Architecture Design

STORES SALES PREDICTION

Document Control

Version	Date	Author	Comments
1	01.10.2021	Nikhil Patil	

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:

In [4]:
```
train_dataset
```
Out[4]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Locat
0	FDA15	9.300	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	
1	DRC01	5.920	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	
2	FDN15	17.500	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	
3	FDX07	19.200	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	
4	NCD19	8.930	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	
...	...	...	...	...	...	...	...	...	...	
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	High	
8519	FDS36	8.380	Regular	0.046982	Baking Goods	108.1570	OUT045	2002	NaN	
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	Small	
8521	FDN46	7.210	Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Medium	
8522	DRG01	14.800	Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	Small	

8523 rows × 12 columns

In [4]:
```
train_dataset
```
Out[4]:

_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.1380
Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.2700
Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	Tier 3	Grocery Store	732.3800
Low Fat	0.000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052
...	...	...	...	...	...	...	...	...	...
Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	High	Tier 3	Supermarket Type1	2778.3834
Regular	0.046982	Baking Goods	108.1570	OUT045	2002	NaN	Tier 2	Supermarket Type1	549.2850
Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	Small	Tier 2	Supermarket Type1	1193.1136
Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Medium	Tier 3	Supermarket Type2	1845.5976
Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	Small	Tier 1	Supermarket Type1	765.6700

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	12.792854	0.065953	141.004977	1997.830681	1308.865489
std	4.652502	0.051459	62.086938	8.371664	1699.791423
min	4.555000	0.000000	31.290000	1985.000000	0.000000
25%	8.710000	0.027036	94.012000	1987.000000	0.000000
50%	12.600000	0.054021	142.247000	1999.000000	559.272000
75%	16.750000	0.094037	185.855600	2004.000000	2163.184200
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 Deployment

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

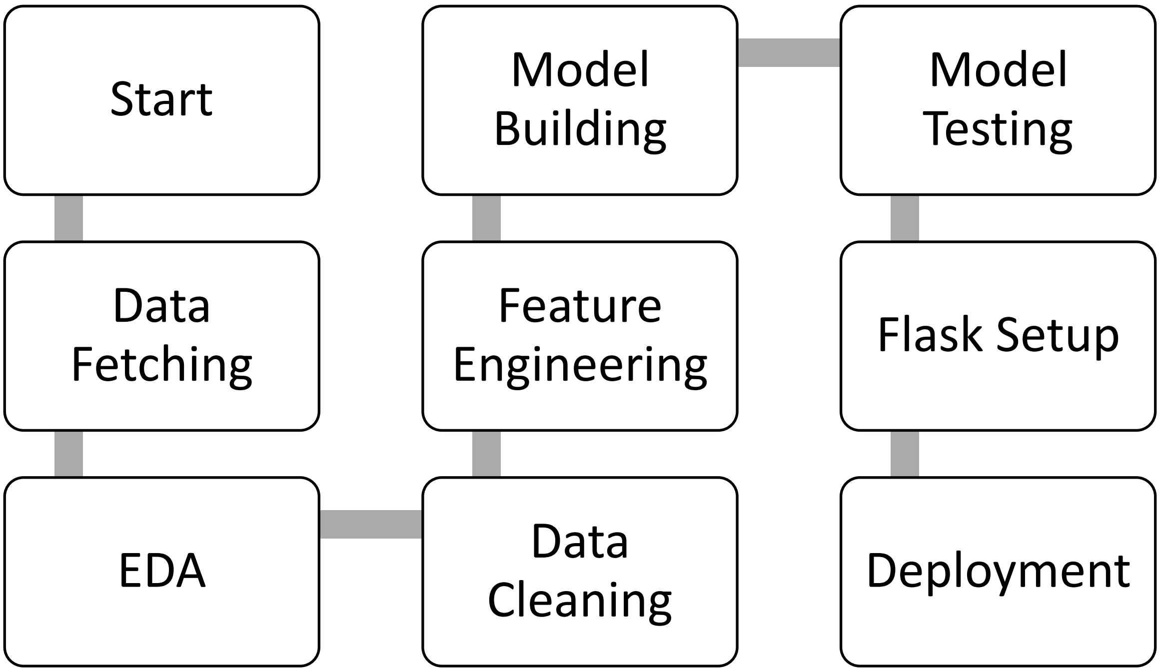
3. Technology Stack

Front End	HTML/CSS
Backend	Python/ Flask
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

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.