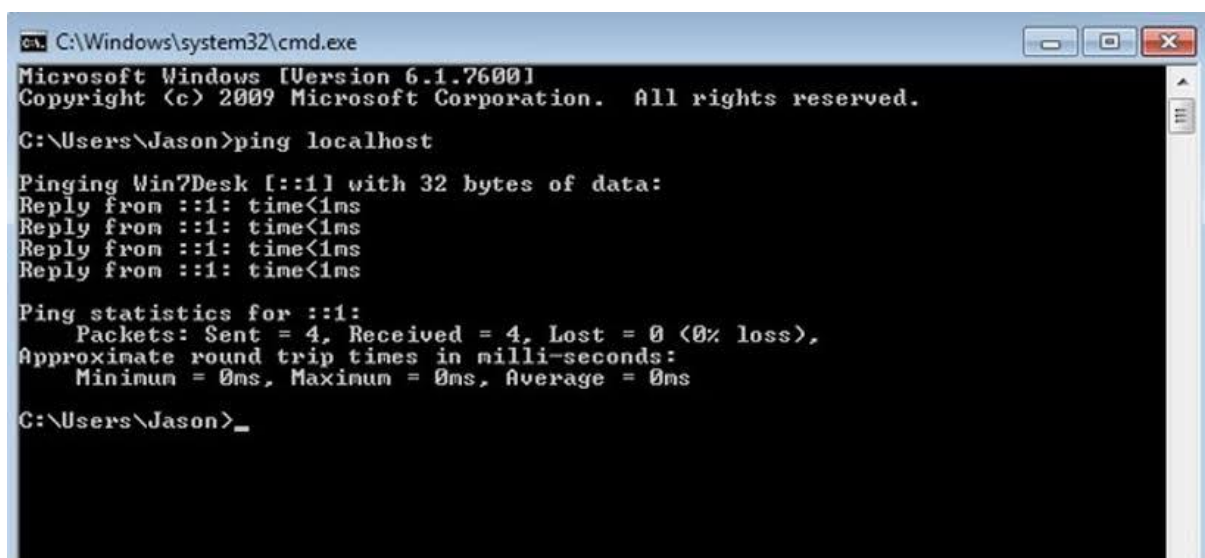
2.3 DataBase

Every request must be entered into the system's database, and it must be stored in a fashion that makes it simple to keep track of and review the data.

Every piece of information a user provided, as well as any predictions made using that information, should be recorded by the system.

2.4 Deployment

We'll make use of Localhost to host the project.

A screenshot of a Windows Command Prompt window. The title bar reads 'C:\Windows\system32\cmd.exe'. The window content shows the following text:

```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Jason>ping localhost

Pinging Win7Desk [::1] with 32 bytes of data:
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Reply from ::1: time<1ms
Reply from ::1: time<1ms

Ping statistics for ::1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\Jason>_
```

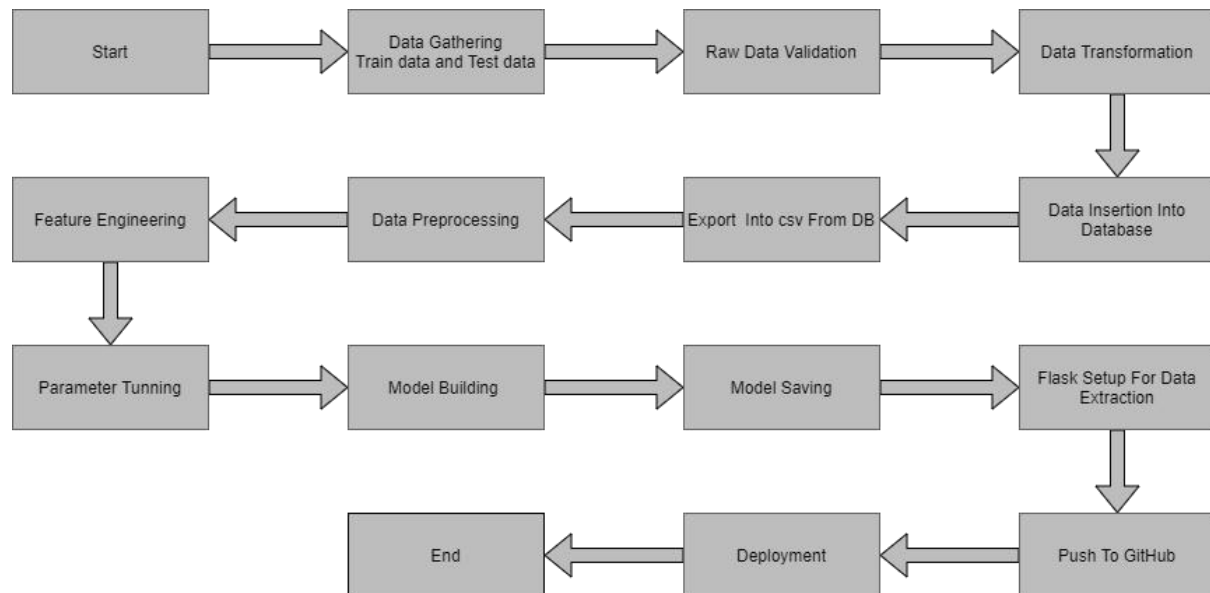
3. Technology Stack

Front End	HTML/JavaScript
Backend	Python
Deployment	Local host

4. Proposed Solution

To identify the significant relationships between various parameters, we will do EDA. To forecast future sales demand, we will utilise a machine-learning system. The client will input the necessary feature and receive results via the online application. The system will collect features, which will then be sent to the backend for validation and preprocessing before being fed to a machine learning model with hyperparameter tuning to forecast the result.

5 Architecture detail



5.1 Data Gathering

Data source: <https://www.kaggle.com/brijbhushannanda1979/bigmart-sales-data>

Train and Test data are stored in .csv format.

5.2 Raw Data Validation

Before moving on with any operation, multiple sorts of validation must be performed on the loaded data. Validations include ensuring that all of the columns have a standard deviation of zero and ensuring that no columns have any complete missing data. These are necessary because the qualities that include them are useless. It won't have any impact on how much a product sells at the relevant stores.

For example, if an attribute has zero standard deviation, all of its values are the same and the attribute's mean is zero. This suggests that regardless of whether sales are up or down, the attribute will remain the same. Similar to this, it serves no purpose to take any property into account when operating if all of its values are missing. Increasing the likelihood of the dimensionality curse is needless.

5.3 Data Transformation

Data transformation is necessary before transferring the data to the database so that it can be transformed into a format that makes database insertion simple. The two properties in

this case, "Item Weight" and "Outlet Type," have the missing data. So they are filled out with supported relevant data types in both the train set and the test set.

5.4 New Feature Generation

We can construct new mrp categories as mrp bins by deriving new item categories from item types.

5.5 Data Preprocessing

All of the steps necessary before transmitting the data for model construction are completed during data preprocessing. For instance, some of the "Item Visibility" characteristics in this instance have values of 0, which is inappropriate given that if an item is available on the market, how can its visibility be 0? In its place, the average value of item visibility for the relevant "Item Identifier" category has been used. A new characteristic called "Outlet years" was added, which subtracts the current year from the supplied establishment year. The first two characters of the Item Identifier, which represent the types of the items, are now part of a new "Item Type" attribute. Next, "Fat content" is mapped based on "Low," "Reg," and "Non-edible."

5.6 Feature Engineering

It was discovered after preprocessing that certain of the attributes are not crucial to the item sales for the specific retailer. Therefore, their qualities are dropped. To turn the categorical data into numerical features, even one hot encoding is carried out.

5.7 Parameter Tuning

Randomized searchCV is used to fine-tune the parameters. In order to solve this issue, Linear Regression and Random Forest are utilised. These two algorithms' parameters are tuned and supplied to the model.

5.8 Model Building

The data set is passed into 2 models, Linear Regression and Random Forest, after executing all of the preparation processes mentioned above, scaling, and hyperparameter tweaking. Radom forest regressor was discovered to perform at its peak with the lowest RMSE value, or 781.64, and the greatest R2 score, or 0.55. As a result, the "Random forest regressor" did well in this problem.

5.9 Model Saving

Model is saved in ".sav" format using the pickle library.

5.10 GitHub

The GitHub repository will be updated with the entire project directory.

5.11 Deployment

The project was uploaded from GitHub to the Heroku cloud platform after the cloud environment had been set up.

App link- <https://bigmartsalesprediction.herokuapp.com/>

6. User Input / Output Workflow.

