CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SCREENS

LIST OF ABBREVATIONS

CSV	Comma Separated Values
SRS	Software Requirement Specification
UML	Unified Modelling Language
Numpy	Numerical Python
ML	Machine Learning
XG Boost	eXtreme Gradient Boost

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 project, the case of Big Mart, a one stop-shopping centre, has been discussed to predict the sales of different types of items and for understanding the effects of different factors on the item’s 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 take decisions to improve sales

The main idea of this project is to apply Linear, Random Forest, & Ridge regression for the specific set of outlets of one of the Big Mart. Then we use the method XG Boost regressor, a gradient-based algorithm to predict sales. Finally, all the approaches are evaluated on a real-world data set obtained from the Big Mart Company. After performing the processing of all the methods cross validation is performed, sales are predicted using the trained model.

Keywords: Machine learning, Big Mart, sales prediction, Regression, XG Boost regressor.

CHAPTER-1
INTRODUCTION

Big malls & marts, in today's contemporary world, collect data connected to the sales of commodities or products with their many dependent or independent aspects as a crucial step to assist estimate future demand and inventory management. The dataset is a combination of item characteristics, customer data, and data linked to inventory management in a data warehouse, all constructed using numerous dependent and independent factors. In order to achieve more accurate forecasts and discover new and fascinating findings, the data is then tweaked and analyzed further. Random forests and simple or complex linear regression models may then be used to predict future sales.

For the purpose of establishing new hubs, Big Mart patterns are closely scrutinized by data scientists. To get the best outcomes, data scientists use a machine to anticipate the transactions of Big Mart and then test different patterns by shop and product. Forecasting market trends is essential for many businesses that largely depend on their information base. Shopping centres and stores are always looking for ways to make it easier for the store owner to pull in more customers throughout the day, so that the total volume of sales can be calculated for purposes of inventory management, logistics and transportation management.

Different Machine Learning techniques including Linear Regression, Random Forest, Ridge Regression, and XG Boost are used to gauge the number of trades in various Big Mart sectors depending on future customer demands. It is conceivable that the location of the shop is in a rural area or an urban area, thus it is likely that the conclusion of the transaction depends on these factors: the kind of business, the population in the area surrounding the store, the city in which the store is situated. The store's capacity, as well as the area's demographics and other factors, should be taken into account while analysing sales.

Sales predictions are critical at a shopping mall since every company faces considerable demand. When it comes to creating and refining a company's market strategy, the ability to make better predictions is always beneficial.

As a result, the company can predict client demand and better manage its inventory by tracking every item in its stores and Big Mart. Big Mart is a massive retail chain with locations in almost every country.

1.1 Objective of the Project:

The objective is to perform techniques for sales predictions. On the basis of a performance evaluation, a best suited predictive model is suggested for the company sale. The results are summarized in terms of accuracy of machine learning techniques taken for prediction. The main objective of the system is to analyse the future sales of a Big Mart company and to predict whether a particular sale will increase or decrease by using a different machine learning algorithm.

1.2 Machine Learning:

Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed. Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own.

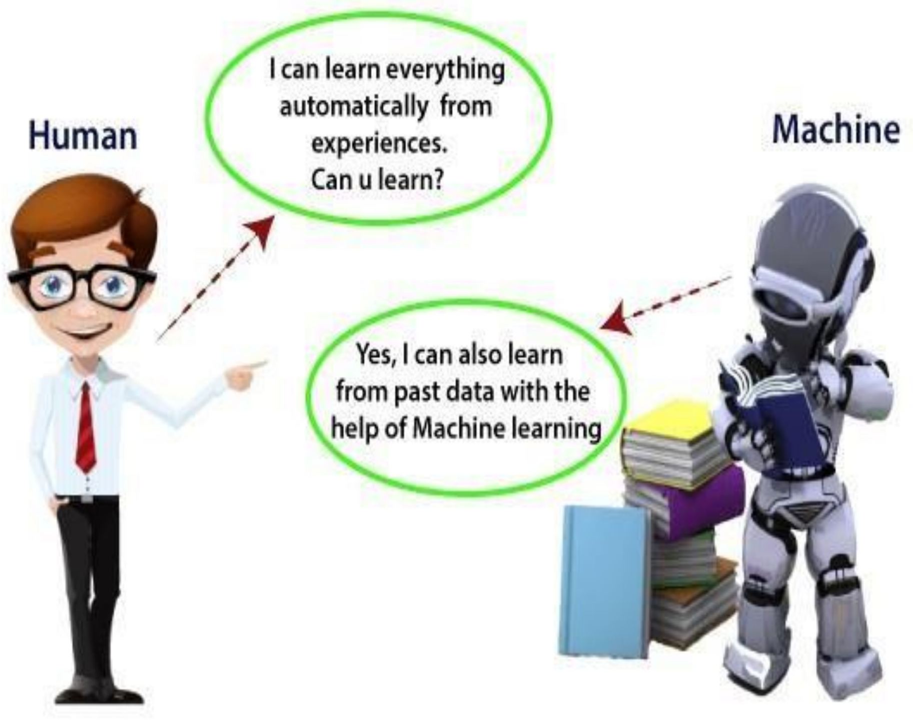

Human
I can learn everything automatically from experiences. Can u learn?
Machine
Yes, I can also learn from past data with the help of Machine learning

Fig 1.2 How Machine learns

With the help of sample historical data, which is known as training data, machine learning algorithms build a mathematical model that helps in making predictions or decisions without being explicitly programmed. A Machine Learning system learns from historical data**,** builds the prediction models**,** and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

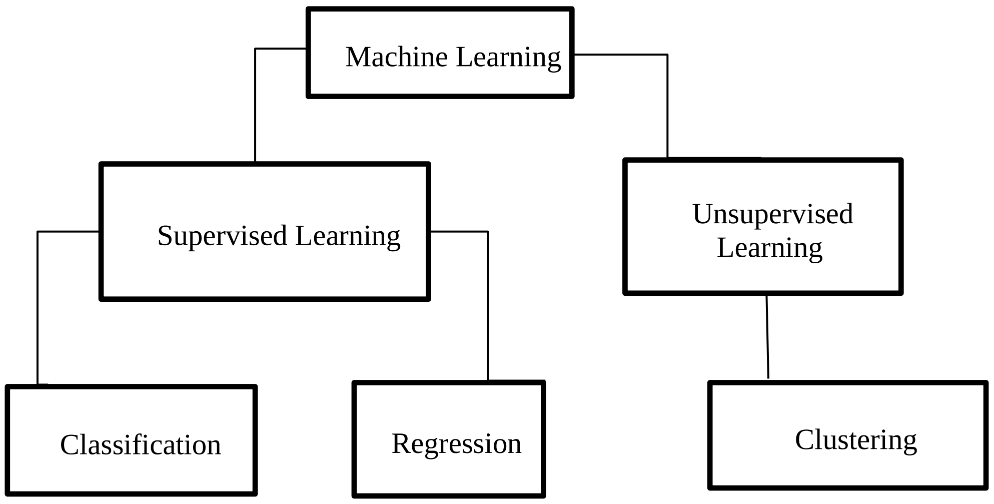


Fig 1.2.1 Types of Machine Learning

1.3 Supervised Learning:

Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y). Supervised learning is the type of machine learning in which machines are trained using well "labelled" trained data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

❖ Steps Involved in Supervised Learning:

1. First Determine the type of training dataset.
2. Collect/Gather the labelled training data.
3. Split the training dataset into training dataset, test dataset, and validation dataset.
4. Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
5. Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
6. Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
7. Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

The working of Supervised learning can be easily understood by the below example and diagram:

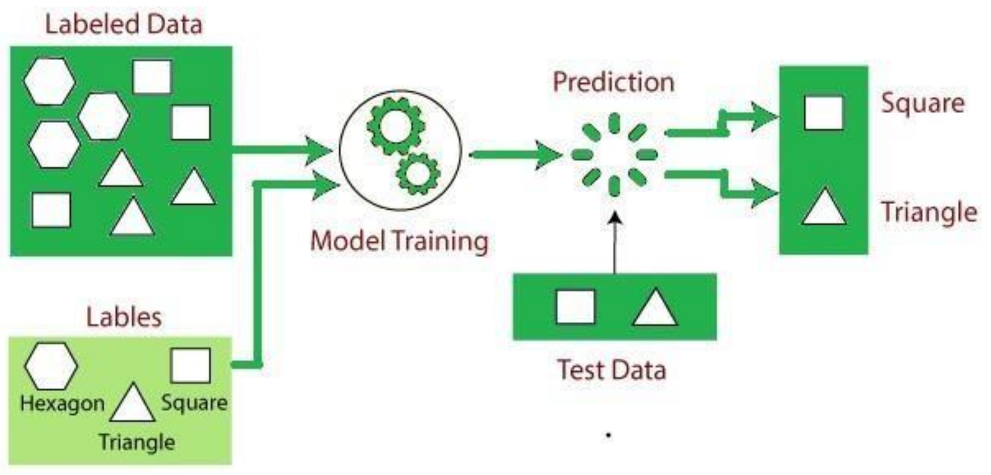


Fig 1.3 Process of any ML algorithm

Suppose we have a dataset of different types of shapes which includes square, rectangle, triangle, and Polygon. Now the first step is that we need to train the model for each shape.

- If the given shape has four sides, and all the sides are equal, then it will be labelled as a Square.
- If the given shape has three sides, then it will be labelled as a triangle.

- If the given shape has six equal sides then it will be labelled as hexagon.

Now, after training, we test our model using the test set, and the task of the model is to identify the shape. The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

- ### Regression:

Regression is a supervised learning technique which helps in finding the correlation between variables and enables us to predict the continuous output variable based on the one or more predictor variables. It is mainly used for prediction. Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as price, etc. Some real-world examples for regression are predicting the sales based on input parameters etc.

- ### Classification:

Classification is supervised learning. It can be performed on both structured and unstructured data. Classification is the process of finding a model that helps to separate the data into different categorical classes. In this process, data is categorized under different labels according to some parameters given in input and then the labels are predicted for the data.

1.3.1 Linear Regression:

Linear regression is one of the most basic types of regression in machine learning. The linear regression model consists of a predictor variable and a dependent variable related linearly to each other. In case the data involves more than one independent variable, then linear regression is called multiple linear regression models. The below-given equation is used to denote the linear regression model:

$$y=mx+c+e$$

where, m is the slope of the line, c is an intercept, and e represents the error in the model.

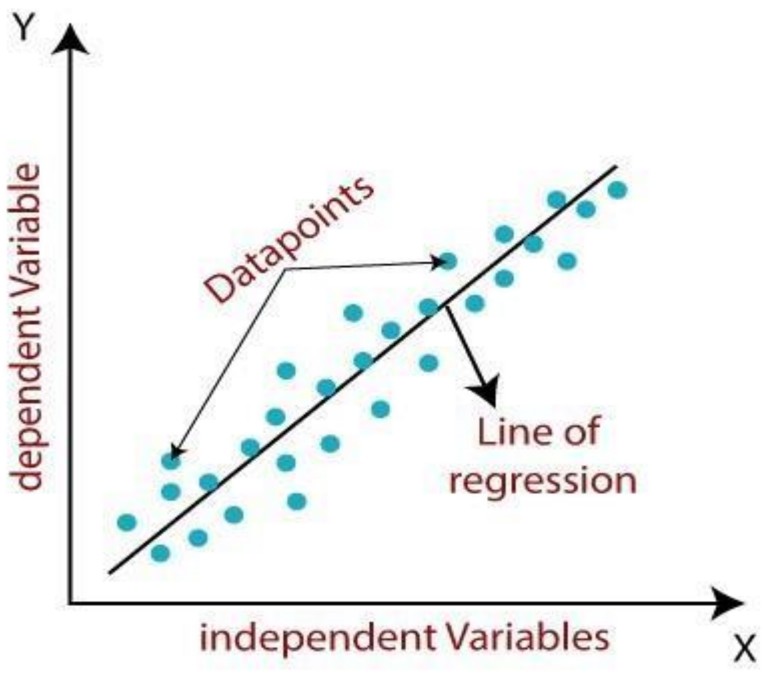


Fig 1.3.1 Linear Regression

The best fit line is determined by varying the values of m and c. The predictor error is the difference between the observed values and the predicted value. The values of m and c get selected in such a way that it gives the minimum predictor error. It is important to note that a simple linear regression model is susceptible to outliers. Therefore, it should not be used in case of big size data.

1.3.2 Random Forest Regression:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning**,** which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:

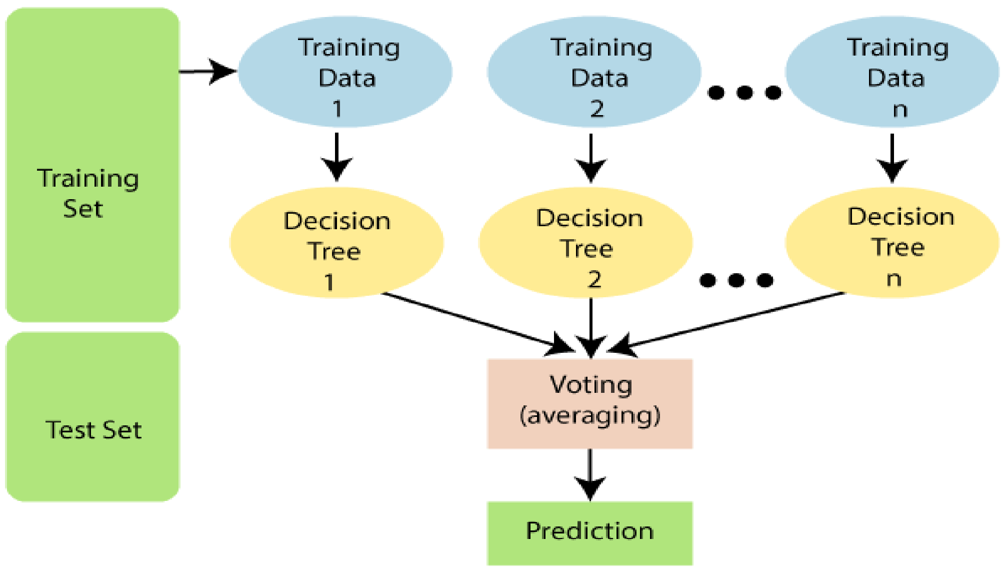


Fig 1.3.2 Random Forest Regression

1.3.3 Ridge Regression:

The Ridge Regression is a regularization technique or in simple words it is a variation of Linear Regression. This is one of the method of regularization technique which the data suffers from multicollinearity. In this multicollinearity, the least squares are unbiased and the variance is large and which deviates the predicted value from the actual value. In this equation also have an error term.

Y=mx+c+error term

Prediction errors are occurred due to bias and variance in this the multicollinearity are reduced by using lamda function. In **Ridge Regression** there is no feature selection and it shrinks the value but never reaches to zero. It is also called as L2 regularization technique.

1.3.4 XG Boost:

XG Boost stands for eXtreme Gradient Boosting. XG Boost is an implementation of gradient boosted decision trees designed for speed and performance. It uses a gradient boosting framework. XG Boost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. The implementation of the algorithm was engineered for the efficiency of computing time and memory resources. Boosting is a sequential technique which works on the principle of an ensemble. It combines a set of weak learners and improves prediction accuracy. At any instant t, the model outcomes are weighed based on the outcomes of previous instant t-1. The outcomes predicted correctly are given a lower weight and the ones misclassified are weighted higher.

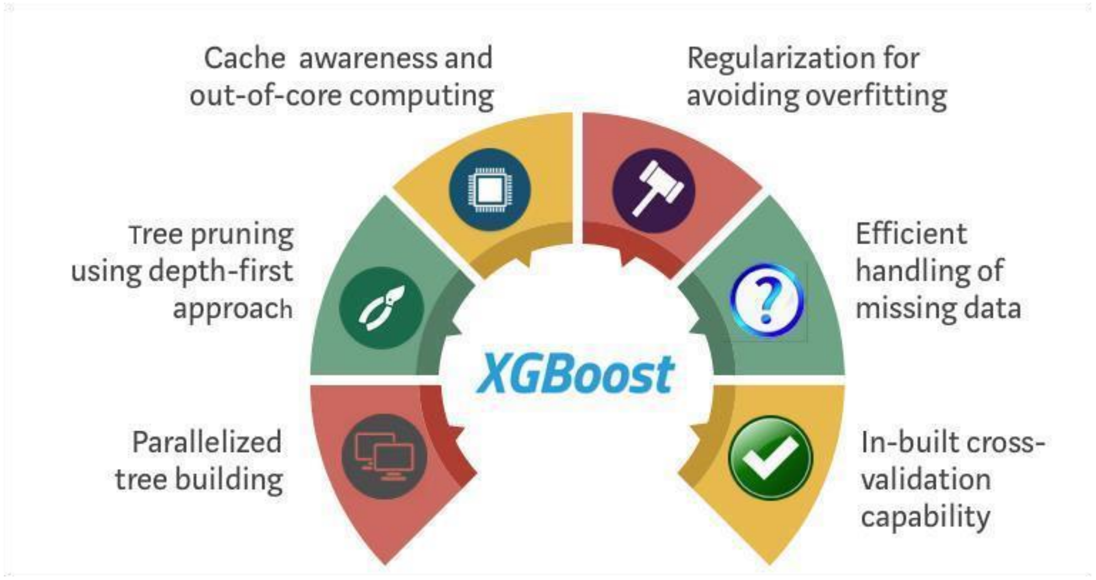

Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

Fig 1.3.4 XG Boost Regression

1.4 Unsupervised Learning:

Unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Models itself find the hidden patterns and insights from the given data. It mainly deals with the unlabelled data. Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.

1.5 Dimensionality Reduction:

Dimensionality reduction technique can be defined as, "It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information." These techniques are widely used in machine learning for obtaining a better fit predictive model while solving the classification and regression problems. It is commonly used in the fields that deal with high dimensional data. It can also be used for data visualization, etc.

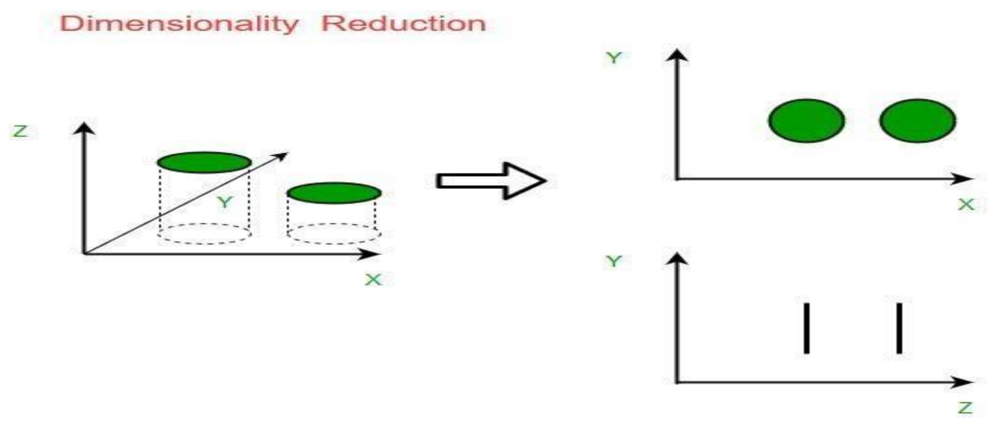


Fig 1.5 Dimensionality Reduction

1.5.1 Feature Selection:

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high accuracy. In other words, it is a way of selecting the optimal features from the input dataset.

1.5.2 Feature Extraction:

Feature extraction is the process of transforming the space containing many dimensions into space with fewer dimensions. This approach is useful when we want to keep the whole information but use fewer resources while processing the information.

1.5.3 Benefits of applying Dimensionality Reduction:

- By reducing the dimensions of the features, the space required to store the dataset also gets reduced.
- Less Computation training time is required for reduced dimensions of features.
- Reduced dimensions of features of the dataset help in visualizing the data quickly.
- It removes the redundant features (if present) by taking care of multi collinearity

CHAPTER-2
LITERATURE SURVEY

[1] Nikita Malik, Karan Singh, "Big Mart Sales Prediction", Maharaja Surajmal Institute Journal of Applied Reasearch, where the paper published, vol 3, Issue 1; January-June 2020.

Description:

The data available is increasing day by day and such a huge amount of unprocessed data is needed to be analysed precisely, as it can give very informative and finely pure gradient results as per current standard requirements. It is not wrong to say as with the evolution of Artificial Intelligence (AI) over the past two decades, Machine Learning (ML) is also on a fast pace for its evolution. ML is an important mainstay of IT sector and with that, a rather central, albeit usually hidden, part of our life. As the technology progresses, the analysis and understanding of data to give good results will also increase as the data is very useful in current aspects. In machine learning, one deals with both supervised and unsupervised types of tasks and generally a classification type problem accounts as a resource for knowledge discovery. It generates resources and employs regression to make precise predictions about future, the main emphasis being laid on making a system self efficient, to be able to do computations and analysis to generate much accurate and precise results. By using statistic and probabilistic tools, data can be converted into knowledge. The statistical inferencing uses sampling distributions as a conceptual key. ML can appear in many guises. In this paper, firstly, various applications of ML and the types of data they deal with are discussed. Next, the problem statement addressed through this work is stated in a formalized way. This is followed by explaining the methodology ensued and the prediction results observed on implementation. Various machine learning algorithms include

- Linear Regression: It can be termed as a parametric technique which is used to predict a continuous or dependent variable on basis of a provided set of independent variables. This technique is said to be parametric as different assumptions are made on basis of data set.
- Random Tree: It is an efficient algorithm for achieving scalability and is used in identification problems for building approximate system. The decisions are taken

considering the choices made on basis of possible consequences, the variables which are included, input factor. Other algorithms can include SVM, xgboost, logistic regression and so on.

[2] Meghana N, Pavan Chatradi, Avinash Chakravarthy V, Sai Mythri Kalavala,, Mrs.Neetha K S,” Improvizing Big Mart Sales Prediction”, Journal of Xi'an University of Architecture & Technology, ISSN No : 1006-7930.

Description:

The objective of this framework is to predict the future sales from given data of the previous year's using machine Learning techniques. In this paper, we have discussed how different machine learning models are built using different algorithms like Linear regression, Random forest regressor, and XG booster algorithms. These algorithms have been applied to predict the final result of sales. We have addressed in detail about how the noisy data is been removed and the algorithms used to predict the result. Based on the accuracy predicted by different models we conclude that the random forest approach and XG Booster approach are best models. Our predictions help big marts to refine their methodologies and strategies which in turn helps them to increase their profit.

[3] Gopal Behara, Neeta Nain,”A Comparative study of Big Mart Sales Prediction” Malaviya National Institute of Technology Jaipur, India

Description:

Nowadays shopping malls and Big Marts keep the track of their sales data of each and every individual items for predicting future demand of the customer and update the inventory management as well. These data stores basically contain a large number of customer data and individual item attributes in a data warehouse. Further anomalies and frequent patterns are detected by mining the data store from the data warehouse. The resultant data can be used for predicting future sales volume with the help of different machine learning techniques for the retailers like Big Mart. In this paper, we proposed a predictive model using xgboost technique for predicting the sales of a company like Big Mart and we found our model produces better performance as compare to other model. Finally a comparative analysis of existing model with other in terms performance metrics is also explained in detailed.

2.1. Proposed system:

The proposed system is used to predict the sales of the markets using the Machine learning algorithms. Model trains the previous data of a Big Mart provided using the csv file and creates a prediction model using different Machine learning algorithms like Linear regression, Ridge regression, Random forest and XG Boost Regressor.

After training all the algorithms using the data available then they are available to predict the values for the future and then the process of predicting the further values can be made and used in the real world for any kind of businesses. This will help us to find the sales and predict the amount of stock that should be available in the market.

CHAPTER-3

REQUIREMENTS

3.1 Hardware Requirements

Any Contemporary PC.

3.2 Software Requirements

Operating system	:	Windows 7 0r Windows 10
Languages Used	:	Python
Dataset	:	.CSV file
Tools	:	Jupyter notebook

3.3 Python

Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming. Python combines remarkable power with very clear syntax.

It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. Finally, Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows. In this project we are using python 3 version.

3.4 Libraries Used

Python is increasingly being used as a scientific language. Matrix and vector manipulation are extremely important for scientific computations. Both NumPy and Pandas have emerged to be essential libraries for any scientific computation, including machine learning, in python due to their intuitive syntax and high-performance matrix computation capabilities.

- ### Pip

The pip command is a tool for installing and managing Python packages, such as those found in the Python Package Index. It's a replacement for easy installation. The easiest way to install the nfl* python modules and keep them up-to-date is with a Python-based package manager called pip.

pip install (module name)

- ## NumPy

NumPy stands for ‘Numerical Python’ or ‘Numeric Python’. It is an open source module of Python which provides fast mathematical computation on arrays and matrices. Since arrays and matrices are an essential part of the Machine Learning ecosystem, NumPy along with Machine Learning modules like Scikit-learn, Pandas, Matplotlib, TensorFlow, etc. complete the Python Machine Learning Ecosystem. NumPy provides the essential multi-dimensional array-oriented computing functionalities designed for high-level mathematical functions and scientific computation. NumPy can be imported into the notebook using

import numpy as np.

- ## Pandas

Similar to NumPy, Pandas is one of the most widely used python libraries in data science. It provides high-performance, easy to use structures and data analysis tools. Pandas provides an in-memory 2d table object called Data frame. It is like a spreadsheet with column names and row labels. Hence, with 2d tables, pandas are capable of providing many additional functionalities like creating pivot tables, computing columns based on other columns and plotting graphs. Pandas can be imported into Python using below function. In this project, we use pandas for data framing i.e., to convert excel sheet to .csv file.

import pandas as pd.

- ## Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.Matplotlib comes with a wide variety of plots. Plots help to understand trends, patterns, and to make correlations. They’re typically instruments for reasoning about quantitative information. Matplotlib can be imported into Python using below function. We use Matplotlib library for graphical relationship between the attributes

import matplotlib.pyplot as plt

- ## Seaborn

Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data. Seaborn offers the functionalities like Dataset oriented API to determine the relationship between variables, Automatic estimation and plotting of linear regression plots, It supports high level abstractions for multi-plot grids and Visualizing univariate and bivariate distribution. It can be imported into Python using below function. For univariate and bivariate analysis and to different the graphs easier to non-technical people.

import seaborn as sns

- ## Sklearn

Scikit-learn is a free software machine library for Python programming language. It features various classification, regression and clustering algorithms including Linear regression, Polynomial Regression and XG Boost Regression. In our project we have used different features of sklearn library like:

from sklearn.preprocessing import LabelEncoder

For importing ML algorithms and for train-test splitting we use sklearn library. In machine learning, we usually deal with datasets which contain multiple labels in one or more than one column. These labels can be in the form of words or numbers. To make the data understandable or in human readable form, the training data is often labelled in words.

Label Encoding refers to converting the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important preprocessing step for the structured dataset in supervised learning.

Label encoding converts the data in machine readable form, but it assigns a unique number (starting from 0) to each class of data. This may lead to the generation of priority issues in training of data sets. A label with high value may be considered to have high priority than a label having lower value.

from sklearn.preprocessing import PolynomialFeatures

Polynomial features of sklearn is mainly useful for the implementation of Polynomial regression algorithm of different kind of degrees like 1,2,3,4…....This feature of sklearn help to fit the dataset with Polynomial regression algorithm.So that it helps to Predict the sales.

from sklearn.model_selection import train_test_split

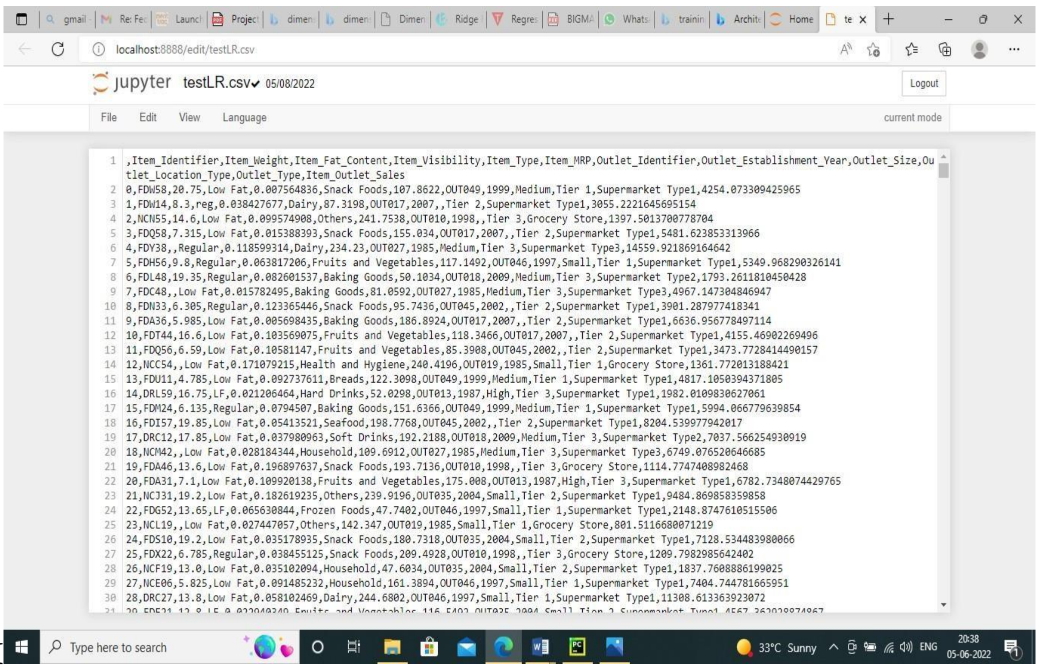
- **.CSV file:**

The dataset used in this project is a .CSV file.

In computing, a comma-separated values (.CSV) file is a delimited text file that uses a comma to separate values. A .CSV file stores tabular data (numbers and text) in plain text. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format. .CSV is a simple file format used to store tabular data, such as a spreadsheet or database. Files in the .CSV format can be imported to and exported from programs that store data in tables, such as Microsoft Excel or OpenOffice Calc. Its data fields are most often separated, or delimited, by a comma.

A .CSV is a comma-separated values file, which allows data to be saved in a tabular format. .CSVs look like a garden-variety spreadsheet but with a .csv extension. C.SV files can be used with most any spreadsheet program, such as Microsoft Excel or Google Spreadsheets.

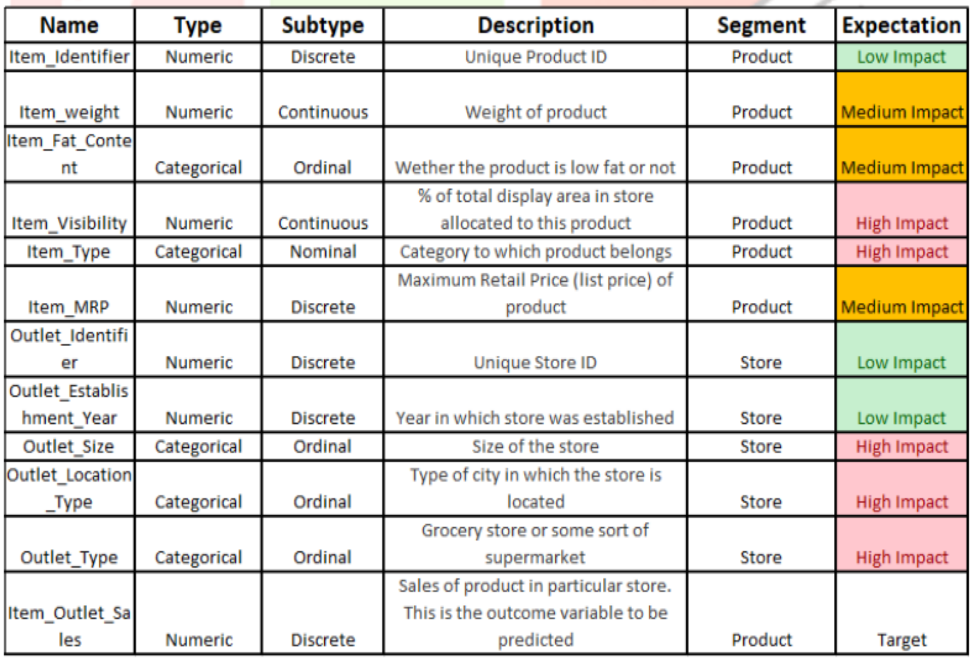
The difference between .CSV and XLS file formats is that .CSV format is a plain text format in which values are separated by commas (Comma Separated Values), while XLS file format is an Excel Sheets binary file format which holds information about all the worksheets in a file, including both content and formatting.



```
,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,FDW58,20.75,Low Fat,0.007564836,Snack Foods,107.8622,OUT049,1999,Medium,Tier 1,Supermarket Type1,4254.073309425965
1,FDW14,8.3,reg,0.038427677,Dairy,87.3198,OUT017,2007,,Tier 2,Supermarket Type1,3055.2221645695154
2,NCN55,14.6,Low Fat,0.099574908,Others,241.7538,OUT010,1998,,Tier 3,Grocery Store,1397.5013700778704
3,FDQ58,7.315,Low Fat,0.015388393,Snack Foods,155.034,OUT017,2007,,Tier 2,Supermarket Type1,5481.623853313966
4,FDY38,,Regular,0.118599314,Dairy,234.23,OUT027,1985,Medium,Tier 3,Supermarket Type3,14559.921869164642
5,FDH56,9.8,Regular,0.063817206,Fruits and Vegetables,117.1492,OUT046,1997,Small,Tier 1,Supermarket Type1,5349.968290326141
6,FDL48,19.35,Regular,0.082601537,Baking Goods,50.1034,OUT018,2009,Medium,Tier 3,Supermarket Type2,1793.2611810450428
7,FDC48,,Low Fat,0.015782495,Baking Goods,81.0592,OUT027,1985,Medium,Tier 3,Supermarket Type3,4967.147304846947
8,FDN33,6.305,Regular,0.123365446,Snack Foods,95.7436,OUT045,2002,,Tier 2,Supermarket Type1,3901.287977418341
9,FDA36,5.985,Low Fat,0.005698435,Baking Goods,186.8924,OUT017,2007,,Tier 2,Supermarket Type1,6636.956778497114
10,FDT44,16.6,Low Fat,0.103569075,Fruits and Vegetables,118.3466,OUT017,2007,,Tier 2,Supermarket Type1,4155.46902269496
11,FDQ56,6.59,Low Fat,0.10581147,Fruits and Vegetables,85.3908,OUT045,2002,,Tier 2,Supermarket Type1,3473.7728414490157
12,NCC54,,Low Fat,0.171079215,Health and Hygiene,240.4196,OUT019,1985,Small,Tier 1,Grocery Store,1361.772013188421
13,FDU11,4.785,Low Fat,0.092737611,Breads,122.3098,OUT049,1999,Medium,Tier 1,Supermarket Type1,4817.1050394371805
14,DRL59,16.75,LF,0.021206464,Hard Drinks,52.0298,OUT013,1987,High,Tier 3,Supermarket Type1,1982.0109830627061
15,FDM24,6.135,Regular,0.0794507,Baking Goods,151.6366,OUT049,1999,Medium,Tier 1,Supermarket Type1,5994.066779639854
16,FDI57,19.85,Low Fat,0.05413521,Seafood,198.7768,OUT045,2002,,Tier 2,Supermarket Type1,8204.539977942017
17,DRC12,17.85,Low Fat,0.037980963,Soft Drinks,192.2188,OUT018,2009,Medium,Tier 3,Supermarket Type2,7037.566254930919
18,NCM42,,Low Fat,0.028184344,Household,109.6912,OUT027,1985,Medium,Tier 3,Supermarket Type3,6749.076520646685
19,FDA46,13.6,Low Fat,0.196897637,Snack Foods,193.7136,OUT010,1998,,Tier 3,Grocery Store,1114.7747408982468
20,FDA31,7.1,Low Fat,0.109920138,Fruits and Vegetables,175.008,OUT013,1987,High,Tier 3,Supermarket Type1,6782.7348074429765
21,NCJ31,19.2,Low Fat,0.182619235,Others,239.9196,OUT035,2004,Small,Tier 2,Supermarket Type1,9484.869858359858
22,FDG52,13.65,LF,0.065630844,Frozen Foods,47.7402,OUT046,1997,Small,Tier 1,Supermarket Type1,2148.8747610515506
23,NCL19,,Low Fat,0.027447057,Others,142.347,OUT019,1985,Small,Tier 1,Grocery Store,801.5116680071219
24,FDS10,19.2,Low Fat,0.035178935,Snack Foods,180.7318,OUT035,2004,Small,Tier 2,Supermarket Type1,7128.534483980066
25,FDX22,6.785,Regular,0.038455125,Snack Foods,209.4928,OUT010,1998,,Tier 3,Grocery Store,1209.7982985642402
26,NCF19,13.0,Low Fat,0.035102094,Household,47.6034,OUT035,2004,Small,Tier 2,Supermarket Type1,1837.7608886199025
27,NCE06,5.825,Low Fat,0.091485232,Household,161.3894,OUT046,1997,Small,Tier 1,Supermarket Type1,7404.744781665951
28,DRC27,13.8,Low Fat,0.058102469,Dairy,244.6802,OUT046,1997,Small,Tier 1,Supermarket Type1,11308.613363923072
```

Fig 3.4 Dataset in an Excel Format

In our project we are using the dataset of Big Mart company across different stores and cities. And the dataset contains 12 attributes among first 11 are independent variables and the last one is dependent variable. The below table shows the attributes which is having high impact to improve the accuracy of the model.



Name	Type	Subtype	Description	Segment	Expectation
Item_Identifier	Numeric	Discrete	Unique Product ID	Product	Low Impact
Item_weight	Numeric	Continuous	Weight of product	Product	Medium Impact
Item_Fat_Content	Categorical	Ordinal	Wether the product is low fat or not	Product	Medium Impact
Item_Visibility	Numeric	Continuous	% of total display area in store allocated to this product	Product	High Impact
Item_Type	Categorical	Nominal	Category to which product belongs	Product	High Impact
Item_MRP	Numeric	Discrete	Maximum Retail Price (list price) of product	Product	Medium Impact
Outlet_Identifier	Numeric	Discrete	Unique Store ID	Store	Low Impact
Outlet_Establishment_Year	Numeric	Discrete	Year in which store was established	Store	Low Impact
Outlet_Size	Categorical	Ordinal	Size of the store	Store	High Impact
Outlet_Location_Type	Categorical	Ordinal	Type of city in which the store is located	Store	High Impact
Outlet_Type	Categorical	Ordinal	Grocery store or some sort of supermarket	Store	High Impact
Item_Outlet_Sales	Numeric	Discrete	Sales of product in particular store. This is the outcome variable to be predicted	Product	Target

Table 3.4.1 Attributes of the dataset

3.5 Jupyter notebook

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. .Jupyter Notebook is maintained by the people at Project Jupyter. Notebooks are a spinoff project from the IPython project, which used to have an IPython Notebook project itself.

The name, Jupyter, comes from the core supported programming languages that it supports:Julia, Python and R. Jupyter ships with the IPython kernel, which allows you to write your programs in python, but there are currently over 100 other kernels that you can also use.

IPython notebook was developed by Fernando Perez as a web based front end to IPython kernel. As an effort to make an integrated interactive computing environment for multiple languages, the Notebook project was shifted under Project Jupyter providing front end for programming environments Juila and R in addition to python.

A notebook document consists of rich text elements with HTML, formatted text, figures, mathematical equations etc. The notebook is also an executable document consisting of code blocks in python or other supporting languages.

Jupyter notebook is a client-server application. The application starts the server on a local machine and opens the notebook interface in the web browser where it can be edited and run from. The notebook is saved as an ipynb file and can be exported as html, pdf and LaTex files.

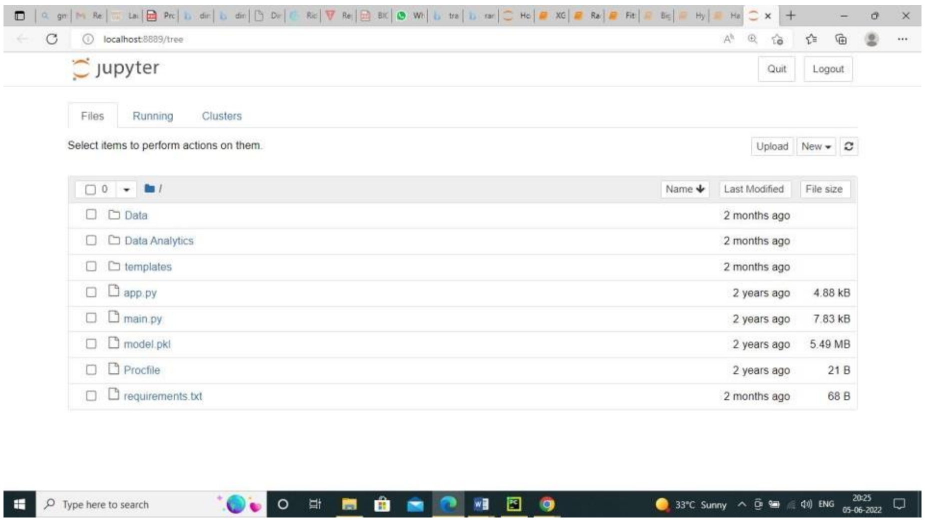
The Jupyter Notebook is not included with Python, so if you want to try it out, you will need to install Jupyter. There are many distributions of the Python language. This article will focus on just two of them for the purposes of installing Jupyter Notebook. The most popular is CPython, which is the reference version of Python that you can get from their website. It is also assumed that you are using Python.

3.5.1 Installation

If so, then you can use a handy tool that comes with Python called pip to install Jupyter Notebook like this:

```
$pip install jupyter
```

This will start up Jupyter and your default browser should start (or open a new tab) to the following URL: Home Page - Select or create a notebook



Select items to perform actions on them.

Name	Last Modified	File size
Data	2 months ago	
Data Analytics	2 months ago	
templates	2 months ago	
app.py	2 years ago	4.88 kB
main.py	2 years ago	7.83 kB
model.pkl	2 years ago	5.49 MB
Procfile	2 years ago	21 B
requirements.txt	2 months ago	68 B

Fig 3.5.1 Jupyter Notebook

3.5.2 Creating a NoteBook

Now that you know how to start a Notebook server, you should probably learn how to create an actual Notebook document. All you need to do is click on the New button (upper right), and it will open up a list of choices. On my machine, I happen to have Python 2 and Python 3 installed, so I can create a Notebook that uses either of these. For simplicity's sake, let's choose Python 3.

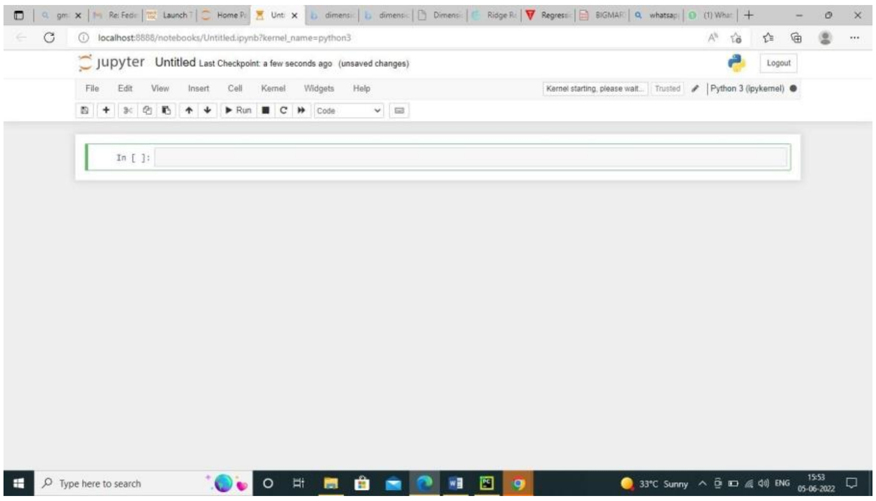
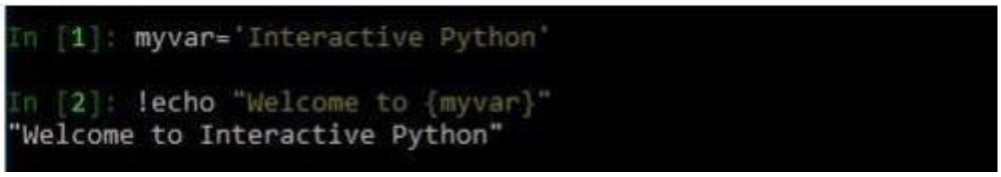

localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: a few seconds ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Kernel starting, please wait...
Trusted
Python 3 (ipykernel)
Run
Code
In [ ]:
Type here to search
33°C Sunny
ENG
15:53
05-06-2022

Fig 3.5.2 Notebook of Jupyter Notebook

3.5.3 Ipython System Commands

If the statement in the input cell starts with the exclamation symbol (!), it is treated as a system command for the underlying operating system. For example,!ls (for linux) and !dir (for windows) displays the contents of current directory.

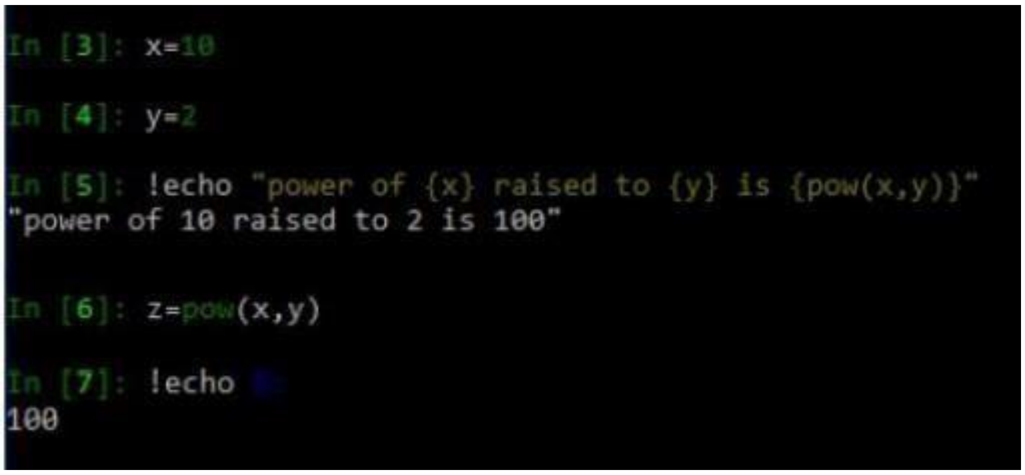
The output of system command can also be assigned to a Python variable as shown below − The variable stores output without colors and splits at newline characters. It is also possible to combine Python variables or expressions with system command calls. Variables in curly brackets {} can be embedded in command text. Observe the following example –



```
In [1]: myvar='Interactive Python'

In [2]: !echo "Welcome to {myvar}"
"Welcome to Interactive Python"
```

Fig 3.5.3 Basic program in Jupyter Notebook

Here is another example to understand that prefixing a Python variable with $ also achieves the same result.



```
In [3]: x=10

In [4]: y=2

In [5]: !echo "power of {x} raised to {y} is {pow(x,y)}"
"power of 10 raised to 2 is 100"

In [6]: z=pow(x,y)

In [7]: !echo
100
```

Fig 3.5.3.1 Program to find the power of a number

3.5.4 Project Jupyter-Overview

Project Jupyter started as a spin-off from the IPython project in 2014. IPython's language-agnostic features were moved under the name – Jupyter. The name is a reference to core programming languages supported by Jupyter which are Julia, Python and RProducts under the Jupyter project are intended to support interactive data science and scientific computing. The project Jupyter consists of various products described as under –

- IPykernel − This is a package that provides the IPython kernel to Jupyter.
- Jupyter client − This package contains the reference implementation of the Jupyter protocol. It is also a client library for starting, managing and communicating with Jupyter kernels.
- Jupyter notebook − This was earlier known as IPython notebook. This is a web based interface to IPython kernel and kernels of many other programming languages.
- Jupyter kernels − Kernel is the execution environment of a programming language for Jupyter products.
- Qtconsole − A rich Qt-based console for working with Jupyter kernels.
- nbconvert − Converts Jupyter notebook files in other formats.
- JupyterLab − Web based integrated interface for notebooks, editors, consoles etc.
- nbviewer − HTML viewer for notebook file.

CHAPTER-4
METHODOLOGY

4.1 Steps involved in Design:

- ➢ Data Collection
- ➢ Data Preprocessing
- ➢ Model Training
- ➢ Model Evaluation

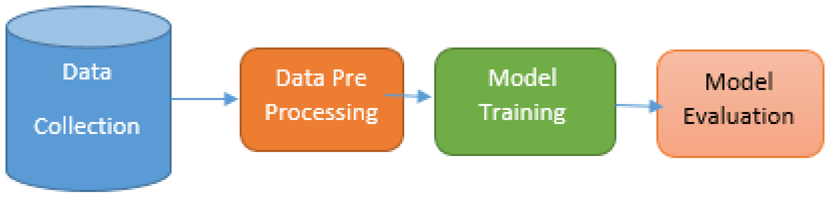


Fig 4.1 Steps involved in Design

4.1.1 Data Collection

- Data is an important asset for developing any kind of Machine learning model. Data collection is the process of gathering and measuring information from different kinds of sources.
- This is an initial step that has to be performed to carry out an Machine learning project. In the present internet world these datasets are available in different websites (Ex: Kaggle, Google public datasets, Data.gov etc).
- The dataset used in our project is downloaded from the Kaggle website and it contains nearly 12000 records and 12 different attributes. (Item_identifier, MRP, Weight, Item scale, Outlet sales etc).
- The dataset consists of 11 independent attributes and one dependent attribute (outlet sales).
- So, the aim of the project is to predict the dependent variables using independent variables.

4.1.2 Data Preprocessing

- Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.
- When creating a machine learning project, it is not always the case that we come across clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So for this, we use data preprocessing tasks. Preprocessing of the data consists of different kinds of steps in which analysis of the data, Data cleaning, Data encoding are part of this.

4.1.2.1 Explanatory Data Analysis

- Exploratory data analysis is an approach of analyzing datasets to summarize their main characteristics, often using statistical graphics and other data visualization methods. The main purpose of EDA is to help look at data before making any assumptions.
- It can help identify obvious errors, as well as better understanding patterns within the data, detect outliers or anomalous events, and find interesting relations among the variables. Specific statistical functions and techniques you can perform with EDA tools include:
- Clustering and dimension reduction techniques, which help create graphical display of high dimensional data containing many variables.
- Univariate visualization of each field in the raw dataset, with summary statistics.
- Bivariate visualizations and summary statistics that allows you to assess the relationship between each variable in the dataset and the target variable in the dataset and the target variable you’re looking at.
- Multivariate visualizations, for mapping and understanding interactions between different fields in the data.
- Predictive models, such as linear regression, statistics and data to predict outcomes. This data analysis is of two types:

a. Univariate analysis
b. Bivariate analysis

Univariate analysis is the simplest form of data analysis where the data being analyzed contains only one variable. Since it's a single variable it doesn't deal with causes or relationships.

Bivariate data is data that involves two different variables whose values can change. Bivariate data deals with relationships between these two variables.

4.1.2.2 Filling Missing Data & Data Encoding

- The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.
- By calculating the mean and Mode: In this way, we will calculate the mean or Mode of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc.
- **Data encoding:** Since the machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers.
- Our dataset also consists of different categorical data in which they are encoded in this step.

4.1.3 Training Model:

In this step the model is trained using the algorithms that are suitable. Sales prediction is a kind of problem in which One variable has to be determined using some independent variables. Regression model is suitable for this kind of scenario.

A training model is a dataset that is used to train an ML algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output. The training model is used to run the input data through the algorithm to correlate the processed output against the sample output.

Our project implements four algorithms. Linear regression, Ridge regression, Random forest regression ,XG Boost Regression where Linear regression is a normal regression algorithm and XG Boost is an Gradient descent algorithm.

4.1.3.1 Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning. Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output).To determine the response variable in our dataset called Item_Outlet_Sales by using independent variables in the dataset.

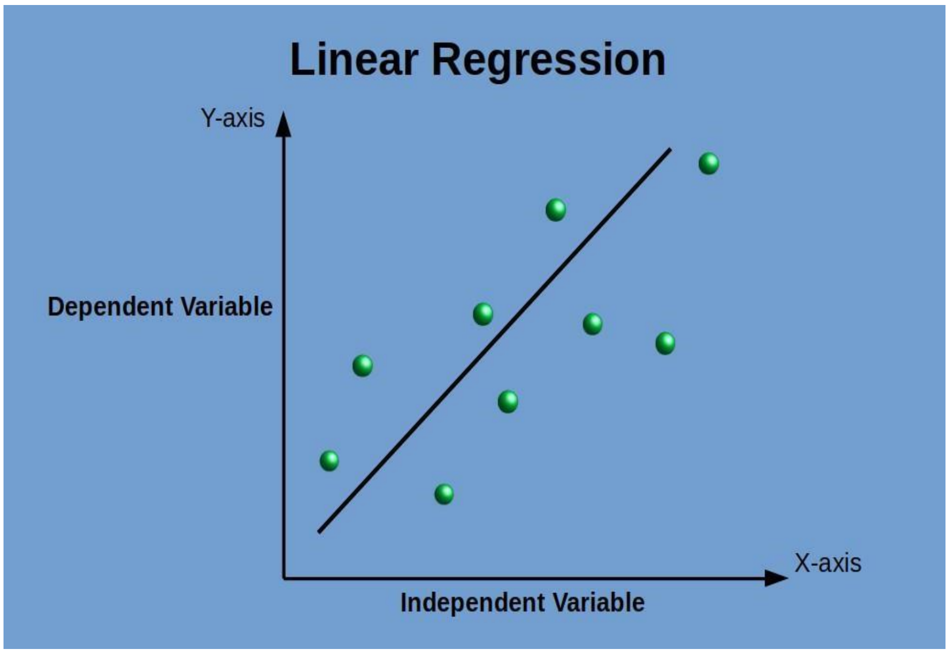


Fig 4.1.3.1 Linear Regression

4.1.3.2 Ridge Regression

Ridge Regression is a method used where multi collinearity (independent variables are highly correlated) affects outcomes. While the least square estimates (OLS) are objective in multi collinearity, their variances are broad and deviate from the true value. By applying a degree of bias to regression calculations, ridge regression eliminates standard errors. The Linear Regression Loss function is increased in Ridge Regression so as not only to minimize the number of square residuals but also to penalize the estimates of the parameters.

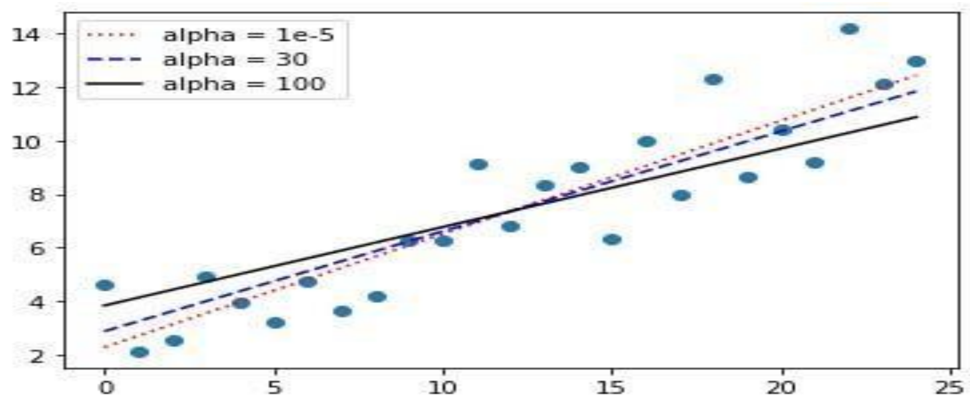


Fig 4.1.3.2 Ridge Regression

4.1.3.3 Random Forest Regression

Random Forest is a tree-based bootstrapping algorithm that combines a certain number of weak learners (decision trees) to construct a powerful model of prediction. For each person learner, a random set of rows and a few randomly selected variables are used to create a decision tree model. Final prediction may be a function of all the predictions made by the individual learners. In the event of a regression problem, the final prediction may be the mean for all predictions.

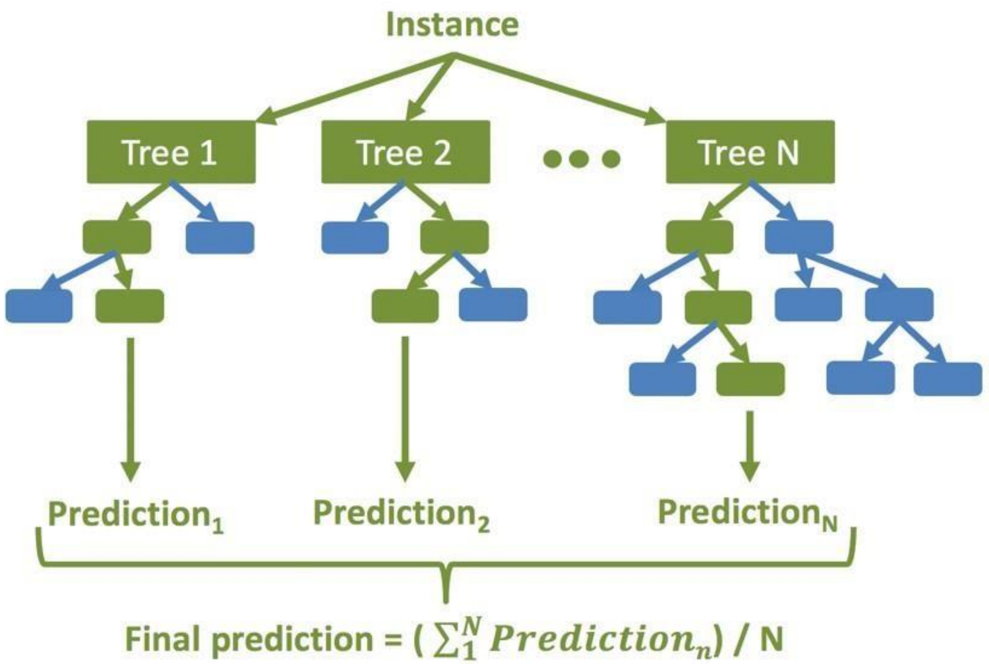


Fig 4.1.3.3 Random Forest Regression

4.1.3.4 XG Boost Regression:

XG Boost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.). A wide range of applications: Can be used to solve regression, classification, ranking, and user-defined prediction problems.

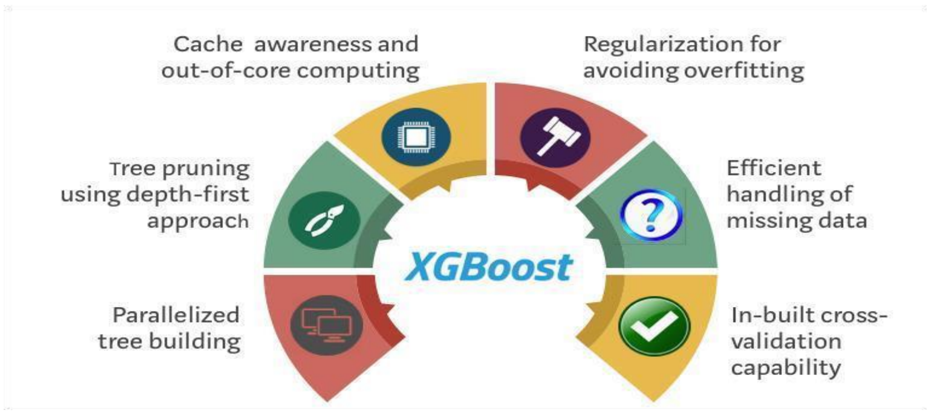

Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

Fig 4.1.3.4 XG Boost Regression

4.1.4 Model Evaluation

In this step the trained model is evaluated by determining the accuracy of the model against the test data. various ways to check the performance of our machine learning or deep learning model and why to use one in place of the other. We will discuss terms like:

1. Accuracy
2. Precision
3. Recall
4. Specificity
5. F1 score
6. Precision-Recall or PR curve
7. ROC (Receiver Operating Characteristics) curve
8. PR vs ROC curve.

Out of these we used Accuracy for evaluating our model. Accuracy is the most commonly used metric to judge a model. Accuracy is the measurement used to determine which model is best at identifying relationships and patterns between variables in a dataset based on the input, or training data.

CHAPTER-5
DESIGN

5.1. UML Introduction:

A UML diagram is a partial graphical representation (view) of a model of a system under design, implementation, or already in existence. UML diagram contains graphical elements (symbols) - UML nodes connected with edges (also known as paths or flows) - that represent elements in the UML model of the designed system. The UML model of the system might also contain other documentation such as use cases written as templated texts. The kind of the diagram is defined by the primary graphical symbols shown on the diagram. For example, a diagram where the primary symbols in the contents area are classes is class diagram.

A diagram which shows use cases and actors is use case diagram. A sequence diagram shows sequence of message exchanges between lifelines. UML specification does not preclude mixing of different kinds of diagrams, e.g. to combine structural and behavioural elements to show a state machine nested inside a use case. Consequently, the boundaries between the various kinds of diagrams are not strictly enforced. At the same time, some UML Tools do restrict set of available graphical elements which could be used when working on specific type of diagram.

UML specification defines two major kinds of UML diagram: structure diagrams and behaviour diagrams. Structure diagrams show the static structure of the system and its parts on different abstraction and implementation levels and how they are related to each other. The elements in a structure diagram represent the meaningful concepts of a system, and may include abstract, real world and implementation concepts. Behaviour diagrams show the dynamic behaviour of the objects in a system, which can be described as a series of changes to the system over time.

5.1.1 Usage of UML in Project

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time to the market. These techniques include component technology visual programming, patterns and frameworks. Additionally, the development for the World Wide Web, while making some things simpler, has exacerbated these architectural problems. The UML was designed to respond to these needs. Simply, systems design refers to the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements which can be done easily through UML diagrams.

5.2 UML Diagrams:

5.2.1 Use Case Diagram:

In the Unified Modelling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent: Scenarios in which your system or application interacts with people, organizations, or external systems. Goals that your system or application helps those entities (known as actors) achieve.

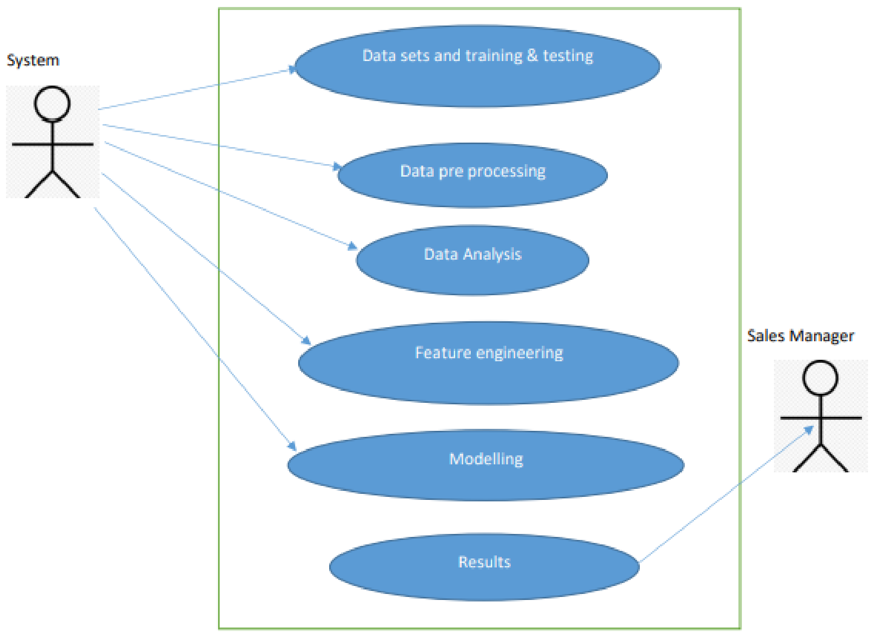


Fig 5.2.1 Use Case Diagram

5.2.2 Sequence Diagram:

A sequence diagram is a type of interaction diagram because it describes how and in what order a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

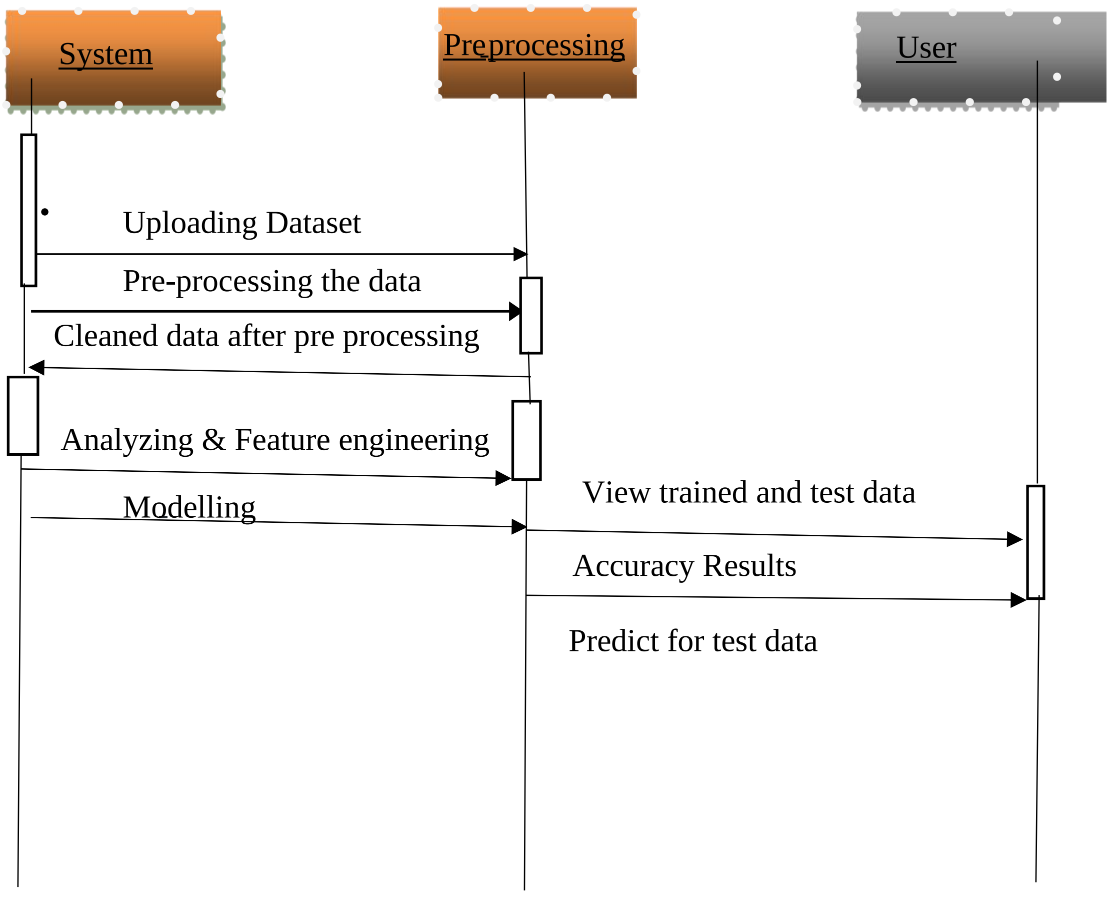


Fig 5.2.2 Sequence Diagram

5.3 Architecture of project:

An architecture is a way of representing the flow of data of a process or a system (usually an information system). This also provides information about the outputs and inputs of each entity and the process itself. Machine learning architecture defines the various layers involved in the machine learning cycle and involves the major steps being carried out in the transformation of raw data into training data sets capable for enabling the decision making of a system.

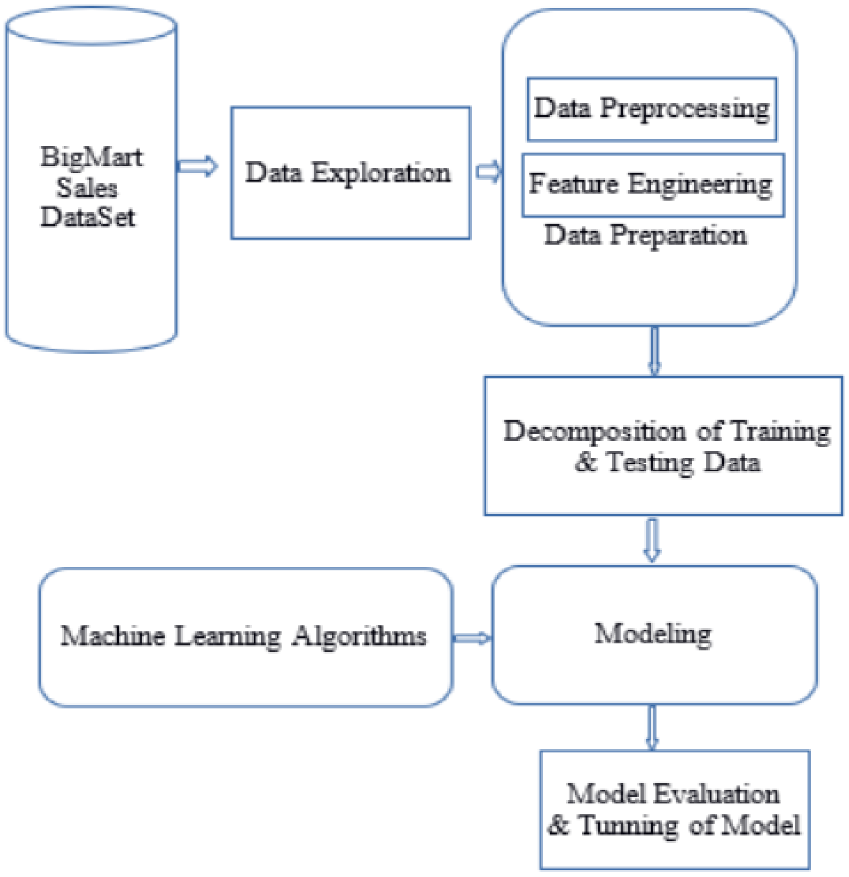

BigMart Sales DataSet
Data Exploration
Data Preprocessing
Feature Engineering
Data Preparation
Decomposition of Training & Testing Data
Machine Learning Algorithms
Modeling
Model Evaluation & Tunning of Model

Fig 5.3 Architecture of Big Mart sales prediction

The above architecture describes how sales prediction can be done. Initially obtain the datasets from kaggle website. After obtaining the datasets, perform data transformation to it in such a way that there shouldn't be any integration problem or any redundancy issue.

Now, apply feature selection techniques to the big mart sales dataset which has 12 features in it. Then a subset of features which are most important in the prediction of future sales. choose the algorithm that gives the best possible accuracy with the subset of features obtained after feature selection. Applying the algorithms to the dataset actually means that it needs to train the model with the algorithms and test the data so that the model will be fit.

And we perform model evaluation and tuning on the model to improve the accuracy of the model by using error term called "Regularization Parameter". By doing the model tuning, we get the higher accuracy.

CHAPTER-6
IMPLEMENTATION

Implementation part is made using CSV file containing 12 different attributes with nearly 12000 records. Sales are predicted using the data collected with Machine Learning algorithms like Linear regression, Ridge regression, Random Forest and XG Boost Regressor. All these algorithms helps to predict the sales. Sales are predicted by implementing all three algorithms separately and are compared one with another.

6.1 Train-Test split

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced.

- Train Dataset: Used to fit the machine learning model.
- Test Dataset: Used to evaluate the fit machine learning mode

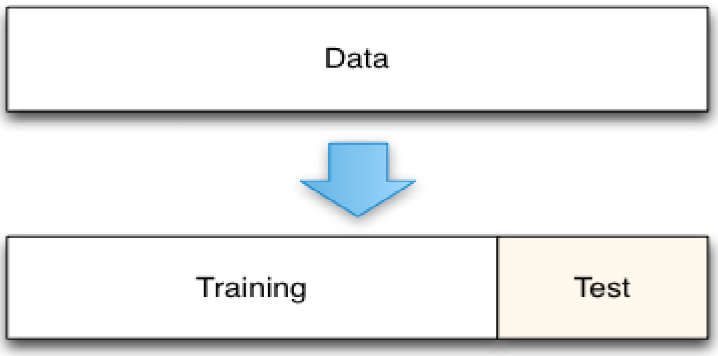


Fig 6.1 Train_Test_Split

XG Boost is an open source library providing a high-performance implementation of gradient boosted decision trees. An underlying C++ codebase combined with a Python interface sitting on top makes for an extremely powerful yet easy to implement package. It's become the go-to library for winning many Kaggle competitions. Its gradient boosting implementation is second to none and there's only more to come as the library continues to garner praise.

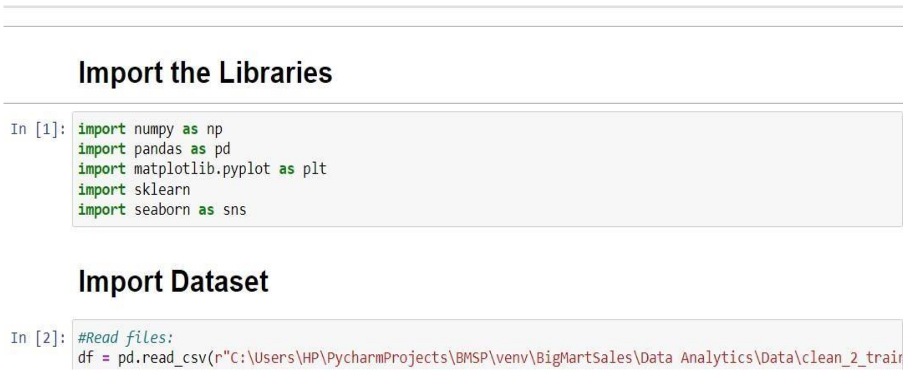
Boosting on the other hand, takes a more iterative approach. It's still technically an ensemble technique in that many models are combined together to perform the final one, but takes a more clever approach.

6.2 Implementation:

6.2.1 Importing all the required Modules and Libraries:

All the required libraries like Sklearn and Modules like Numpy, Pandas, Matplotlib, Seaborn are imported into the Jupyter notebook initially into the file created in the notebook.

After importing all the modules and libraries into the notebook, A csv file has to be loaded using Pandas into the notebook. The implementation of these will be as follows:



Import the Libraries

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import sklearn
        import seaborn as sns
```

Import Dataset

```
In [2]: #Read files:
        df = pd.read_csv(r"C:\Users\HP\PycharmProjects\BMSP\venv\BigMartSales\Data Analytics\Data\clean_2_trair
```

Fig 6.2.1 Importing the libraries

6.2.2 Data Preprocessing

- Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.
- When creating a machine learning project, it is not always the case that we come across clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So for this, we use data preprocessing tasks. Preprocessing of the data consists of different kinds of steps in which analysis of the data, Data cleaning

6.2.3 Data Visualization

As it contains large amounts of data it is not possible to analyze with the human eye normally so the feature of Data Visualization helps to analyze the entire data. The relation between any two features can be only analyzed with the Data Visualization technique.

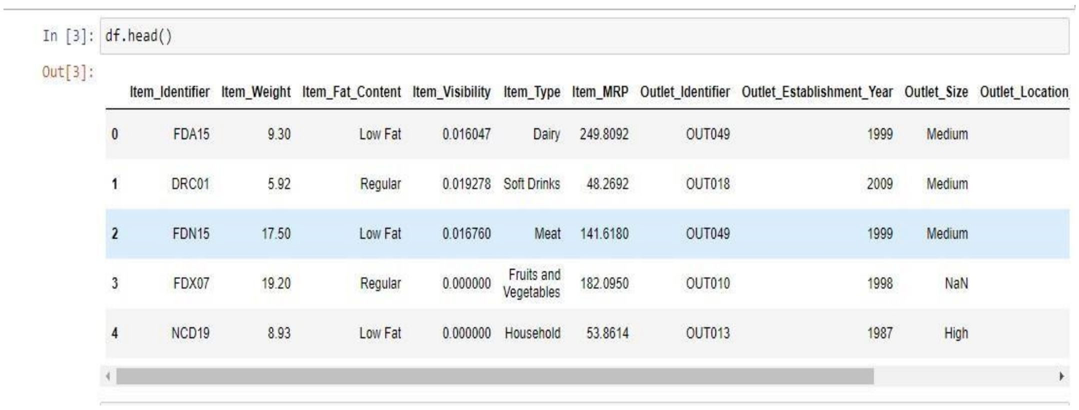
This Visualization can be made in different forms by representing the data in the pictorial forms like graph, bar chart and many other forms.

Some Visualizations are made for the dataset that is collected for the Prediction of sales. They are as follows:

6.2.4 Descriptive Analyzation

It is important to know about the information of each and every attribute, such information can be easily predicted and is analyzed as follows:

Pandas head() method is used to return top n (5 by default) rows of a data frame or series.



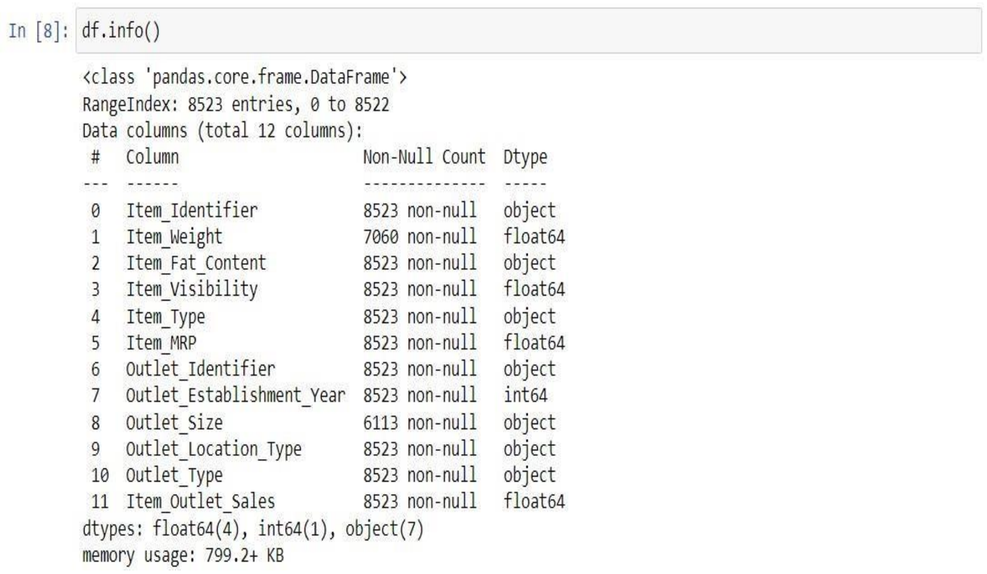
```
In [3]: df.head()
Out[3]:
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	

Fig 6.2.4 Using head() function

The info() function is used to print a concise summary of a DataFrame. This method prints information about a DataFrame including the index dtype and column dtypes, non-null values and memory usage.

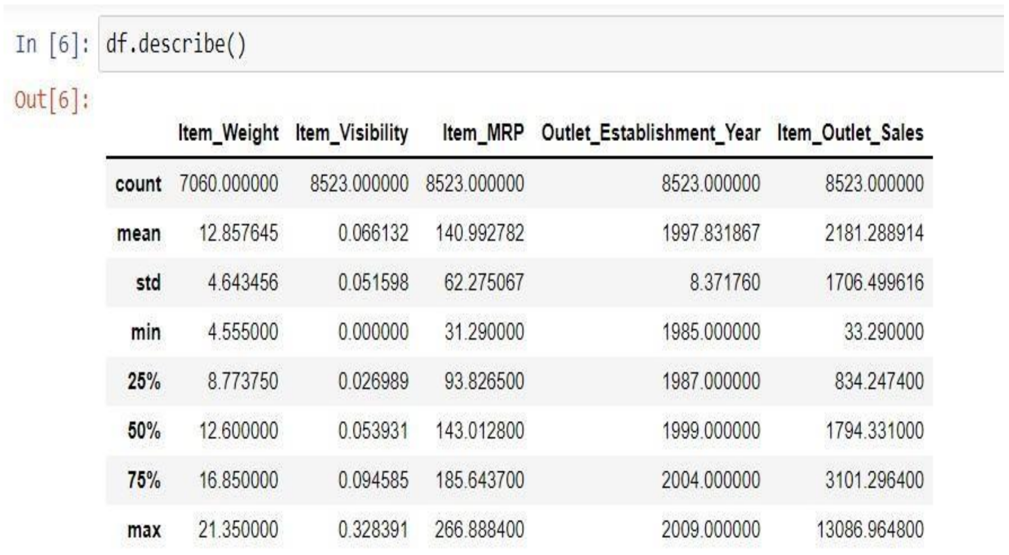
This method helps to provide a very small and important summary of the entire dataset so that it is easy to have an idea on the entire dataset that is present. It provides information about Count of the attribute, Type of the attribute along with the attribute name etc.



```
In [8]: 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
```

Fig 6.2.4.1 Displaying the datatype of the attributes

Pandas describe() is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values. When this method is applied to any kind of dataset the output will be as follows:



```
In [6]: df.describe()
Out[6]:
```

	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

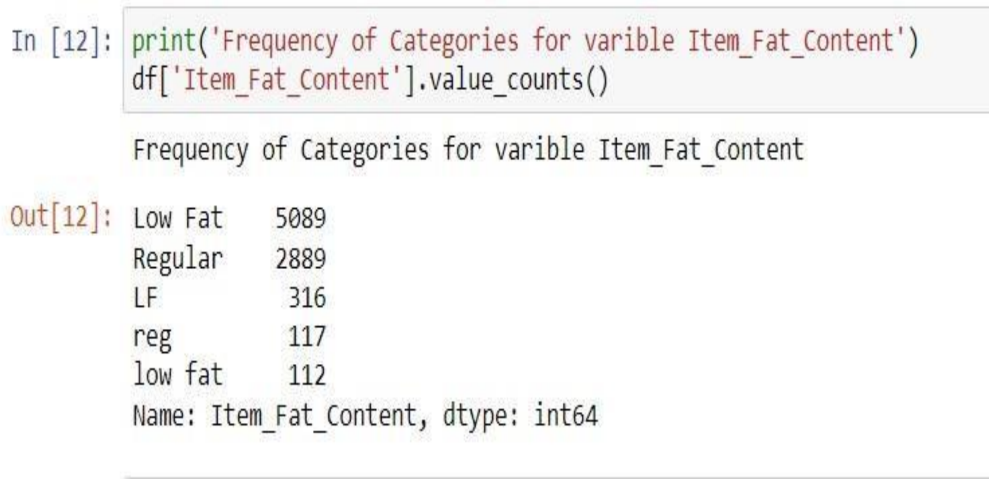
Fig **6.2.4.2 Numerical attributes**

Mean: Mean value of the entire column and in the dataset.

Min & Max: Minimum and Maximum values of the entire column.

6.2.5 Univariate Analysis

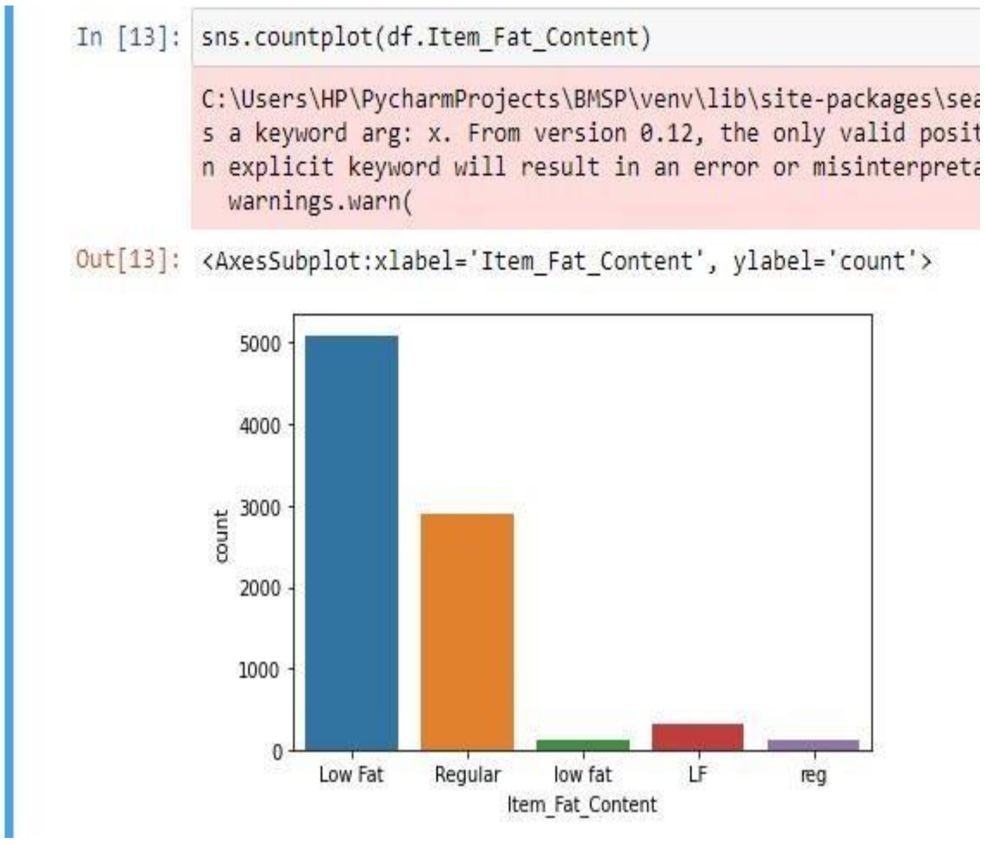
This kind of analysis which is Univariate helps to find and analyze all the information about a single attribute and Variable in the dataset. So that we can determine the uniformity and any kind of uneven nature of the variable can also be predetermined.



```
In [12]: print('Frequency of Categories for varible Item_Fat_Content')
         df['Item_Fat_Content'].value_counts()

         Frequency of Categories for varible Item_Fat_Content

Out[12]: Low Fat    5089
         Regular    2889
         LF          316
         reg         117
         low fat     112
         Name: Item_Fat_Content, dtype: int64
```

Fig 6.2.5 Frequency of categories for variable Item_Fat_Content

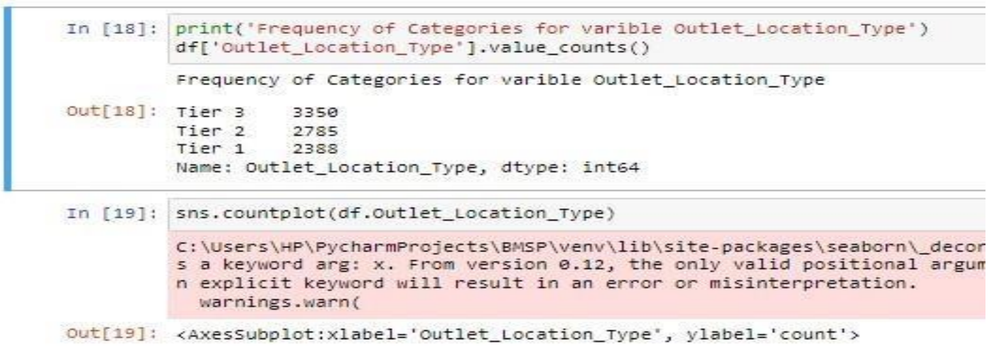
From the above analysis we can find some irregularities in the distribution of the Item_Fat_Content so that can be made normalized to acquire more Uniformity which in order gives the more accuracy of the Model of any algorithm that is trained.



```
In [13]: sns.countplot(df.Item_Fat_Content)

         C:\Users\HP\PycharmProjects\BMSP\venv\lib\site-packages\sea
         s a keyword arg: x. From version 0.12, the only valid posit
         n explicit keyword will result in an error or misinterpreta
           warnings.warn(

Out[13]: <AxesSubplot:xlabel='Item_Fat_Content', ylabel='count'>
```

Fig 6.2.5.1 Showing the count of observations

seaborn.countplot() method is used to Show the counts of observations in each categorical bin using bars. It returns the axes object with the plot drawn onto it.



```
In [18]:  print('Frequency of Categories for varible Outlet_Location_Type')
          df['Outlet_Location_Type'].value_counts()

          Frequency of Categories for varible Outlet_Location_Type
Out[18]:  Tier 3    3350
          Tier 2    2785
          Tier 1    2388
          Name: Outlet_Location_Type, dtype: int64

In [19]:  sns.countplot(df.Outlet_Location_Type)

          C:\Users\HP\PycharmProjects\BMSP\venv\lib\site-packages\seaborn\_decor
          s a keyword arg: x. From version 0.12, the only valid positional argum
          n explicit keyword will result in an error or misinterpretation.
            warnings.warn(

Out[19]:  <AxesSubplot:xlabel='Outlet_Location_Type', ylabel='count'>
```

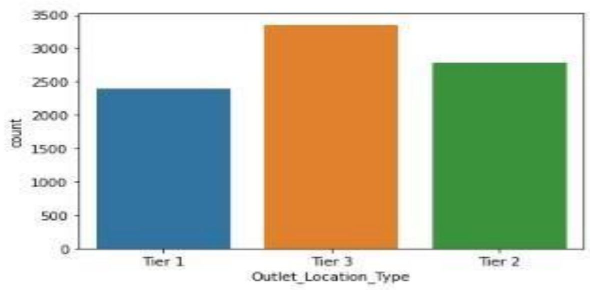
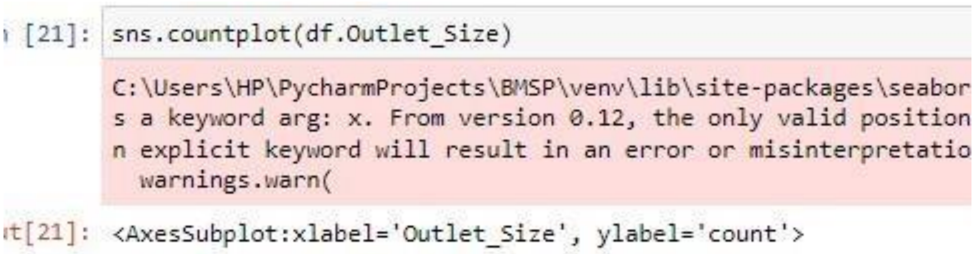


Fig 6.2.5.2 Showing the no. of observations in different locations

The attribute Outlet_Location_Type is a kind of attribute which will affect the sales of any kind of mart. In our dataset, the data of Tier3 markets are more compared to Tier1 and Tier2. So, they can be normalized further.



```
[21]:  sns.countplot(df.Outlet_Size)

       C:\Users\HP\PycharmProjects\BMSP\venv\lib\site-packages\seabor
       s a keyword arg: x. From version 0.12, the only valid position
       n explicit keyword will result in an error or misinterpretatio
         warnings.warn(

t[21]:  <AxesSubplot:xlabel='Outlet_Size', ylabel='count'>
```

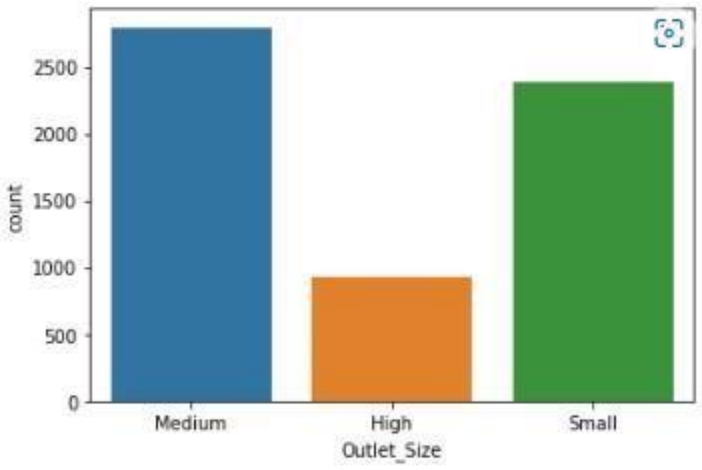


Fig 6.2.5.3 Showing the observations based on Outlet_Size

6.2.6 Bivariate analysis

Bivariate analysis means the analysis of the bivariate data. This is a single statistical analysis that is used to find out the relationship that exists between two value sets. The variables that are involved are X and Y.

```
In [30]:  plt.xlabel("Item_Identifier")
          plt.ylabel("SQRT Item_Outlet_Sales")
          plt.title("Item_Identifier  and Item_Outlet_Sales Analysis")
          plt.plot(df.Item_Identifier , df["Item_Outlet_Sales"],'.', alpha = 0.3)

Out[30]:  [<matplotlib.lines.Line2D at 0x2aad5882820>]
```

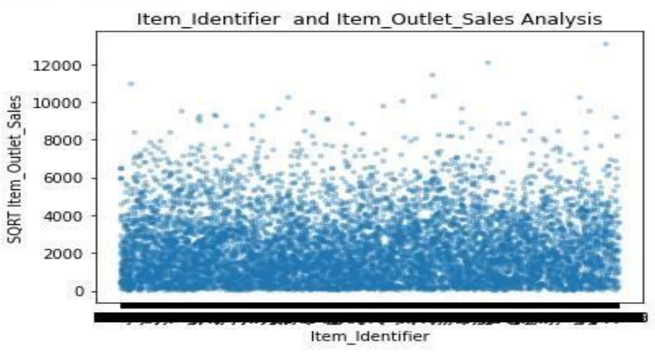


Fig 6.2.6 Relation between Item_Identifier and Item_Outlet_Sales

```
In [183]: sns.regplot(x='Item_MRP',y='Item_Outlet_Sales',data=df)

Out[183]: <AxesSubplot:xlabel='Item_MRP', ylabel='Item_Outlet_Sales'>
```

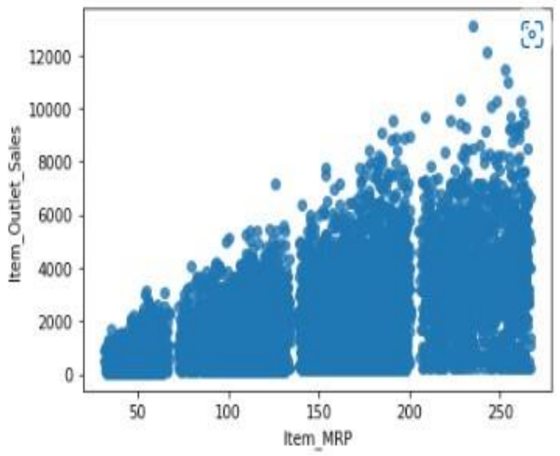
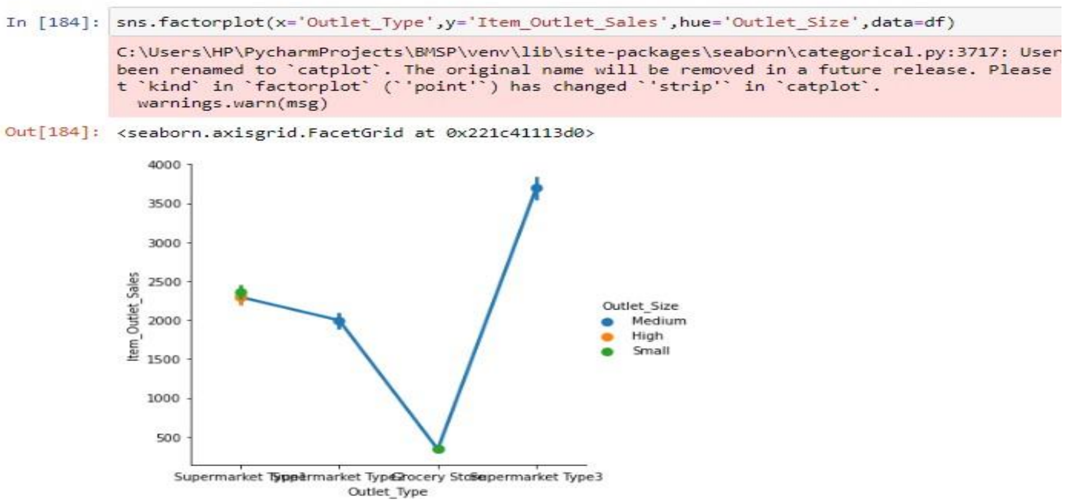
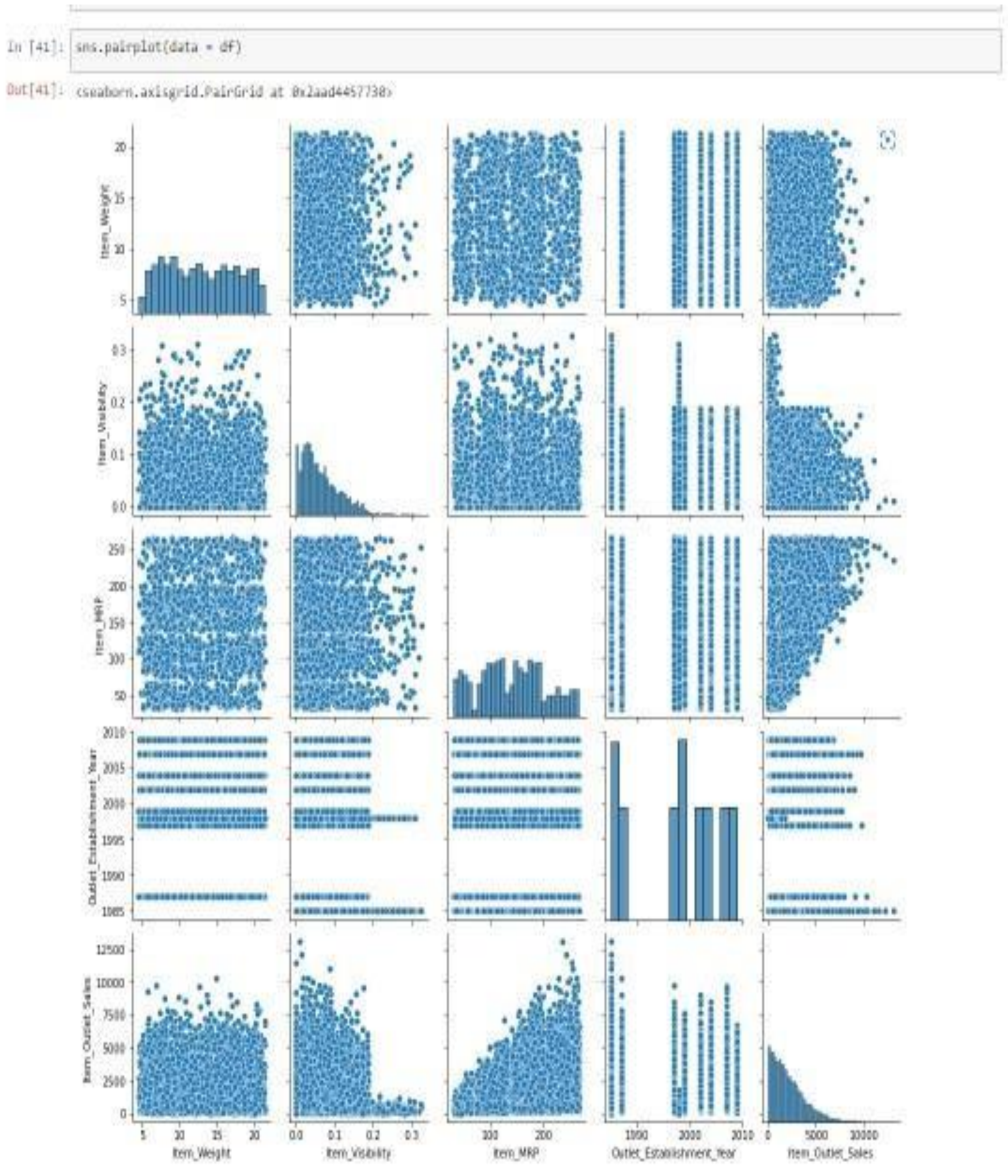


Fig 6.2.6.1 Relation between Item_MRP and Item_Outlet_Sales



```
In [184]: sns.factorplot(x='Outlet_Type',y='Item_Outlet_Sales',hue='Outlet_Size',data=df)

          C:\Users\HP\PycharmProjects\BMSP\venv\lib\site-packages\seaborn\categorical.py:3717: User
          been renamed to `catplot`. The original name will be removed in a future release. Please
          t `kind` in `factorplot` (`'point'`) has changed `'strip'` in `catplot`.
            warnings.warn(msg)

Out[184]: <seaborn.axisgrid.FacetGrid at 0x221c41113d0>
```

Fig 6.2.6.2 Relation between Outlet_Type and Item_Outlet_Sales

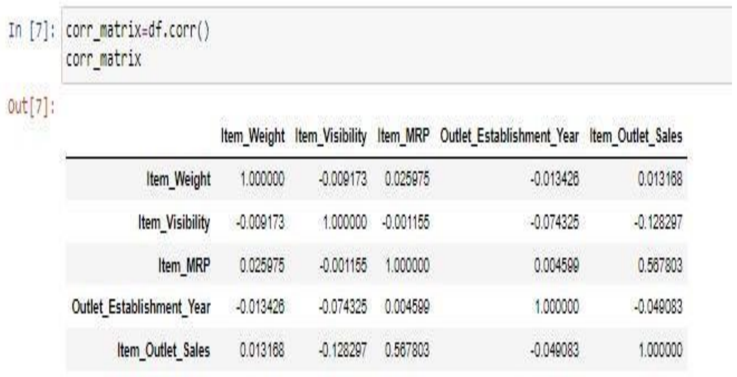


```
In [41]: sns.pairplot(data = df)

Out[41]: <seaborn.axisgrid.PairGrid at 0x2aad4457738>
```

Fig 6.2.6.3 Relationship between all attributes

6.2.7 Correlation matrix

Correlation is an indication about the changes between two variables. We can plot correlation matrix to show which variable is having a high or low correlation in respect to another variable.

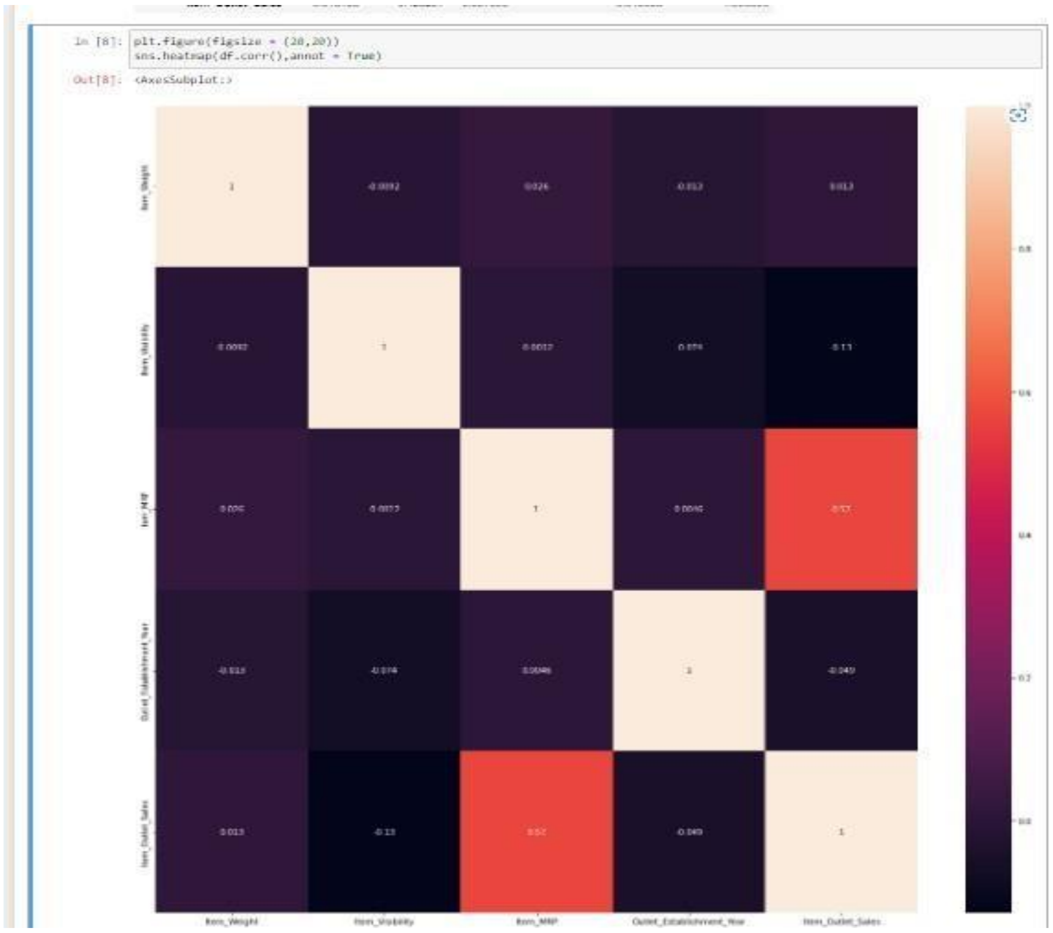


```
In [7]: corr_matrix=df.corr()
        corr_matrix
Out[7]:
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
Item_Weight	1.000000	-0.009173	0.025975	-0.013426	0.013168
Item_Visibility	-0.009173	1.000000	-0.001155	-0.074325	-0.128297
Item_MRP	0.025975	-0.001155	1.000000	0.004599	0.567803
Outlet_Establishment_Year	-0.013426	-0.074325	0.004599	1.000000	-0.049083
Item_Outlet_Sales	0.013168	-0.128297	0.567803	-0.049083	1.000000

Fig 6.2.7 Correlation between the variables

The above values show the correlation between two variables. They help to determine the rate of change between two variables.



```
In [8]: plt.figure(figsize = (20,20))
        sns.heatmap(df.corr(),annot = True)
Out[8]: <AxesSubplot:>
```

Fig 6.2.7.1 Correlation among all the attributes

6.3. Feature Engineering

What is a feature and why do we need the engineering of it? Basically, all machine learning algorithms use some input data to create outputs. This input data comprise features, which are usually in the form of structured columns. Algorithms require features with some specific characteristics to work properly. Here, the need for feature engineering arises. I think feature engineering efforts mainly have two goals:

- Preparing the proper input dataset, compatible with the machine learning algorithm requirements.
- Improving the performance of machine learning models.

Item_Visibility is one of the features of the dataset in which maximum number of values in that row are Zeros so they has to be normalised and are set to some value.So,mean of the entire column of data_visibilty is calculated and the value is set to that mean value.

This make all the Zeros of the column to particular value and accuracy of the model can be made more efficient.

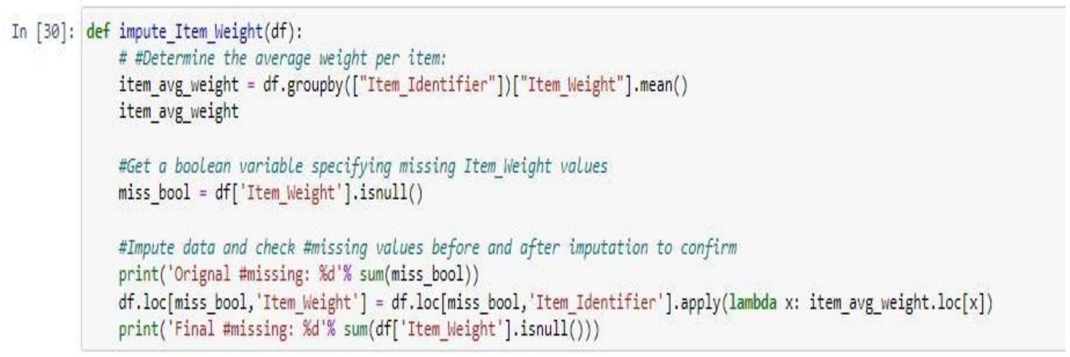
In this step the Item_type attribute has to be normalized with the uniformity in values so such a process is made for the attributes Food, Non-consumable and Drinks. There are attributes Low fat, LF, low fat and also Regular, reg where they belong to same category but are having different names so they can be combined and can be made into single variable for Low fat and one more variable for Regular kind of items. This categorization can reduce 5 different kind of variables to two.

All the non-consumable products of the Item_type Low fat are divided and are made into separate categories so that they can be accessed more well and better, than in combined state. This helps to make values more accurate and normalized than before.

6.4. Filling Missing values and Label encoding

6.4.1. Filling Missing values

With Median and Mode:



```
In [30]: def impute_Item_Weight(df):
             # #Determine the average weight per item:
             item_avg_weight = df.groupby(["Item_Identifier"])["Item_Weight"].mean()
             item_avg_weight

             #Get a boolean variable specifying missing Item_Weight values
             miss_bool = df['Item_Weight'].isnull()

             #Impute data and check #missing values before and after imputation to confirm
             print('Orignal #missing: %d'% sum(miss_bool))
             df.loc[miss_bool,'Item_Weight'] = df.loc[miss_bool,'Item_Identifier'].apply(lambda x: item_avg_weight.loc[x])
             print('Final #missing: %d'% sum(df['Item_Weight'].isnull()))
```

Fig 6.4.1 Filling the missing values

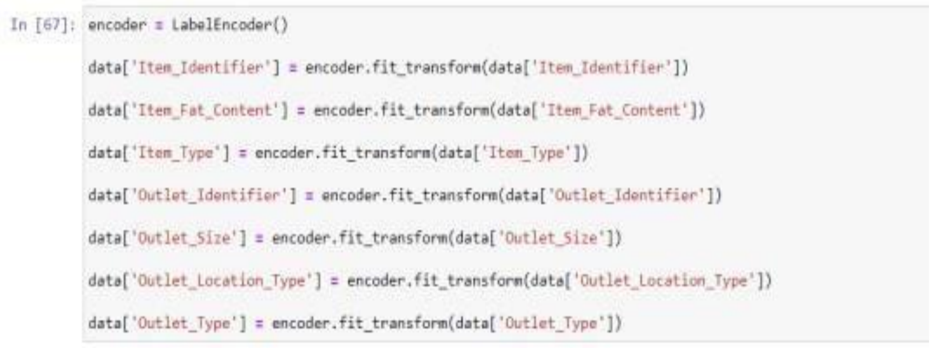
In the dataset the values of the attributes with outlet_size and Outlet_type are having the NULL values in them. Those can affect the accuracy of the model. These situations can be handled by either removing the column or filling it by Mean or Mode of the column.

If the column is removed this may adversely affect the accuracy, So this can be done by filling the columns of numeric values with Mean of the column and Column with categorical values are filled with the Mode of the column.

6.4.2. Label encoding

Label encoding is done for all the columns with the categorical variables where all they are strings so they have to be converted into the numbers for all the columns.

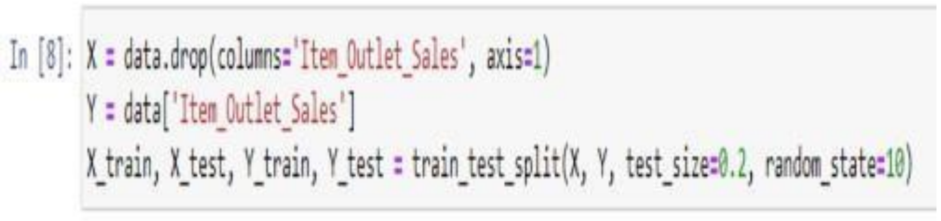
In the dataset collected for this project there are 7 attributes or columns which are having categorical attributes so they can be encoded with the labels by performing the process of Label Encoding.



```
In [67]: encoder = LabelEncoder()
         data['Item_Identifier'] = encoder.fit_transform(data['Item_Identifier'])
         data['Item_Fat_Content'] = encoder.fit_transform(data['Item_Fat_Content'])
         data['Item_Type'] = encoder.fit_transform(data['Item_Type'])
         data['Outlet_Identifier'] = encoder.fit_transform(data['Outlet_Identifier'])
         data['Outlet_Size'] = encoder.fit_transform(data['Outlet_Size'])
         data['Outlet_Location_Type'] = encoder.fit_transform(data['Outlet_Location_Type'])
         data['Outlet_Type'] = encoder.fit_transform(data['Outlet_Type'])
```

Fig 6.4.2 Label Encoding

6.4.3. Splitting the dataset



```
In [8]: X = data.drop(columns='Item_Outlet_Sales', axis=1)
        Y = data['Item_Outlet_Sales']
        X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=10)
```

Fig 6.4.3 Splitting the dataset

For all the Machine learning models to train with any algorithm of their choice the dataset has to be divided into two parts called Training dataset and Testing dataset.

Generally, the training dataset will be 80% of the entire dataset and 20% of the data as the Testing dataset.

Training dataset will be used to train the model and testing dataset will be used to find the accuracy of our predicted model. Performance evaluation can be made with the accuracy of the trained model with required algorithm.

Those splitting of the dataset to train and test splitting can be made using the command from the sklearn library as shown above.

X_train-Represents train dataset.

X_test-Represents test dataset.

6.5. Model prediction

In this prediction the entire data is trained with three different models in which each model provides different output values of different accuracies. All the models are compared and the conclusions are made.

6.5.1. Modelling with Linear Regression Algorithm

Step1: Load the dataset

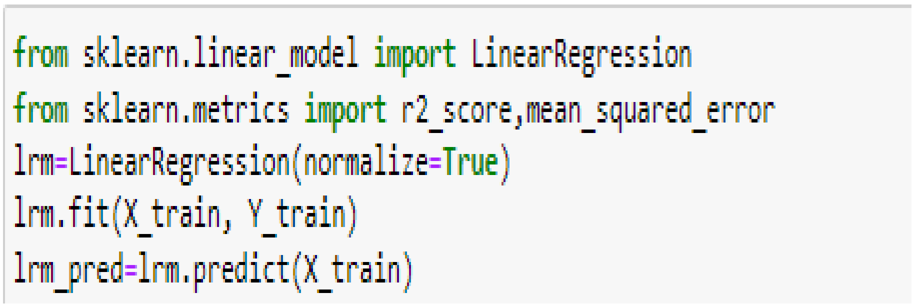
Step2: Divide the dataset

Step3: Assign the linear regression algorithm to a variable.

Step4: Fit the training dataset using Linear regression algorithm

Step5: Predict the values for the testing dataset using a trained model.

Step6: Check the accuracy of the model.



```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error
lrm=LinearRegression(normalize=True)
lrm.fit(X_train, Y_train)
lrm_pred=lrm.predict(X_train)
```

Fig6.5.1 Code for Linear Regression Model

lrm -	Variable of Linear regression algorithm
.fit() -	fit method of sklearn
.predict()-	predict method for predicting the values
X_train,Y_train-	Training data
X_test,Y_test-	Testing data

6.5.2. Modelling with Ridge Regression Algorithm:

Step1: Load the dataset

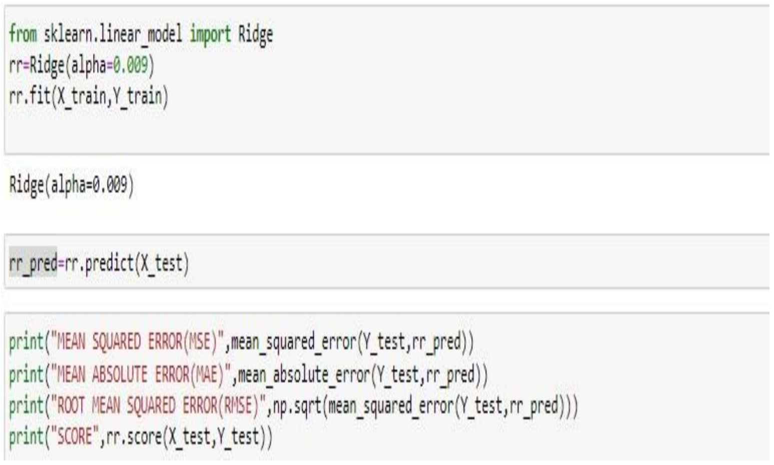
Step2: Divide the dataset

Step3: Assign the Ridge regression algorithm to a variable.

Step4: Fit the training dataset using Ridge regression algorithm

Step5: Predict the values for the testing dataset using a trained model.

Step6: Check the accuracy of the model.



```
from sklearn.linear_model import Ridge
rr=Ridge(alpha=0.009)
rr.fit(X_train,Y_train)
```

```
Ridge(alpha=0.009)
```

```
rr_pred=rr.predict(X_test)
```

```
print("MEAN SQUARED ERROR(MSE)",mean_squared_error(Y_test,rr_pred))
print("MEAN ABSOLUTE ERROR(MAE)",mean_absolute_error(Y_test,rr_pred))
print("ROOT MEAN SQUARED ERROR(RMSE)",np.sqrt(mean_squared_error(Y_test,rr_pred)))
print("SCORE",rr.score(X_test,Y_test))
```

Fig 6.5.2 Code for Ridge Regression

rr	Variable of Ridge regression algorithm
.fit()-	fit method of sklearn
.predict()-	predict method for predicting the values

X_train,Y_train-	Training data
X_test,Y_test-	Testing data

6.5.3. Modelling with Random Forest Algorithm:

Step1: Load the dataset

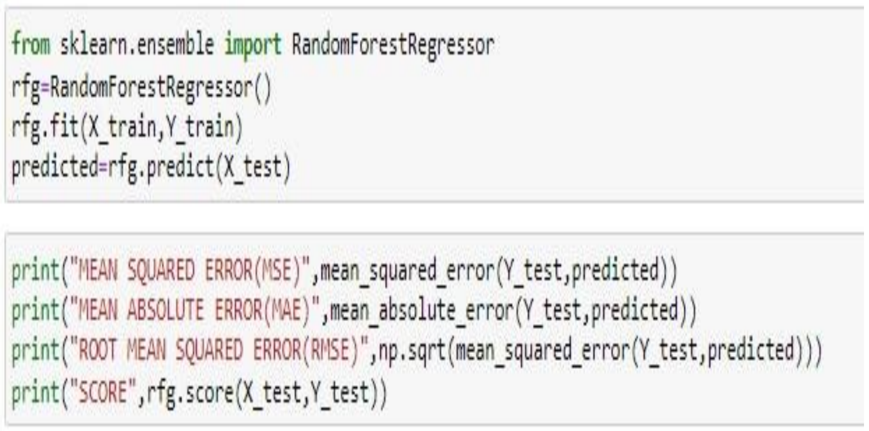
Step2: Divide the dataset

Step3: Assign the Random Forest algorithm to a variable.

Step4: Fit the training dataset using Random Forest algorithm

Step5: Predict the values for the testing dataset using a trained model.

Step6: Check the accuracy of the model.



```
from sklearn.ensemble import RandomForestRegressor
rfg=RandomForestRegressor()
rfg.fit(X_train,Y_train)
predicted=rfg.predict(X_test)
```

```
print("MEAN SQUARED ERROR(MSE)",mean_squared_error(Y_test,predicted))
print("MEAN ABSOLUTE ERROR(MAE)",mean_absolute_error(Y_test,predicted))
print("ROOT MEAN SQUARED ERROR(RMSE)",np.sqrt(mean_squared_error(Y_test,predicted)))
print("SCORE",rfg.score(X_test,Y_test))
```

Fig 6.5.3: Code for Random Forest Regressor

rfg-	Variable of Random forest algorithm
.fit()-	fit method of sklearn
.predict()-	predict method for predicting the values
X_train,Y_train-	Training data
X_test,Y_test-	Testing data

6.5.4 Modelling with XG Boost Algorithm:

Step1: Load the dataset

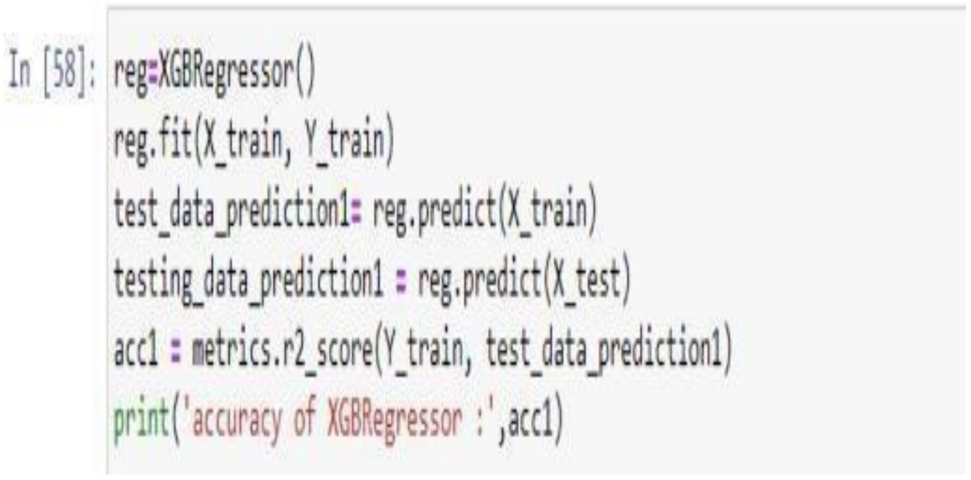
Step2: Divide the dataset

Step3: Assign the XG Boost regression algorithm to a variable.

Step4: Fit the training dataset using XG Boost regression algorithm

Step5: Predict the values for the testing dataset using a trained model.

Step6: Check the accuracy of the model.



```
In [58]: reg=XGBRegressor()
         reg.fit(X_train, Y_train)
         test_data_prediction1= reg.predict(X_train)
         testing_data_prediction1 = reg.predict(X_test)
         acc1 = metrics.r2_score(Y_train, test_data_prediction1)
         print('accuracy of XGBRegressor :',acc1)
```

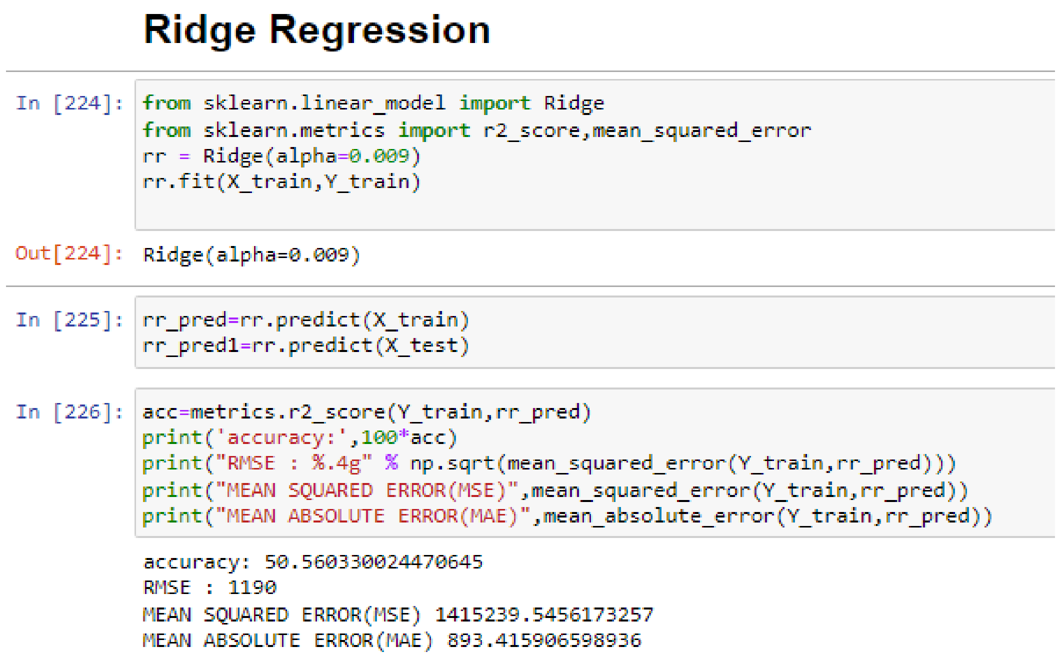
Fig 6.5.4 Code for XG Boost Regression

reg-	Variable of XG Boost regression algorithm
.fit()-	fit method of sklearn
.predict()-	predict method for predicting the values
X_train,Y_train-	Training data
X_test,Y_test-	Testing data

CHAPTER-7

PERFORMANCE EVLUATION

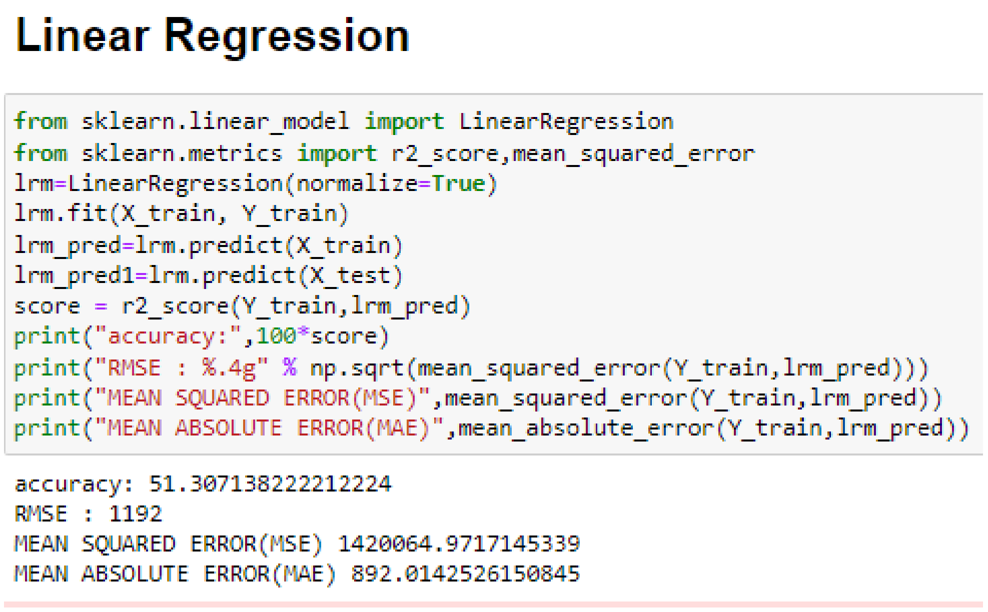
7.1 Score method: It is a kind of method used to evaluate the performance of the model. Performance evaluation is made for this project using Score method of the sklearn library of Python. The score method is applied for all three algorithms as follows:



Ridge Regression

```
In [224]: from sklearn.linear_model import Ridge
          from sklearn.metrics import r2_score,mean_squared_error
          rr = Ridge(alpha=0.009)
          rr.fit(X_train,Y_train)

Out[224]: Ridge(alpha=0.009)

In [225]: rr_pred=rr.predict(X_train)
          rr_pred1=rr.predict(X_test)

In [226]: acc=metrics.r2_score(Y_train,rr_pred)
          print('accuracy:',100*acc)
          print("RMSE : %.4g" % np.sqrt(mean_squared_error(Y_train,rr_pred)))
          print("MEAN SQUARED ERROR(MSE)",mean_squared_error(Y_train,rr_pred))
          print("MEAN ABSOLUTE ERROR(MAE)",mean_absolute_error(Y_train,rr_pred))

          accuracy: 50.560330024470645
          RMSE : 1190
          MEAN SQUARED ERROR(MSE) 1415239.5456173257
          MEAN ABSOLUTE ERROR(MAE) 893.415906598936
```

Fig 7.1 Accuracy of Ridge regression

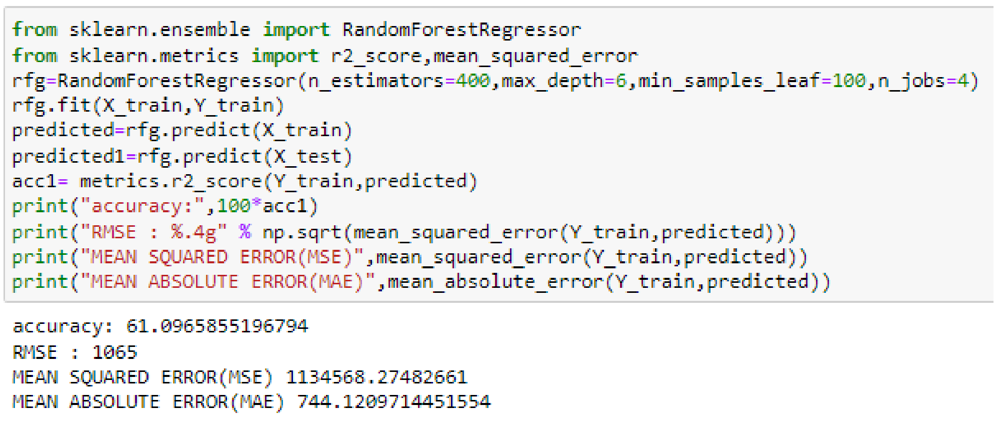


Linear Regression

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error
lrm=LinearRegression(normalize=True)
lrm.fit(X_train, Y_train)
lrm_pred=lrm.predict(X_train)
lrm_pred1=lrm.predict(X_test)
score = r2_score(Y_train,lrm_pred)
print("accuracy:",100*score)
print("RMSE : %.4g" % np.sqrt(mean_squared_error(Y_train,lrm_pred)))
print("MEAN SQUARED ERROR(MSE)",mean_squared_error(Y_train,lrm_pred))
print("MEAN ABSOLUTE ERROR(MAE)",mean_absolute_error(Y_train,lrm_pred))

accuracy: 51.307138222212224
RMSE : 1192
MEAN SQUARED ERROR(MSE) 1420064.9717145339
MEAN ABSOLUTE ERROR(MAE) 892.0142526150845
```

Fig 7.1.1 Accuracy of Linear regression

Random Forest

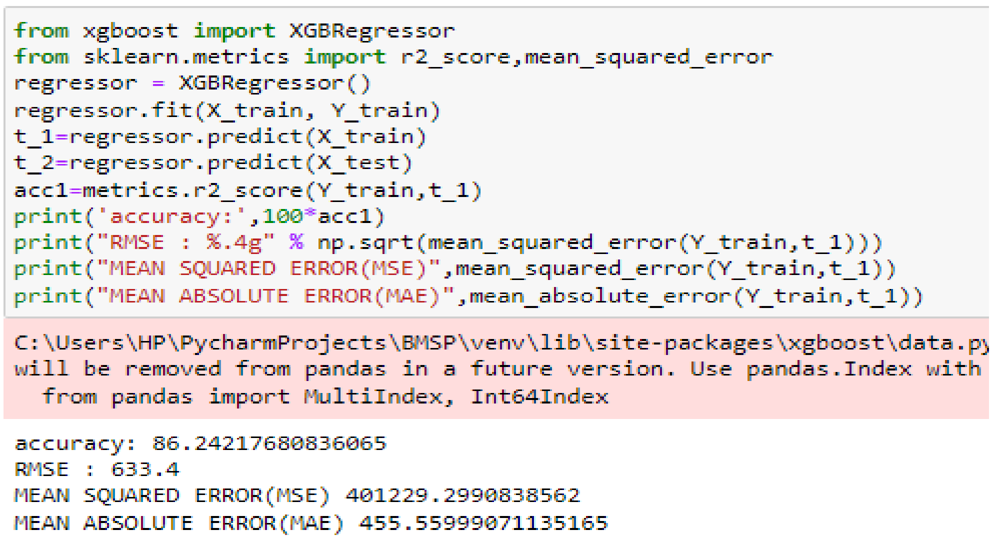


```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score,mean_squared_error
rfg=RandomForestRegressor(n_estimators=400,max_depth=6,min_samples_leaf=100,n_jobs=4)
rfg.fit(X_train,Y_train)
predicted=rfg.predict(X_train)
predicted1=rfg.predict(X_test)
acc1= metrics.r2_score(Y_train,predicted)
print("accuracy:",100*acc1)
print("RMSE : %.4g" % np.sqrt(mean_squared_error(Y_train,predicted)))
print("MEAN SQUARED ERROR(MSE)",mean_squared_error(Y_train,predicted))
print("MEAN ABSOLUTE ERROR(MAE)",mean_absolute_error(Y_train,predicted))
```

```
accuracy: 61.0965855196794
RMSE : 1065
MEAN SQUARED ERROR(MSE) 1134568.27482661
MEAN ABSOLUTE ERROR(MAE) 744.1209714451554
```

Fig 7.1.2 Accuracy of Random Forest regression

XGBoost



```
from xgboost import XGBRegressor
from sklearn.metrics import r2_score,mean_squared_error
regressor = XGBRegressor()
regressor.fit(X_train, Y_train)
t_1=regressor.predict(X_train)
t_2=regressor.predict(X_test)
acc1=metrics.r2_score(Y_train,t_1)
print('accuracy:',100*acc1)
print("RMSE : %.4g" % np.sqrt(mean_squared_error(Y_train,t_1)))
print("MEAN SQUARED ERROR(MSE)",mean_squared_error(Y_train,t_1))
print("MEAN ABSOLUTE ERROR(MAE)",mean_absolute_error(Y_train,t_1))
```

```
C:\Users\HP\PycharmProjects\BMSP\venv\lib\site-packages\xgboost\data.py
will be removed from pandas in a future version. Use pandas.Index with
  from pandas import MultiIndex, Int64Index
```

```
accuracy: 86.24217680836065
RMSE : 633.4
MEAN SQUARED ERROR(MSE) 401229.2990838562
MEAN ABSOLUTE ERROR(MAE) 455.55999071135165
```

Fig 7.1.3 Accuracy of XG Boost regression

7.2 User Interface

Here we developed a web page where we give the values of the attributes and then by clicking the button called “Predict” it gives the value which is the future scope of a particular product in the means of highest value.

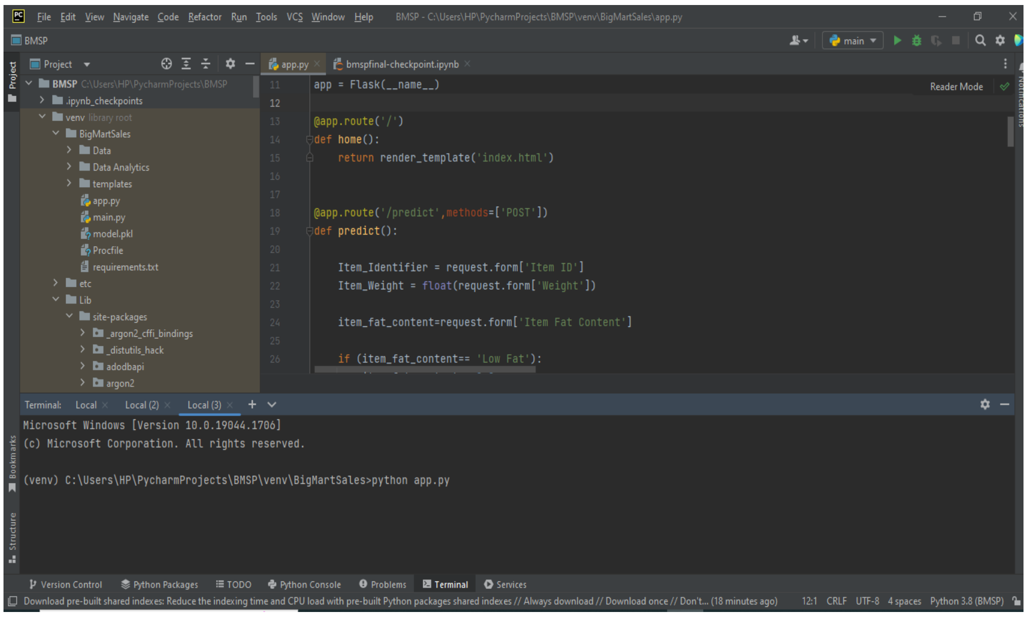
Flask is basically a python module. It can work with python only and it is a web-developing framework. It is a collection of libraries and modules. Frameworks are used for developing web platforms. Flask is such a type of web application framework. It is completely written in Python language. Unlike Django, it is only written in Python. As a new user Flask is to be used. As it is easier to handle. As it is only written in Python, before installing Flask on the machine, Python should be installed previously. Also, Python Pip should be installed. Nowadays, the latest version of Python Pip is already installed.

Features:

- Flask is easy to use and easily understandable for new users in Web Framework.
- It can also be used as any third-party plugin extension.
- It is also used for prototyping purposes.

We are using flask web framework of version 2.0.0 to implement the web page. It provides libraries to build lightweight web applications in python language. Install the flask by using

pip install flask



```python
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')


@app.route('/predict',methods=['POST'])
def predict():

    Item_Identifier = request.form['Item ID']
    Item_Weight = float(request.form['Weight'])

    item_fat_content=request.form['Item Fat Content']

    if (item_fat_content== 'Low Fat'):
```

```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

(venv) C:\Users\HP\PycharmProjects\BMSP\venv\BigMartSales>python app.py
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't... (18 minutes ago)

Fig 7.2 Code for developing web page

The above code is about the importing flask framework and using pickle library which is used to import the Machine learning model to front-end. Here, we are taking XG Boost Regression algorithm because we got high accuracy for this model.

app=flask(--name--)

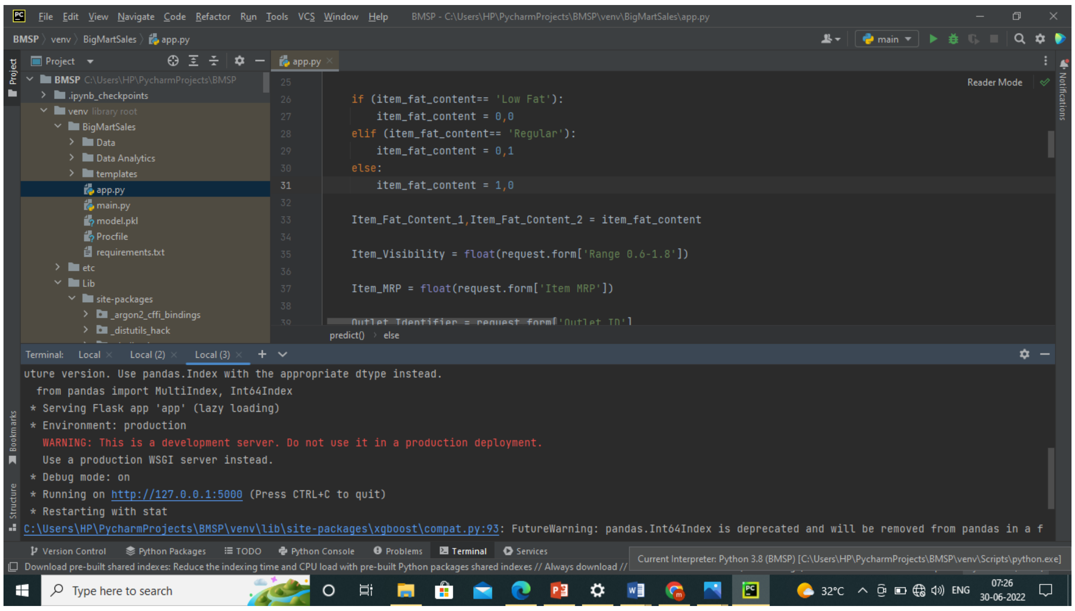
@app.route("/")

This is the syntax to create the web application in flask using python language. App Routing means mapping the URLs to a specific function that will handle the logic for that URL. Modern web frameworks use more meaningful URLs to help users remember the URLs and make navigation simpler. To bind a function to an URL path we use the app.route decorator.

We save our file app.py in the BigMart Sales folder in order to change the directory by using cd command. To run the file app.py, use the command python app.py. By clicking the enter it gives link which redirects to the page where a sales manager can fill the values of the attributes and can predict the future sale in the means of highest number.

Here is the link: **http://127.0.0.1:5000**

The below screenshot showing command to run the application and it gives the link.



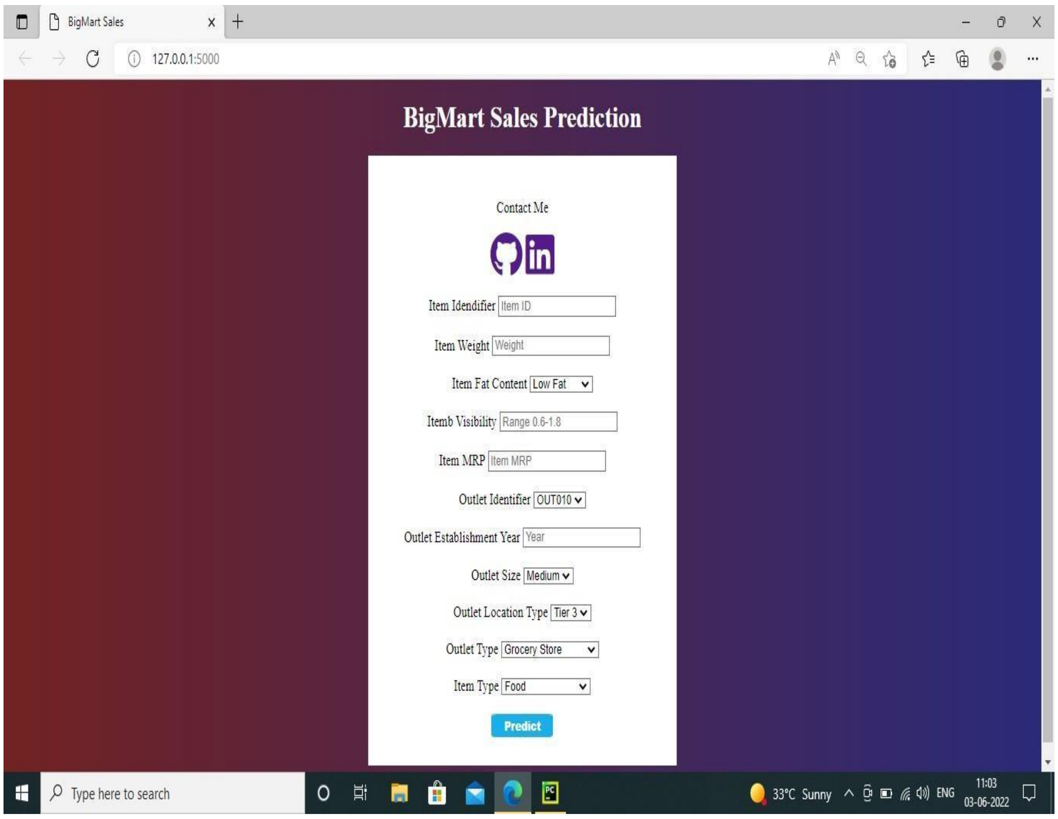
```
    if (item_fat_content== 'Low Fat'):
        item_fat_content = 0,0
    elif (item_fat_content== 'Regular'):
        item_fat_content = 0,1
    else:
        item_fat_content = 1,0

    Item_Fat_Content_1,Item_Fat_Content_2 = item_fat_content

    Item_Visibility = float(request.form['Range 0.6-1.8'])

    Item_MRP = float(request.form['Item MRP'])

    Outlet_Identifier = request.form['Outlet_ID']
```

```
uture version. Use pandas.Index with the appropriate dtype instead.
  from pandas import MultiIndex, Int64Index
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
 * Restarting with stat
C:\Users\HP\PycharmProjects\BMSP\venv\lib\site-packages\xgboost\compat.py:93: FutureWarning: pandas.Int64Index is deprecated and will be removed from pandas in a f
```

BMSP - C:\Users\HP\PycharmProjects\BMSP\venv\BigMartSales\app.py
BMSP › venv › BigMartSales › app.py
Project
BMSP C:\Users\HP\PycharmProjects\BMSP
.ipynb_checkpoints
venv library root
BigMartSales
Data
Data Analytics
templates
app.py
main.py
model.pkl
Procfile
requirements.txt
etc
Lib
site-packages
_argon2_cffi_bindings
_distutils_hack
predict() › else
Terminal: Local Local (2) Local (3)
Current Interpreter: Python 3.8 (BMSP) [C:\Users\HP\PycharmProjects\BMSP\venv\Scripts\python.exe]
Type here to search
32°C
07:26
30-06-2022

Fig 7.2.1 Running the application program

By clicking the above link, it will redirects to this page.



BigMart Sales Prediction

Contact Me

Item Idendifier: Item ID

Item Weight: Weight

Item Fat Content: Low Fat

Itemb Visibility: Range 0.6-1.8

Item MRP: Item MRP

Outlet Identifier: OUT010

Outlet Establishment Year: Year

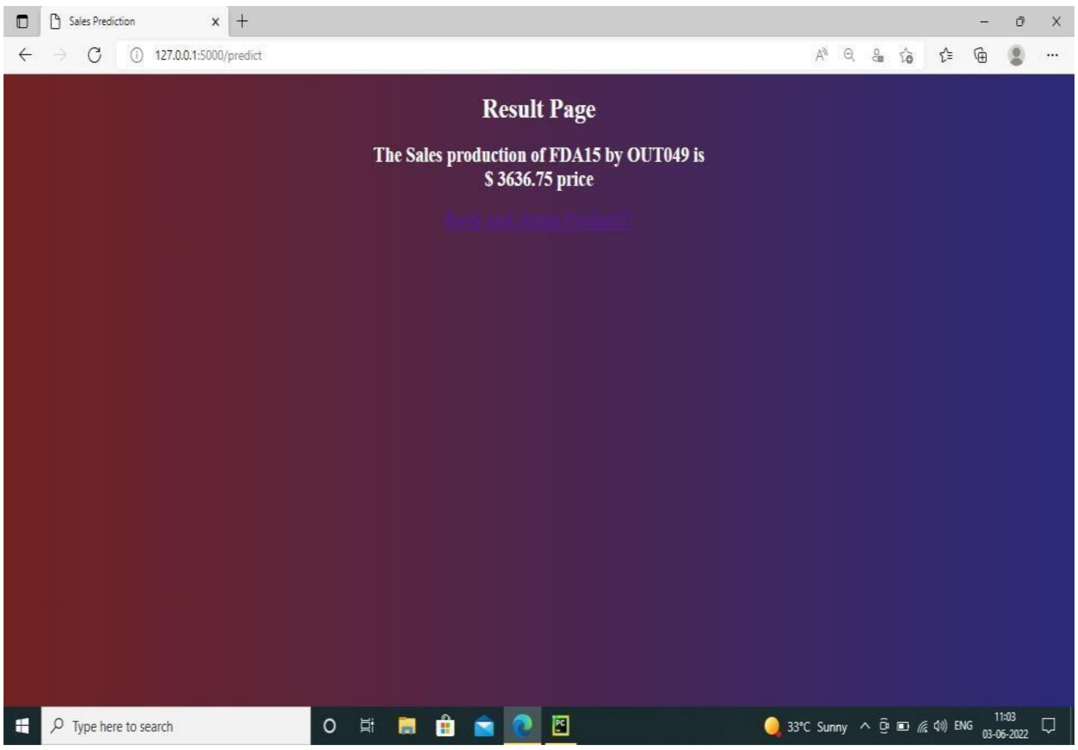
Outlet Size: Medium

Outlet Location Type: Tier 3

Outlet Type: Grocery Store

Item Type: Food

Fig 7.2.2 Web page containing the values of 11 attributes



Result Page

The Sales production of FDA15 by OUT049 is
$ 3636.75 price

Fig 7.2.3 Predicted output value

CONCLUSION

Prediction of sales is very much necessary for any kind of business irrespective of size of the business. Profit is very important for any business thus it is necessary to cut down the loss for any kind of business. This can be made only we can predict the sales of next day or next month or next year. It helps to cut the amount that is invested on Inventory stock and helps to invest more into the business and gain more profit than before.

Sales prediction of any scenario thus can be made using the algorithms of the regression called Linear regression, Ridge regression, Random Forest and XG Boost Regression algorithms. Linear regression fits the data using linear equation as the fitting model and Ridge regression of degree 4 fits the data using Ridge equation of degree 4 and thus fits the data using that model. XG Boost regression uses the technique of ensemble learning and provides the output of the sales prediction. Thus, Sales are predicted using all the three algorithms and acquire the output of sales for all the algorithms separately.

By performing the analysis using all the four algorithms XG Boost regressor provided the high accuracy for the trained model and predicts the sales more accurately compared to the Linear, Random Forest, Ridge regression algorithms. XG Boost regression provided the accuracy of the model with 86.24%. By this we can conclude that the prediction of sales is made using Regression algorithms and among them XG Boost regression can predict more accurately compared to the other algorithms.

REFERENCES

[1] Nikita Malik, Karan Singh, "Big Mart Sales Prediction", Maharaja Surajmal Institute Journal of Applied Reasearch, where the paper published, vol 3, Issue 1; January-June 2020.

[2] Heramb Kadam, Rahul Shevade, Deven Ketkar, Sufiyan Rajguru "A Forecast for Big Mart Sales Based on Random Forest and Multiple Linear Regression", International Journal of Engineering Development and Research(IJEDR 2018),where the paper published, volume 6.

[3] Gopal Behara, Neeta Nain, "A Comparative study of Big mart sales prediction" Malaviya National Institute of Technology Jaipur, India.

[4] Smola, A, & Vishwanathan, S.V.N.(2008). Introduction to machine learning. Cambridge University, UK, 32, 34.

[5] Saltz, J. S., & Stanton, J. M. (2017). An introduction to data science. Sage Publications.

[6] Shashua, A. (2009). Introduction to machine learning: Class notes 67577.

[7] *Dataset* -Downloaded from Kaggle website.

[8] Meghana N, Pavan Chatradi, Avinash Chakravarthy V, Sai Mythri Kalavala,, Mrs.Neetha K S," Improvizing Big Mart Sales Prediction", Journal of Xi'an University of Architecture & Technology, ISSN No : 1006-7930