Architecture Design

STORE SALES PREDICTION

Written By	Shivansh Jayara, Subharata Nath, Anubhav Srivastav, Rohan Sinha
Document Version	1.0
Last Revised Date	05-Sep-2021

Document Control

Change Record:

Version	Date	Author	Comments

Approval Status:

Version	Review date	Reviewed By	Approved By	Comments

Index

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 `Bike Share Prediction System`. 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.

2. Technical Specification

2.1 Dataset

Big Mart's data scientists collected sales data of their 10 stores situated at different locations with each store having 1559 different products as per data collection. Using all the observations it is inferred what role certain properties of an item play and how they affect their sales. The dataset looks like as follow:

```
1 train_df.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	3735.1380
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.2700
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	Tier 3	Grocery Store	732.3800
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052

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

```
In [5]:  1 # datatype of attributes
         2 df.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
```

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 redundant 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.

```
train_df.describe()
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	8519.000000	8519.000000	8519.000000	8519.000000	8519.000000
mean	12.875420	0.069442	141.010019	1997.837892	2181.188779
std	4.646098	0.048880	62.283594	8.369105	1706.511093
min	4.555000	0.003575	31.290000	1985.000000	33.290000
25%	8.785000	0.033085	93.844900	1987.000000	834.247400
50%	12.650000	0.053925	143.047000	1999.000000	1794.331000
75%	16.850000	0.094558	185.676600	2004.000000	3100.630600
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

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 DataBase

The system needs to store every request into the database and we need to store it in such a way that it is easy to retain and look into the records.

The system should capture every data that any user gave and the prediction that has been made by that input.

2.4 Deployment

For the hosting of the project, we will use heroku

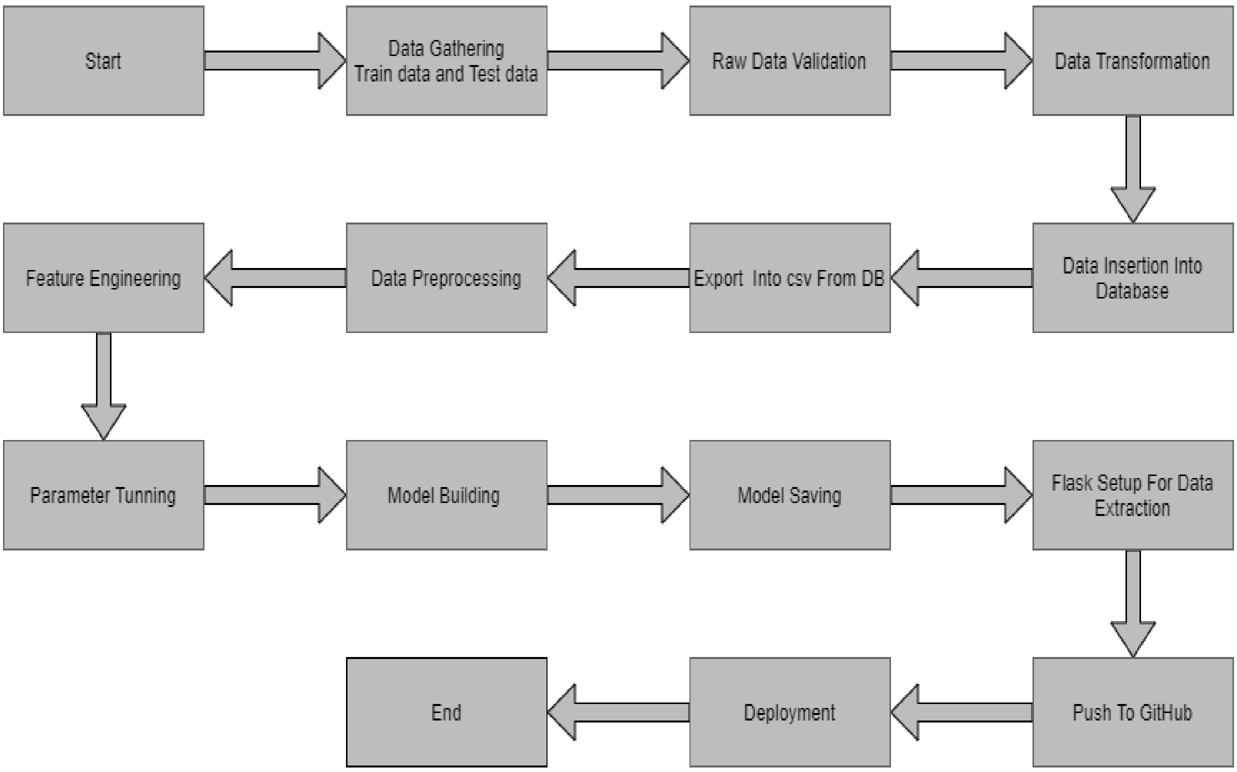
3. Technology Stack

Front End	HTML/JavaScript
Backend	Python/ Flask
Database	MongoDB
Deployment	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



5.1 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

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

Before sending the data into the database, data transformation is required so that data are converted into such form with which it can easily insert into the database. 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 Database Insertion

Both train and test data set are inserted into the database. Here MongoDB database is used to store the data set. Separate collections were created for both train and test sets.

5.5 Export as `CSV` from Database

From the database both the train and test data set are exported into the local system and stored into CSV files. Now this CSV file will have proceeded for further processing.

5.6 Data Preprocessing

In data preprocessing all the processes required before sending the data for model building are performed. Like, here the 'Item Visibility' attributes are having some values equal to 0, which is not appropriate because if an item is present in the market, then how its visibility can be 0. So, it has been replaced with the average value of the item visibility of the respective 'Item Identifier' category. New attributes were added named ''Outlet years", where the given establishment year is subtracted from the current year. A new "Item Type" attribute was added which just takes the first two characters of the Item Identifier which indicates the types of the items. Then mapping of "Fat content" is done based on 'Low', 'Reg' and 'Non-edible'.

5.7 Feature Engineering

After preprocessing it was found that some of the attributes are not important to the item sales for the particular outlet. So those attributes are removed. Even one hot encoding is also performed to convert the categorical features into numerical features.

5.8 Parameter Tuning

Parameters are tuned using Randomized searchCV. Four algorithms are used in this problem, Linear Regression, Gradient boost, Random Forest, and XGBoost regressor. The parameters of all these 4 algorithms are tunned and passed into the model.

5.9 Model Building

After doing all kinds of preprocessing operations mention above and performing scaling and hyperparameter tuning, the data set is passed into all four models, Linear Regression, Gradient boost, Random Forest, and XGBoost regressor. It was found that Gradient boost performs best with the smallest RMSE value i.e. 587.0 and the highest R2 score equals 0.55. So 'Gradient boost' performed well in this problem.

5.10 Model Saving

Model is saved using pickle library in `.pkl` format.

5.11 Flask Setup for Data Extraction

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

5.12 GitHub

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

5.13 Deployment

The cloud environment was set up and the project was deployed from GitHub into the Heroku cloud platform.

App link- **https://salespredictionapp.herokuapp.com/**

7. User Input / Output Workflow.