A PROJECT REPORT ON

**PREDICTIVE ANALYSIS OF BIG MART SALES USING MACHINE LEARNING ALGORITHM**

**Submitted to**

**SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY**

**(Autonomous)**

**Affiliated to JNTUA, ANANTAPURAMU**

*in partial fulfilment of the requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**SUBMITTED BY**

A. GANESH 21785A0501

D. VENKATA CHARAN 21785A0506

D. GIRIA 21785A0507

M. SAI TEJA 21785A0516

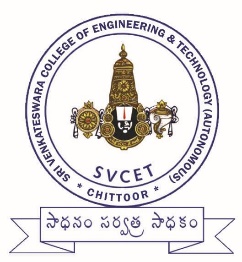
P. KIRANMAI 21785A0518

Under the esteemed guidance of

**MR.P. THIRUMURUGAN,**

Associate Professor,

**Department of Computer Science & Engineering**



**SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY**

**Affiliated to JNTU, ANANTAPURAMU-515002 (A.P)**

**& Approved by AICTE, New Delhi Accredited by NAAC, Bengaluru & NBA, New Delhi**

**An ISO 9001-2000 Certified Institution**

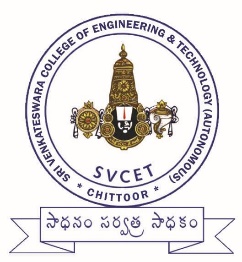
**R.V.S Nagar, CHITTOOR 517127(AP)**

**MAY-2024**

**SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY**

(Autonomous)

Affiliated to JNTUA, ANANTHAPURAMU & Approved by AICTE, New Delhi Accredited by NAAC 'A' grade, Bengaluru & NBA, New Delhi An ISO 9001:2000 Certified Institution R.V.S Nagar, CHITTOOR-517 12 (A.P.) [www.svcetedu.org](http://www.svcetedu.org)



CERTIFICATE

This is to certify that the project entitled **"PREDICTIVE ANALYSIS OF BIG MART SALES USING MACHINE LEARNING ALGORITHM**” is a bonified work done and submitted by "A. GANESH (21785A0501), D. VENKATA CHARAN (21785A0506), D. GIRIJA (21785A0507), M. SAI TEJA (21785A0516), P. KIRANMAI (21785A0518)” Under my supervision and guidance, in partial fulfilment of the requirement for the award of the **"BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING"** during the academic year 2023-2024.

SIGNATURE OF THE GUIDE SIGNATURE OF THE HOD

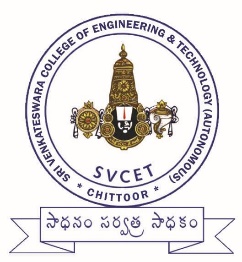
Viva-Voice Conducted on

INTERNAL EXAMINER EXTERNAL EXAMINER

SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY (Autonomous)

Affiliated to JNTUA, ANANTHAPURAMU & Approved by AICTE, New Delhi Accredited by NAAC 'A' grade, Bengaluru & NBA, New Delhi An ISO 9001:2000 Certified Institution R.V.S Nagar, CHITTOOR-517127 (A.P) [www.svcetedu.org](http://www.svcetedu.org)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**DECLARATION**

We A. GANESH (21785A0501), D. VENKATA CHARAN (21785A0506), D. GIRIJA (21785A0507), M. SAI TEJA (21785A0516), P. KIRANMAI (21785A0518) hereby declare that the Project Report entitled

" **PREDICTIVE ANALYSIS OF BIG MART SALES USING MACHINE LEARNING ALGORITHM** " under the guidance of MR.P. THIRUMURUGAN,

Associate Professor, Sri Venkateswara College of Engineering & Technology (Autonomous), Chittoor is submitted in partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING.

This is a record of bonified work carried out by us and the results embodied in this project has not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other university or institute for the award of any other degree.

**SIGNATURE OF THE STUDENT**

A. GANESH 21785A0501

D. VENKATA CHARAN 21785A0506

D. GIRIA 21785A0507

M. SAI TEJA 21785A0516

P. KIRANMAI 21785A0518

**ACKNOWLEDGEMENT**

We are greatly indebted to **BHARATA JYOTHI Dr. RAVURI VENKATASWAMYGARU**, Chairman of the College for giving us the opportunity to fulfil the Project.

We are greatly indebted to **Dr. R. V. SRINIVAS GARU**, Vice Chairman of the College for giving us the opportunity to fulfil the Project.

We would like to express our sincere thanks to **Dr. M. MOHAN BABU**, Principal, S.V.C.E.T., Chittoor, for helping us in many regards throughout the work.

We would like to express our sincere thanks to **Dr. E. LOKANADHA REDDY**, vice- principal, S.V.C.E.T., Chittoor, for helping us in many regards throughout the work.

We wish to express our deep sense of gratitude to **Dr. P JYOTHEESWARI, Head of Department of Computer Science and Engineering**, for her encouragement in this project work.

We would like to express our thanks to Project Coordinator **Mr. P. THIRUMURUGAN,** Associate professor, Dept of CSE, S.V.C.E.T., Chittoor, for helping us in many regards throughout the work.

It is our privilege to record our deep indebtedness to, **Mr. HARE RAM SINGH,** Associate Professor, Department of Computer Science and Engineering, S.V.C.E.T., Chittoor for his constant, encouragement, care, meticulous supervision, and critical evaluation during the investigation.

We will also be thankful to the staff of all laboratories, Department of Computer Science and Engineering, S.V.C.E.T., R.V.S. Nagar, Chittoor for their assistance in laboratory work.

We would like to thank all faculty members of the Computer Science and Engineering Department, S.V.C.E.T., R.V.S. Nagar, Chittoor for their co-operation.

**PROJECT ASSOCIATES**

A. GANESH 21785A0501

D. VENKATA CHARAN 21785A0506

D. GIRIA 21785A0507

M. SAI TEJA 21785A0516

P. KIRANMAI 21785A0518

**ABSTRACT**

Nowadays shopping malls and Big Marts keep the track of their sales data of each and every individual item 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 propose a predictive model using XG boost Regressor technique for predicting the sales of a company like Big Mart and found that the model produces better performance as compared to existing models.

Currently, supermarket run-centres, Big Marts keep track of each individual item's sales data in order to anticipate potential consumer demand and update inventory management. Anomalies and general trends are often discovered by mining the data warehouse's data store. For retailers like Big Mart, the resulting data can be used to forecast future sales volume using various machine learning techniques like big mart. A predictive model was developed using Xgboost, Linear regression, Polynomial regression, and Ridge regression techniques for forecasting the sales of a business such as Big-Mart, and it was discovered that the model outperforms existing models.

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **CONTENTS** | **PAGE NO.** |
|  | **Abstract** |  |
| **1.** | **Introduction**  1.1 Proposed Algorithm | 1 - 8 |
| **2.** | **Literature Survey** | 9 – 13 |
| **3.** | **System Requirements Specification**  3.1 Functional Requirements  3.2 Non-Functional Requirements  **System Specifications**  3.3 Hardware Specifications  3.4 Software Specifications  **Software Installation for Machine Learning Project**  3.5 Installation Python | 14 – 20 |
| **4.** | **System Analysis**  4.1 Purpose  4.2 Scope  4.3 Existing System  4.4 Proposed System | 21 – 23 |
| **5.** | **Introduction To Python**  5.1 What is Script  5.2 Difference Between a Script and a Program  5.3 History of Python  5.4 Python Features  5.5 Dynamic vs Static  5.5.1 Variables  5.5.2 Standard Data Types  5.5.3 Python Numbers  5.5.4 Python Strings  5.5.5 Python Lists  5.5.6 Python Dictionary  5.5.7 Different Modes in Python  5.6 Python Libraries  5.7 NumPy  5.8 Matplotlib  5.9 Testing Codes  5.10 Functions in Python  5.11 Python Namespaces  5.12 Garbage Collection  5.13 Python XML Parser  5.14 Python Web Frameworks | 24 – 42 |
| **6.** | **System Study**  6.1 Feasibility Study  6.2 Economic Feasibility  6.3 Technical Feasibility | 43 – 45 |
| **7.** | **System Design**  7.1 Input Design  7.2 Output Design  7.3 Data Flow Design  7.4 UMl Diagram  7.5 Use Case Diagram  7.6 Sequence Diagram | 46 – 50 |
| **8.** | **Module Implementations**  8.1 NumPy  8.2 Pandas  8.3 Matplotlib  8.4 Seaborn  8.5 Data Collection  8.6 Data Set  8.7 Data Preparation  8.8 Model Selection  8.9 Analyse and Prediction  8.10 Accuracy on Test set  8.11 Saving the Trained Model | 51 – 57 |
| **9.** | **System Testing** | 58 – 62 |
| **10.** | **Output ScreenShots**  10.1 Loading the Data Set  10.2 Data Validation  10.3 Scatter Plot  10.4 Result | 63 – 65 |
| **11.** | **Conclusion** | 66 |
| **12.** | **References** | 67 – 69 |

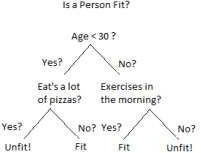
**CHAPTER – 1**

**INTRODUCTION**

Big Mart is a big supermarket chain, with stores all around the country and its current board set out a challenge to all Data Scientist out there to help them create a model that can predict the sales, per product, for each store to give accurate results. Big Mart has collected sales data from Kaggle, for various products across different stores in different cities. With this information the corporation hopes we can identify the products and stores which play a key role in their sales and use that information to take the correct measures to ensure success of their business.

**1.1 PROPOSED ALGORITHMS**

Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split.



An example of a decision tree can be explained using above binary tree. Let’s say you want to predict whether a person is fit given their information like age, eating habit, and physical activity, etc. The decision nodes here are questions like „What’s the age?‟, „Does he exercise?‟, „Does he eat a lot of pizzas‟? And the leaves, which are outcomes like either „fit‟, or „unfit‟. In this case this was a binary classification problem (a yes no type problem).

There are two main types of Decision Trees:

Classification trees (Yes/No types)

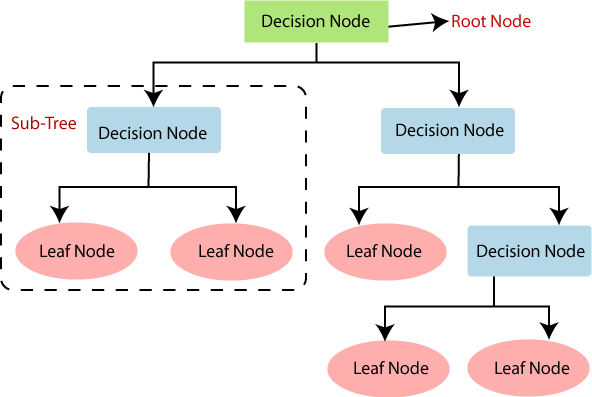
What we’ve seen above is an example of classification tree, where the outcome was a variable like „fit‟ or „unfit‟. Here the decision variable is Categorical.

Regression trees (Continuous data types)

Here the decision or the outcome variable is Continuous, e.g. a number like 123. Working

Now that we know what a Decision Tree is, we’ll see how it works internally. There are many algorithms out there which construct Decision Trees, but one of the best is called as ID3 Algorithm. ID3 Stands for Iterative Dichotomise 3.

Before discussing the ID3 algorithm, we’ll go through few definitions.

* Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
* In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
* A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
* Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

* Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
* The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies:

* **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
* **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
* **Splitting**: Splitting is the process of dividing the decision node/root node into sub- nodes according to the given conditions.
* **Branch/Sub Tree:** A tree formed by splitting the tree.
* **Pruning**: Pruning is the process of removing the unwanted branches from the tree.
* **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

**How does the Decision Tree algorithm Work?**

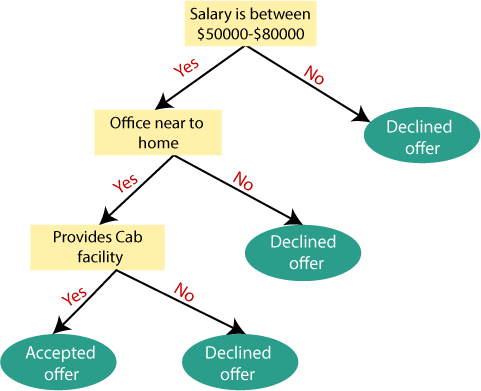
In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub- nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

* **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
* **Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).
* **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
* **Step-4:** Generate the decision tree node, which contains the best attribute.
* **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

* Information Gain
* Gini Index

**1.Information Gain:**

* Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
* It calculates how much information a feature provides us about a class.
* According to the value of information gain, we split the node and build the decision tree.
* A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1. Information Gain= Entropy(S)- [(Weighted Avg) \*Entropy (each feature)

**Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

* **S= Total number of samples**
* **P(yes)= probability of yes**
* **P(no)= probability of no**

**2. Gini Index:**

* Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm.
* An attribute with the low Gini index should be preferred as compared to the high Gini index.
* It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
* Gini index can be calculated using the below formula:

Gini Index= 1- ∑jPj2

Pruning: Getting an Optimal Decision tree

*Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.*

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree **pruning** technology used:

* **Cost Complexity Pruning**
* **Reduced Error Pruning.**

Advantages of the Decision Tree

* It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
* It can be very useful for solving decision-related problems.
* It helps to think about all the possible outcomes for a problem.
* There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

* The decision tree contains lots of layers, which makes it complex.
* It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
* For more class labels, the computational complexity of the decision tree may increase.

**CHAPTER – 2**

**LITERATURE SURVEY**

**1) A comparative study of linear and nonlinear models for aggregate retails sales forecasting**

AUTHORS: Ching Wu Chu and Guoqiang Peter Zhang

The purpose of this paper is to compare the accuracy of various linear and nonlinear models for forecasting aggregate retail sales. Because of the strong seasonal fluctuations observed in the retail sales, several traditional seasonal forecasting methods such as the time series approach and the regression approach with seasonal dummy variables and trigonometric functions are employed. The nonlinear versions of these methods are implemented via neural networks that are generalized nonlinear functional approximators. Issues of seasonal time series modelling such as deseasonalization are also investigated. Using multiple cross-validation samples, we find that the nonlinear models are able to outperform their linear counterparts in out-of- sample forecasting, and prior seasonal adjustment of the data can significantly improve forecasting performance of the neural network model. The overall best model is the neural network built on deseasonalized time series data. While seasonal dummy variables can be useful in developing effective regression models for predicting retail sales, the performance of dummy regression models may not be robust. Furthermore, trigonometric models are not useful in aggregate retail sales forecasting.

**2) Sustainable development and management in consumer electronics using soft computation**

AUTHORS: Wang, Haoxiang

Combination of Green supply chain management, Green product deletion decision and green cradle-to-cradle performance evaluation with Adaptive-Neuro-Fuzzy Inference System (ANFIS) to create a green system. Several factors like design process, client specification, computational intelligence and soft computing are analysed and emphasis is given on solving problems of real domain. In this paper, the consumer electronics and smart systems that produce nonlinear outputs are considered. ANFIS is used for handling these nonlinear outputs and offer sustainable development and management. This system offers decision making considering multiple objectives and optimizing multiple outputs. The system also provides efficient control performance and faster data transfer.

**3) Data Mining based Prediction of Demand in Indian Market for Refurbished Electronics**

AUTHORS: Suma, V., and Shavige Malleshwara Hills

There has been an increasing demand in the e-commerce market for refurbished products across India during the last decade. Despite these demands, there has been very little research done in this domain. The real-world business environment, market factors and varying customer behaviour of the online market are often ignored in the conventional statistical models evaluated by existing research work. In this paper, we do an extensive analysis of the Indian e-commerce market using data-mining approach for prediction of demand of refurbished electronics. The impact of the real-world factors on the demand and the variables are also analysed. Real-world datasets from three random e-commerce websites are considered for analysis. Data accumulation, processing and validation is carried out by means of efficient algorithms. Based on the results of this analysis, it is evident that highly accurate prediction can be made with the proposed approach despite the impacts of varying customer behaviour and market factors. The results of analysis are represented graphically and can be used for further analysis of the market and launch of new products.

**4) Forecasting Monthly Sales Retail Time Series: A Case Study**

AUTHORS: Giuseppe Nunnari, Valeria Nunnari

This paper presents a case study concerning the forecasting of monthly retail time series recorded by the US Census Bureau from 1992 to 2016. The modeling problem is tackled in two steps. First, original time series are de-trended by using a moving windows averaging approach. Subsequently, the residual time series are modeled by Non-linear Auto-Regressive (NAR) models, by using both Neuro-Fuzzy and Feed- Forward Neural Networks approaches. The goodness of the forecasting models, is objectively assessed by calculating the bias, the mae and the rmse errors. Finally, the model skill index is calculated considering the traditional persistent model as reference. Results show that there is a convenience in using the proposed approaches, compared to the reference one.

**5) Multiple Linear Regression Analysis of the Overlay Accuracy Model Zone** AUTHORS: Zone-Ching Lin, Wen-Jang Wu

The multiple linear regression method was used to analyze the overlay accuracy model and study the feasibility of using linear methods to solve parameters of nonlinear overlay equations. The methods of analysis include changing the number of sample points to derive the least sample number required for solving the accurate estimated parameter values. Besides, different high-order lens distortion parameters were ignored, and only the various modes of low-order parameters were regressed to compare their effects on the overlay analysis results. The findings indicate that given a sufficient number of sample points, the usage of multiple linear regression analysis to solve the high-order nonlinear overlay accuracy model containing seventh-order lens distortion parameters is feasible. When the estimated values of low-order overlay distortion parameters are far greater than those of high-order lens distortion parameters, excellent overlay improvement can still be obtained even if the high-order lens distortion parameters are ignored. When the overlay at the four corners of image field obviously exceeds that near the center of image field, it is found, through simulation, that the seventh-order parameters overlay model established in this paper has to be corrected by high-order lens distortion parameters to significantly improve the overlay accuracy.

**DATA MODULES**

**Data Collection Module**

**Data Pre-Processing module**

**Evaluation Module**

**Prediction Module**

**Data Collection Module:**

In this module, the raw data collected by a big mart will be pre-processed for missing data, anomalies and outlier. An algorithm will then be trained to construct a model on that data. It is a system in which three functions are combined. It is used to extract and transform the data from one database into an appropriate format.

**Data Pre-Processing Module:**

The dataset used is Big Mart sales result and there are total 9 attributes such as Store, StoreType, Assortment, CompetitionDistance, CompetitionOpenSinceMonth, CompetitionOpenSinceYear, Promo2, Promo2SinceWeek, Promo2SinceYear, PromoInterval. Item Outlet Sales is the target variable and the other remaining attributes are independent variable. The pre-processing of data is a method for preparing and adapting raw data to a model of learning. This is the first and significant step to construct a machine learning model. Real-world data generally contain noise, missing values and may not be used in an unusable format especially for machine learning models .

**Evaluation Module:**

Evaluation of the model is the vital part of creating an efficient machine learning model. Therefore it is important to create a model and get suggestions from it in terms of metrics. It will take and continue until we achieve good accuracy according to the value obtained from metric improvements. Evaluation metrics describe one model’s results. The ability to distinguish between model outcomes is an important feature of the evaluation metrics. Here, we used Root Mean Squared Error(RMSE)metric for evaluation process,

**Prediction module:**

We propose a predictive model using XG boost Regressor technique for predicting the sales of a company like Big Mart and found that the model produces better performance as compared to existing models

**CHAPTER - 3**

**SYSTEM REQUIREMENTS SPECIFICATIONS**

**3.1 FUNCTIONAL REQUIREMENTS:**

Functional requirements outline the specific functions or features that a system or project must perform to satisfy the needs of its users. For a project focused on predicting Big Mart sales using machine learning algorithms, the functional requirements may include:

1. **Data Collection and Integration**:
   * The system should collect relevant data from various sources, including sales records, store information, product details, and external factors such as weather data.
   * It should integrate and preprocess the collected data to create a unified dataset suitable for analysis and modelling.
2. **Data Exploration and Visualization**:
   * The system should provide tools for exploring and visualizing the dataset to gain insights into the distribution of features, relationships between variables, and potential patterns or trends.
   * It should support various types of visualizations such as histograms, scatter plots, line charts, and heatmaps to facilitate data exploration.
3. **Model Training and Evaluation**:
   * The system should offer functionality for training machine learning models using the prepared dataset. This includes selecting appropriate algorithms, tuning hyperparameters, and optimizing model performance.
   * It should provide options for evaluating model performance using relevant metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), or R-squared (R2) score.
4. **Prediction and Forecasting**:
   * The system should enable users to make predictions and forecasts of future sales based on trained machine learning models.
   * It should allow users to input new data or select specific scenarios to generate predictions for different time periods, store locations, or product categories.
5. **Accuracy and Confidence Intervals**:
   * The system should calculate and display the accuracy of sales predictions on both training and testing datasets.
   * It should provide confidence intervals or prediction intervals to quantify the uncertainty associated with the predicted sales values.
6. **Model Deployment and Integration**:
   * The system should support deploying trained models into production environments for real-time or batch predictions.
   * It should provide integration options with other systems or applications used by Big Mart stakeholders, such as inventory management systems or sales forecasting tools.
7. **User Interface and Interactivity**:
   * The system should have a user-friendly interface that allows users to interact with data, models, and predictions easily.
   * It should offer features for filtering data, adjusting model parameters, and visualizing results dynamically.
8. **Performance Monitoring and Maintenance**:
   * The system should include mechanisms for monitoring model performance over time and detecting potential issues such as concept drift or data drift.
   * It should support ongoing maintenance tasks such as retraining models with updated data, improving model accuracy, and addressing any feedback or issues from users.

**Examples of functional requirements:**

1) Authentication of user whenever he/she logs into the system

2) System shutdown in case of a cyber-attac

3) A verification email is sent to user whenever he/she register for the first time on some software system.

**3.2 NON-FUNCTIONAL REQUIREMENTS**

Non-functional requirements specify criteria that describe how a system should behave or perform, rather than specific functions it must fulfill. For a project focused on predicting Big Mart sales using machine learning algorithms, some non-functional requirements may include:

1. **Performance:**
   * The system should be able to handle large volumes of data efficiently and process predictions within a reasonable time frame.
   * It should be scalable to accommodate increases in data volume or user traffic without significant degradation in performance.
2. **Accuracy:**
   * The accuracy of sales predictions generated by the system should meet predefined thresholds or standards.
   * The system should provide mechanisms for continuously monitoring and improving prediction accuracy over time.
3. **Reliability:**
   * The system should be reliable and available for use whenever needed, with minimal downtime or disruptions.
   * It should include failover mechanisms and backups to ensure data integrity and continuity of service.
4. **Security:**
   * The system should adhere to security best practices to protect sensitive data, such as sales records and customer information.
   * It should implement authentication, authorization, and encryption mechanisms to prevent unauthorized access or data breaches.
5. **Scalability:**
   * The system should be designed to scale horizontally or vertically to accommodate increasing computational and storage requirements.
   * It should be able to handle concurrent user access and support a growing number of users or concurrent prediction requests.
6. **Usability:**
   * The system should have an intuitive and user-friendly interface that is easy to navigate and understand.
   * It should provide clear instructions, error messages, and feedback to users to facilitate their interactions with the system.
7. **Maintainability:**
   * The system should be modular and well-documented, making it easy to maintain, update, and extend in the future.
   * It should adhere to coding standards and best practices to ensure code quality and readability.
8. **Compatibility:**
   * The system should be compatible with a variety of devices, browsers, and operating systems commonly used by Big Mart stakeholders.
   * It should adhere to industry standards and interoperability requirements to facilitate integration with other systems or tools.
9. **Performance Metrics:**
   * Define specific performance metrics to measure system responsiveness, throughput, and resource utilization.
   * Establish benchmarks and targets for performance improvement based on these metrics.

**3.3 HARDWARE SPECIFICATIONS**

1. **Processor (CPU):**
   * Multi-core processor with at least 4 cores for parallel processing.
   * Intel Core i7 or AMD Ryzen 7 series processors are suitable for moderate to high computational workloads.
   * Higher clock speeds (e.g., 3.0 GHz or higher) can expedite data processing and model training.

**2. Memory (RAM):**

* + Minimum of 8 GB DDR4 RAM for handling moderate-sized datasets and model training.
  + Consider 16 GB or more for larger datasets or memory-intensive algorithms.
  + Ensure sufficient memory bandwidth for faster data access and manipulation.

**3.STORAGE:**

* Solid State Drive (SSD) with sufficient capacity for storing datasets, model files, and software
* SSDs offer faster read/write speeds compared to traditional Hard Disk Drives (HDDs), reducing data loading times and improving overall system responsiveness.
* Aim for at least 256 GB SSD for storage. Larger capacities (e.g., 512 GB or 1 TB) are recommended for storing large datasets and model checkpoints.

**4.Internet Connectivity**:

* + Stable high-speed internet connection for accessing online resources, downloading datasets, and utilizing cloud-based services (if applicable).
  + Broadband or fiber-optic internet with sufficient bandwidth for faster data transfer and online collaboration.

**3.4 SOFTWARE SPECIFICATIONS**

1. **Windows:**
   * Windows is a widely used operating system with good compatibility with most machine learning libraries and frameworks.
   * It offers a user-friendly interface and extensive software support for development, data analysis, and visualization tasks.
   * Windows is suitable for developers comfortable with the Windows environment and tools like Visual Studio for development.

**2.Python 3.6**

* Python 3.6 is a widely used version with many libraries and frameworks available for machine learning projects. You can use it to develop server-side scripts, train machine learning models, preprocess data, and deploy applications.

**3.IDE**

* Jupyter Notebook is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text.
* It's well-suited for exploratory data analysis, prototyping machine learning models, and documenting research findings.
* Jupyter Notebook supports multiple programming languages, including Python, R, and Julia, making it versatile for various data science tasks.

**4.Libraries used**

* **NumPy** Fundamental library for numerical computing in Python, providing support for multidimensional arrays and mathematical functions.
* **Pandas** Data manipulation library for working with structured data, offering data structures and functions for data cleaning, transformation, and analysis.
* **matplotlib / seaborn** Visualization libraries for creating plots and charts to visualize model performance, feature importance, and data distributions**.**

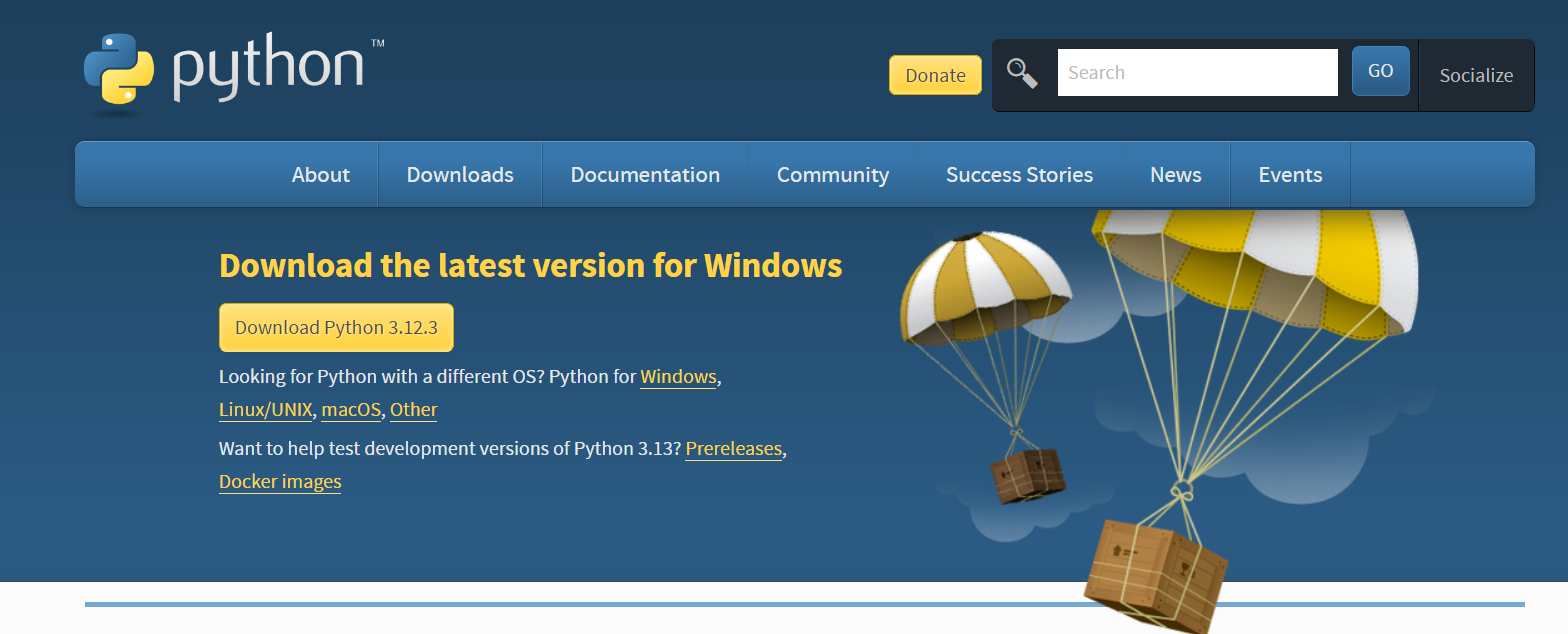
**5.Framework**

* **Flask** is a lightweight and flexible web framework for Python, suitable for creating RESTful APIs to serve machine learning models You can use Flask to wrap your trained model with a simple API endpoint, allowing clients to send input data and receive predictions

**SOFTWARE INSTALLATION FOR MACHINE LEARNING PROJECT:**

**3.5 Installing Python:**

1. To download install and Python visit the official Python https://www.python.org/downloads/ and choose your version.



2. Once the download is complete, run the exe for install Python. Now click on Install Now.

3. You can see Python installing at this point.

4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

**CHAPTER - 4**

**SYSTEM ANALYSIS**

**4.1 PURPOSE**

The purpose of this document is predicting big mart sales using machine learning algorithms. In detail, this document will provide a general description of our project, including user requirements, product perspective, and overview of requirements, general constraints. In addition, it will also provide the specific requirements and functionality needed for this project - such as interface, functional requirements and performance requirements.

**4.2 SCOPE**

The scope of this SRS document persists for the entire life cycle of the project. This document defines the final state of the software requirements agreed upon by the customers and designers. Finally at the end of the project execution all the functionalities may be traceable from the SRS to the product. The document describes the functionality, performance, constraints, interface and reliability for the entire life cycle of the project.

**4.3 EXISTING SYSTEM:**

* Auto-Regressive Integrated Moving Average, (ARMA) Auto-Regressive Moving Average, have been utilized to develop a few deals forecast standards. Be that as it may, deals anticipating is a refined issue and is influenced by both outer and inside factors, and there are two significant detriments to the measurable technique as set out in A. S. Weigend et A mixture occasional quantum relapse approach and (ARIMA) Auto-Regressive Integrated Moving Average way to deal with every day food deals anticipating were recommend by N. S. Arunraj and furthermore found that the exhibition of the individual model was moderately lower than that of
* E. Hadavandi utilized the incorporation of “Genetic Fuzzy Systems (GFS)” and information gathering to conjecture the deals of the printed circuit board. In their paper, K-means bunching delivered K groups of all information records. At that point, all bunches were taken care of into autonomous with a data set tuning and rule-based extraction ability.
* Perceived work in the field of deals gauging was done by P.A. Castillo, Sales estimating of new distributed books was done in a publication market the executives setting utilizing computational techniques. “Artificial neural organizations” are additionally utilized nearby income estimating. Fluffy Neural Networks have been created with the objective of improving prescient effectiveness, and the Radial “Base Function Neural Network (RBFN)” is required to have an incredible potential for anticipating deals.

**DISADVANTAGES OF EXISTING SYSTEM:**

* Complex models like neural networks are overkill for simple problems like regression.
* Existing system models prediction analysis which gives less accuracy.
* Forecasting methods and applications contains Lack of Data and short life cycles. So some of the data like historical data, consumer-oriented markets face uncertain demands, can be prediction for accurate result.

**4.4 PROPOSED SYSTEM:**

* The objective of this proposed system is to predict the future sales from given data of the previous year's using Decision Tree Regression
* Another objective is to conclude the best model which is more efficient and gives fast and accurate result by using Decision Tree Regression.
* To find out key factors that can increase their sales and what changes could be made to the product or store's characteristics.
* Experts also shown that a smart sales forecasting program is required to manage vast volumes of data for business organizations.
* We are predicting the accuracy for Decision Tree Regression. Our predictions help big marts to refine their methodologies and strategies which in turn helps them to increase their profit. The results predicted will be very useful for the executives of the company to know about their sales and profits. This will also give them the idea for their new locations or Centre‟s of Bigmart

**ADVANTAGES OF PROPOSED SYSTEM:**

* Business assessments are based on the speed and precision of the methods used to analyze the results. The Machine Learning Methods presented in this research paper should provide an effective method for data shaping and decision-making.
* New approaches that can better identify consumer needs and formulate marketing plans will be implemented.
* The outcome of machine learning algorithm will help to select the most suitable demand prediction algorithm and with the aid of which BigMart will prepare its marketing campaigns.

**CHAPTER 5**

**INTRODUCTION TO PYTHON**

**5.1 WHAT IS A SCRIPT?**

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

Scripts are reusable

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

Scripts are editable

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

You will need a text editor

Just about any text editor will suffice for creating Python script files.

You can *use Microsoft Notepad, Microsoft WordPad, Microsoft Word*, or just about any word processor if you want to.

**5.2 Difference between a script and a program**

Script:

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

Program:

The program has an executable form that the computer can use directly to execute the instructions.

The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

Python

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don't know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to "big name" languages. Hopefully I can explain it for you.

Python concepts

If you're not interested in the how's and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it's a great one to start programming with.

* Open-source general-purpose language.
* Object Oriented, Procedural, Functional
* Easy to interface with C/ObjC/Java/Fortran
* Easy-is to interface with C++ (via SWIG)
* Great interactive environment
* Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* Python is Interpreted Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* Python is Interactive you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* Python is Object-Oriented Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* Python is a Beginner's Language - Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to www browsers to games.

**5.3 History of Python**

Python was developed by Guido van Possum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and UNIX shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Possum still holds a vital role in directing its progress.

**5.4 Python Features**

Python's features include -

* Easy-to-learn- Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* Easy-to-read-Python code is more clearly defined and visible to the eyes.
* Easy-to-maintain Python's source code is fairly easy-to-maintained.
* A broad standard library Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* Interactive Mode Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* Portable Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* Extendable you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* Databases Python provides interfaces to all major commercial databases.
* GUI Programming Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* Scalable Python provides a better structure and support for large programs than shell scripting.
* Apart from the above-mentioned features, Python has a big list of good features, few are listed below-
* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* IT supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**5.5 Dynamic vs. Static**

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of "thing" each data value is.

For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a "float" type.

This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point.

If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn't require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program.

With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).

If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double.

With Python, it doesn't matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

For example, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it's more accurate to keep track of decimals so it automatically calculates the result as a floating point number

**5.5.1 Variables**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

**5.5.2 Standard Data Types**

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types-

* Numbers
* String
* List
* Tuple
* Dictionary

**5.5.3 Python Numbers**

Number data types store numeric values. Number objects are created when you assign a value to them

**5.5.4 Python Strings**

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

**5.5.5 Python Lists**

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type. The values stored in a list can be accessed using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (\*) is the repetition operator.

**5.4.6 Python Tuples**

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses. The main differences between lists and tuples are: Lists are enclosed in brackets ([]) and their elements and size can be changed, while tuples are enclosed in parentheses (()) and cannot be updated. Tuples can be thought of as read-only lists.

**5.5.7 Python Dictionary**

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object. Dictionaries are enclosed by curly braces ({}) and values can be assigned and accessed using square braces ([]).

**5.5.8 Different modes in python**

Python has two basic modes: normal and interactive.

The normal mode is the mode where the scripted and finished pie files are run in the Python interpreter.

Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

**5.6 Python Libraries**

1. Requests. The most famous http library written by Kenneth remits. It's a must have for every python developer.

2. Scrappy. If you are involved in web scraping then this is a must have library for you. After using this library you won't use any other.

3. Python. A guy toolkit for python. I have primarily used it in place of tinder. You will really love it.

4. Pillow. A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.

5. SQL Alchemy. A database library. Many love it and many hate it. The choice is yours.

6. Beautiful Soup. I know it's slow but this xml and html parsing library is very useful for beginners.

7. Twisted. The most important tool for any network application developer. It has a very beautiful ape and is used by a lot of famous python developers.

8. Numbly. How can we leave this very important library? It provides some advance math functionalities to python.

9. Skippy. When we talk about numbly then we have to talk about spicy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

10. Matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyser.

11. Pygmy. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.

12. Piglet. A 3d animation and game creation engine. This is the engine in which the famous python port of mine craft was made

13. Pit. A GUI toolkit for python. It is my second choice after python for developing GUI's for my python scripts.

14. Pit. Another python GUI library. It is the same library in which the famous Bit torrent client is created.

15. Scaly. A packet sniffer and analyser for python made in python.

16. Pywin32. A python library which provides some useful methods and classes for interacting with windows.

17. Notch. Natural Language Toolkit I realize most people won't be using this one, but it's generic enough. It is a very useful library if you want to manipulate strings. But its capacity is beyond that. Do check it out.

18. Nose. A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.

19. Simply. Simply can do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.

20. I Python. I just can't stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

**5.7 NumPy**

Numpy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In numbly dimensions are called axes. The number of axes is rank.

* Offers Matlab-ish capabilities within Python
* Fast array operations
* 2D arrays, multi-D arrays, linear algebra etc.

**5.8 Matplotlib**

High quality plotting library.

Python class and objects

These are the building blocks of OOP. Class creates a new object. This object can be anything. whether an abstract data concept or a model of a physical object, e.g. a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently "the big thing" in most programming languages. Hence, there are several chapters dedicated to OOP later in the book.

The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something.

Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest,

it took me several months to get a handle on objects.

When I first learned C and C++, I did great; functions just made sense for me.

Having messed around with BASIC in the early '90s, I realized functions were just like subroutines so there wasn't much new to learn.

However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered.

Once you learn OOP, you'll realize that it's actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it's not mandatory to use objects in your code in a way that works best; maybe you don't need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want.

As you've already seen, Python can do just fine with functions. Unlike languages such as Java, you aren't tied down to a single way of doing things, you can mix functions and classes as necessary in the same program. This lets you build the code

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here's a brief list of Python OOP ideas:

* The class statement creates a class object and gives it a name. This creates a new namespace.
* Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: ClassName.Attribute.
* Class attributes export the state of an object and its associated behavior. These attributes are shared by all instances of a class.
* Calling a class (just like a function) creates a new instance of the class.
* This is where the multiple copies part comes in.
* Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.
* Using the term self identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

Inheritance

First off, classes allow you to modify a program without really making changes to it.

To elaborate, by sub classing a class, you can change the behaviour of the program by simply adding new components to it rather than rewriting the existing components.

As we've seen, an instance of a class inherits the attributes of that class.

However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses.

The subclasses can override the logic in a superclass, allowing you to change the behaviour of your classes without changing the superclass at all.

Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behavior closer to Python's object model and the object interfaces are more consistent to Python's built-in objects, hence overloading is easier to learn and use.

User-made classes can override nearly all of Python's built-in operation methods

Exceptions

I've talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program's flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g. trying to open a file that doesn't exist, or when the program reaches a marker, such as the completion of a loop.

Exceptions, by definition, don't occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python.

Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

* Accessing a non-existent dictionary key will raise a Key Error exception.
* Searching a list for a non-existent value will raise a Value Error exception
* Calling a non-existent method will raise an Attribute Error exception.
* Referencing a non-existent variable will raise a Name Error exception.
* Mixing data types without coercion will raise a Type Error exception.

One use of exceptions is to catch a fault and allow the program to continue working, we have seen this before when we talked about files.

This is the most common way to use exceptions. When programming with the Python command line interpreter, you don't need to worry about catching exceptions.

Your program is usually short enough to not be hurt too much if an exception occurs.

Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem.

However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception.

It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don't put any overhead on the code processing.

Because exceptions aren't supposed to happen very often, they aren't processed until they occur.

Exceptions can be thought of as a special form of the if/elf statements. You can realistically do the same thing with if blocks as you can with exceptions.

However, as already mentioned, exceptions aren't processed until they occur; if blocks are processed all the time.

Proper use of exceptions can help the performance of your program.

The more infrequent the error might occur, the better off you are to use exceptions; using if blocks requires Python to always test extra conditions before continuing.

Exceptions also make code management easier: if your programming logic is mixed in with error- handling if statements, it can be difficult to read, modify, and debug your program.

User-Defined Exceptions

I won't spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.

You probably won't have to do this very often but it's nice to have the option when necessary.

However, before making your own exceptions, make sure there isn't one of the built-in exceptions that will work for you.

They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

Making your own exceptions involves object-oriented programming, which will be covered in the next chapter

To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g. making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class.

To make a custom exception, simply inherit the base exception and define what it will do.

Python modules

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. preter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

**5.9 Testing code**

As indicated above, code is usually developed in a file using an editor.

To test the code, import it into a Python session and try to run it.

Usually there is an error, so you go back to the file, make a correction, and test again.

This process is repeated until you are satisfied that the code works. T

His entire process is known as the development cycle.

There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.

This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

**5.10 Functions in Python**

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function.

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task.

To carry out that specific task, the function might or might not need multiple inputs. When the task is carved out, the function can or cannot return one or more values.

There are three types of functions in python:

Help (), min (), print ().

**5.11 Python Namespaces:**

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python:

* global names of a module
* local names in a function or method invocation.
* built-in names: this namespace contains built-in functions (e.g. abs(), camp(), ...) and built-in exception names

**5.12 Garbage collection**

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

**5.13 Python XML Parser**

XML is a portable, open source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML.

This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone.

XML Parser Architectures and APIs the Python standard library provides a minimal but useful set of interfaces to work with XML.

The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces.

Simple API for XML SAX: Here, you register callbacks for events of interest and then let the parser proceed through the document.

This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

Document Object Model DOM API: This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree represent all the features of an XML document. based form to

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files.

SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

**5.14 Python Web Frameworks**

A web framework is a code library that makes a developer's life easier when building reliable, scalable and maintainable web applications.

WHY ARE WEB FRAMEWORKS USEFUL?

Web frameworks encapsulate what developers have learned over the past twenty years while programming sites and applications for the web. Frameworks make it easier to reuse code for common HTTP operations and to structure projects so other developers with knowledge of the framework can quickly build and maintain the application.

Common web framework functionality

Frameworks provide functionality in their code or through extensions to perform common operations required to run web applications. These common operations include:

1. URL routing

2. HTML, XML, JSON, and other output format tinplating

3. Database manipulation

4. Security against Cross-site request forgery (CSRF) and other attacks

5. Session storage and retrieval

Not all web frameworks include code for all of the above functionality. Frameworks fall on the spectrum from executing a single use case to providing every known web framework feature to every developer. Some frameworks take the "batteries-included" approach where everything possible comes bundled with the framework while others have a minimal core package that is amenable to extensions provided by other packages.

COMPARING WEB FRAMEWORKS

There is also a repository called compare-python-web-frameworks where the same web application is being coded with varying Python web frameworks, tinplating engines and object.

WEB FRAMEWORK RESOURCES

* When you are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing.
* Frameworks is a really well done short video that explains how to choose between web frameworks. The author has some particular opinions about what should be in a framework. For the most part I agree although I've found sessions and database ORMs to be a helpful part of a framework when done well.
* What is a web framework? Is an in-depth explanation of what web frameworks are and their relation to web servers?
* Jingo vs. Flash vs. Pyramid: Choosing a Python web framework contains background information and code comparisons for similar web applications built in these three big Python frameworks.
* This fascinating blog post takes a look at the code complexity of several Python web frameworks by providing visualizations based on their code bases.
* Python's web frameworks benchmarks is a test of the responsiveness of a framework with encoding an object to JSON and returning it as a response as well as retrieving data from the database and rendering it in a template. There were no conclusive results but the output is fun to read about nonetheless.
* What web frameworks do you use and why are they awesome? Is a language agnostic Reedit discussion on web frameworks? It's interesting to see what programmers in other languages like and dislike about their suite of web frameworks compared to the main Python frameworks.
* This user-voted question & answer site asked "What are the best general purpose Python web frameworks usable in production?" The votes aren't as important as the list of the many frameworks that are available to Python developers.

WEB FRAMEWORKS LEARNING CHECKLIST

1. Choose a major Python web framework (Jingo or Flask are recommended) and stick with it. When you're just starting it's best to learn one framework first instead of bouncing around trying to understand every framework.
2. Work through a detailed tutorial found within the resource’s links on the framework's page.
3. Study open-source examples built with your framework of choice so you can take parts of those projects and reuse the code in your application.

**CHAPTER-6**

**SYSTEM STUDY**

**System study** also known as system analysis is a comprehensive examination of an existing or proposed system to understand its components, functionality, and performance. It involves analysing the requirements, constraints, and objectives of the system to determine the feasibility of implementing or improving it.

**System study** for a machine learning project predicting Big Mart sales involves analysing the requirements, constraints, and objectives of the project to design a system that meets the stakeholders' needs effectively. Here's an outline of key aspects to consider in the system study:

**1.Project Objectives and Scope**:

* + Define the main objectives of the project, such as improving sales forecasting accuracy, optimizing inventory management, or identifying factors influencing sales performance.
  + Clarify the scope of the project, including the specific tasks to be accomplished, the data sources to be used, and the expected outcomes.

**2**.**Requirements Gathering:**

* Gather requirements from stakeholders, including business managers, data analysts, and end-users, to understand their needs and expectations
* Define the functional and non-functional requirements of the system, such as data sources, features, performance criteria, and integration points.

**3**.**Feasibility Assessment:**

* Assess the technical, economic, and operational feasibility of the project to determine if it is viable and achievable.
* Evaluate the availability of data, resources, and expertise needed to develop and implement the predictive model.

**4.Stakeholder Analysis**:

* + Identify the stakeholders involved in the project, including business managers, data analysts, IT professionals, and end-users.
  + Understand the needs, expectations, and roles of each stakeholder group in the project.

**5.Data Requirements:**

* + Determine the types of data required for predicting Big Mart sales, such as historical sales data, store information, product attributes, and external factors like demographics or economic indicators.
  + Assess the availability, quality, and relevance of existing data sources, and identify any gaps or limitations that need to be addressed.

**6.System Architecture:**

* + Design the overall architecture of the system, including the components, modules, and interfaces required to collect, preprocess, analyze, and visualize data.
  + Consider the scalability, flexibility, and maintainability of the architecture to accommodate future growth and changes in requirements.

**7.Data Collection and Integration:**

* + Plan the process for collecting and integrating data from various sources, such as internal databases, external APIs, or third-party vendors.
  + Define data extraction, transformation, and loading (ETL) workflows to preprocess and standardize the data for analysis.

**8.Model Development and Evaluation:**

* + Select appropriate machine learning algorithms and techniques for predicting Big Mart sales based on the project objectives and data characteristics.
  + Define evaluation metrics and validation strategies to assess the performance of the models and compare different approaches.

**9.User Interface and Interaction:**

* + Design the user interface for accessing and interacting with the system, considering the needs and preferences of different user groups.
  + Determine the features, visualizations, and functionalities required to support data exploration, model training, and result interpretation.

**10.Integration and Deployment:**

* + Plan how the machine learning models will be integrated into the system and deployed for real-world use.
  + Consider deployment options such as cloud-based solutions, on-premises servers, or edge devices, and ensure compatibility with existing infrastructure and technologies.

**11.Security and Privacy:**

* + Identify potential security risks and privacy concerns associated with handling sensitive data such as sales records or customer information.
  + Implement security measures such as data encryption, access controls, and compliance with relevant regulations (e.g., GDPR, CCPA) to protect data integrity and confidentiality.

**12.Testing and Validation:**

* + Develop a testing strategy to validate the functionality, performance, and accuracy of the system under different scenarios and use cases.
  + Conduct thorough testing, including unit testing, integration testing, and user acceptance testing, to ensure the system meets the specified requirements.

**13.Documentation and Training:**

* + Document the system architecture, design decisions, data flows, and implementation details to facilitate future maintenance and knowledge transfer.
  + Provide training and support for users and stakeholders to familiarize them with the system functionality, features, and best practices.

**CHAPTER - 7**

**SYSTEM DESIGN**

**7.1 INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**7.2 OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system‟s relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

• Convey information about past activities, current status or projections of the

• Future.

• Signal important events, opportunities, problems, or warnings.

• Trigger an action.

• Confirm an action

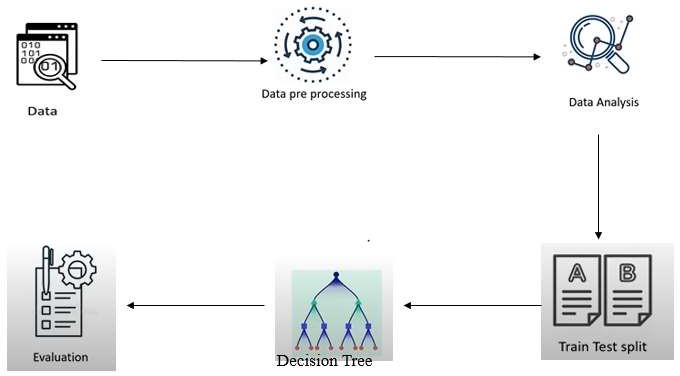
**7.3 DATA FLOW DIAGRAM**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



**7.4 UML DIAGRAMS**

* UML stands for Unified Modelling Language. UML is a standardized general- purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.
* The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.
* The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.
* The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.
* The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

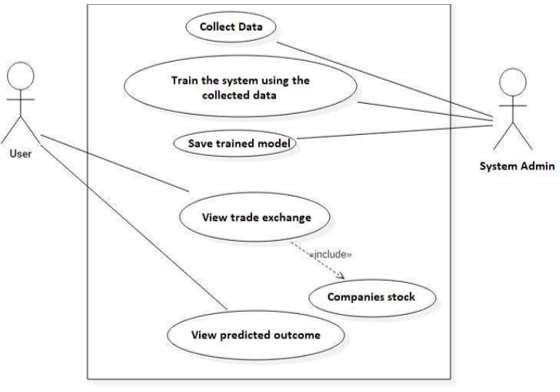
**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

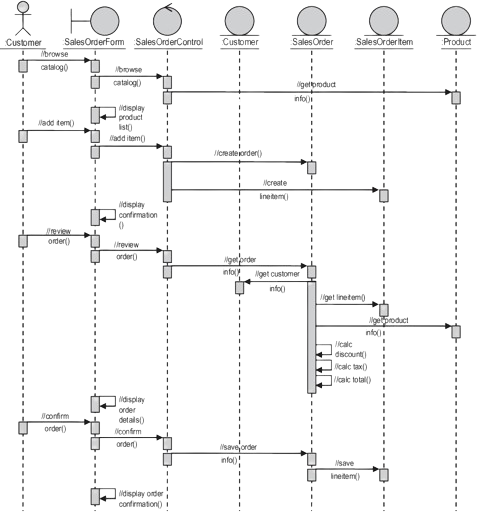
**7.5 USE CASE DIAGRAM**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**7.6 SEQUENCE DIAGRAM**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

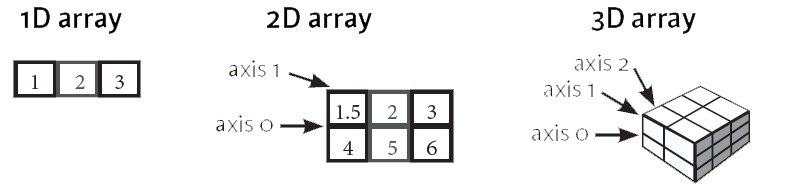


**CHAPTER - 8**

**Module Implementations**

**8.1 NUMPY**

**Numpy** is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.  
Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.



**8.2 PANDAS**

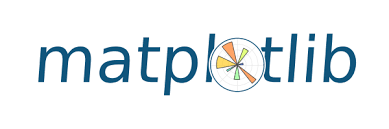
Pandas is a powerful and open-source Python library. The Pandas library is used for data manipulation and analysis. Pandas consist of data structures and functions to perform efficient operations on data.



**FUNCTIONS OF PANDAS:**

* Data set cleaning, merging, and joining.
* Easy handling of missing data (represented as NaN) in floating point as well as non-floating-point data.
* Columns can be inserted and deleted from Data Frame and higher-dimensional objects.
* Powerful group by functionality for performing split-apply-combine operations on data sets.
* Data Visualization.

**8.3 MATPLOTLIB**

****

Matplotlib is a powerful plotting library in Python used for creating static, animated, and interactive visualizations. Matplotlib’s primary purpose is to provide users with the tools and functionality to represent data graphically, making it easier to analyze and understand. It was originally developed by John D. Hunter in 2003 and is now maintained by a large community of developers.

**Key Features of Matplotlib:**

1. **Versatility:** Matplotlib can generate a wide range of plots, including line plots, scatter plots, bar plots, histograms, pie charts, and more.
2. **Customization**: It offers extensive customization options to control every aspect of the plot, such as line styles, colors, markers, labels, and annotations.
3. **Integration with NumPy:** Matplotlib integrates seamlessly with NumPy, making it easy to plot data arrays directly.
4. **Publication Quality:** Matplotlib produces high-quality plots suitable for publication with fine-grained control over the plot aesthetics.
5. **Extensible:** Matplotlib is highly extensible, with a large ecosystem of add-on toolkits and extensions like Seaborn, Pandas plotting functions, and Basemap for geographical plotting.
6. **Cross-Platform**: It is platform-independent and can run on various operating systems, including Windows, macOS, and Linux.
7. **Interactive Plots:** Matplotlib supports interactive plotting through the use of widgets and event handling, enabling users to explore data dynamically.

**8.4 SEABORN**

****

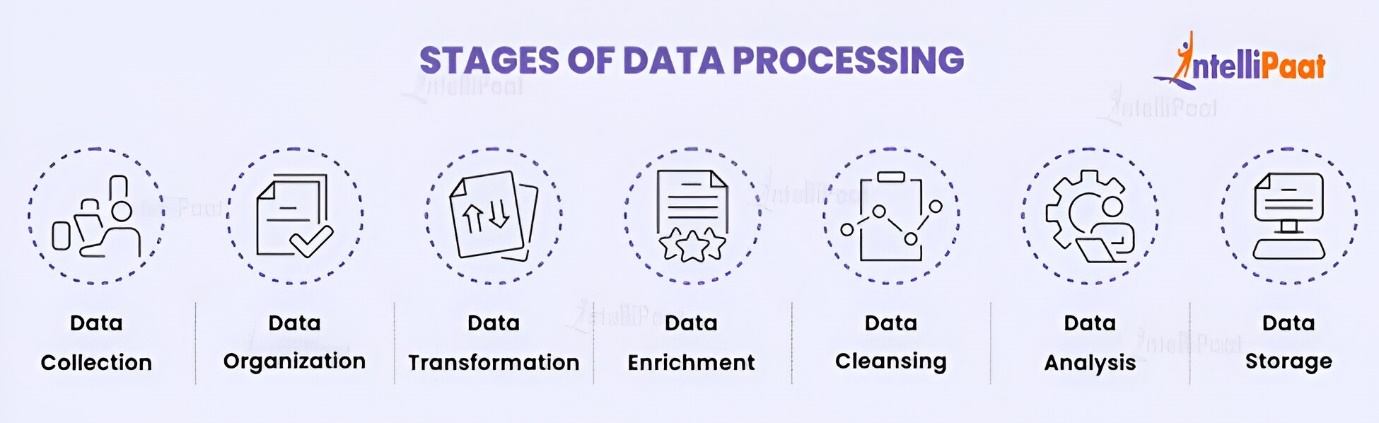
Seaborn is a powerful data visualization library in Python that provides an intuitive and easy-to-use interface for creating informative statistical graphics.

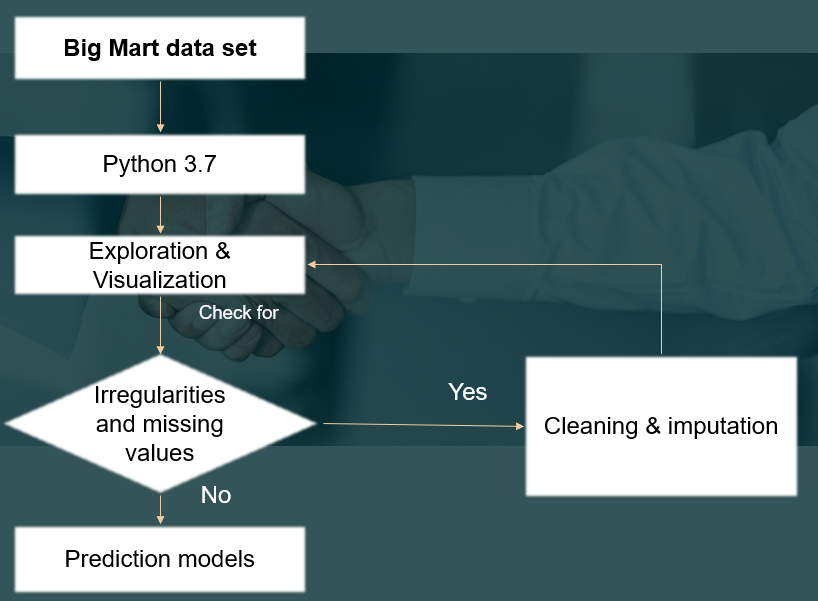
**Objective of Seaborn library:**

Seaborn library aims to make a more attractive visualization of the central part of understanding and exploring data. It is built on the core of the matplotlib library and also provides dataset-oriented APIs.

Seaborn is also closely integrated with the Panda's data structures, and with this, we can easily jump between the various different visual representations for a given variable to better understand the provided dataset.

**WORK FLOW OF THE PROJECT:**

****



**8.5 Data Collection**

This is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.

There are several techniques to collect the data, like web scraping, manual interventions and etc. Predictive Analysis for Big Mart Sales Using Machine Learning Algorithms

Data set Link: <https://www.kaggle.com/shivan118/big-mart-sales-prediction-datasets>

**8.6 Dataset:**

The dataset consists of 8523 individual data. There are 12 columns in the dataset, which are described below.

**1.** **ItemIdentifier** ----

**2.ItemWeight** ---- Unique Weight product of ID product

**3**.**ItemFatContent** ---- Whether the product is low fat or not

**4**.**ItemVisibility** ---- The % of the total display area of all products in a store allocated to

**5**.**ItemType** ----

**6**.**ItemMRP** ---- The the category to particular which the product product belongs Maximum Retail Price (list price) of the product

**7**.**OutletIdentifier** ---- Unique store ID

**8**.**OutletEstablishmentYear** ---- The year in which the store was established

**9.OutletSize** ---- The size of the store in terms of ground area covered

**10.OutletLocationType** ---- The type of city in which the store is located

**11**.\***OutletType** ---- Whether the outlet is just a grocery store or some sort of supermarket

**12.ItemOutletSales** ---- sales of the product in t particular store. This is the outcome variable to be predicted.

**8.7 Data Preparation:**

Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.)

Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data

Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis

Split into training and evaluation sets

**8.8 Model Selection:**

We used decision tree regression machine learning algorithm, we got a accuracy of 95.7% on test set so we implemented this algorithm.

**Decision tree regression**

Decision Tree is a decision-making tool that uses a flowchart-like tree structure or is a model of decisions and all of their possible results, including outcomes, input costs, and utility. Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables. The branches/edges represent the result of the node and the nodes have either

Conditions [Decision Nodes]

Result [End Nodes]

The branches/edges represent the truth/falsity of the statement and take makes a decision based on that in the example below which shows a decision tree that evaluates the smallest of three numbers.

**Decision tree regression**

Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values.

**8.9 Analyze and Prediction:**

In the actual dataset, we chose only 9 features:

**1.ItemWeight** ---- Weight of product

**2.ItemFatContent** ---- Weight Whether the product of is low fat product or not

**3.ItemVisibility** ---- The % of the total display area of all products in a store allocated to

**4.ItemType** ---- The the category to particular which the product product belongs

**5.ItemMRP** ---- Maximum Retail Price (list price) of the product

**6.OutletEstablishmentYear** ---- The year in which the store was established

**7.OutletSize** ---- The size of the store in terms of ground area covered

**8.OutletLocationType** ---- The type of city in which the store is located

**9.\*OutletType** ---- Whether the outlet is just a grocery store or some sort of supermarket

**8.10 Accuracy on test set:**

We got an accuracy of 95.80% on test set.

**8.11 Saving the Trained Model:**

Once you’re confident enough to take your trained and tested model into the production ready environment, the first step is to save it into a .h5 (or) .pkl file using a library like pickle

Make sure you have pickle installed in your environment.

Next, let’s import the module and dump the model into .pkl

**CHAPTER 9**

**SYSTEM TESTING**

**System testing** for a machine learning project predicting Big Mart sales involves evaluating the functionality, performance, and reliability of the entire system, including data pipelines, machine learning models, and user interfaces.

1. **Data Integrity Testing**:
   * Ensure the integrity and quality of the data used for training and testing the machine learning models.
   * Validate that data preprocessing steps, such as data cleaning, feature engineering, and normalization, are performed correctly and consistently.
   * Verify that the input data format and structure are compatible with the requirements of the machine learning algorithms.
2. **Model Evaluation Testing**:
   * Evaluate the performance of the machine learning models using appropriate evaluation metrics, such as accuracy, precision, recall, F1-score, and ROC-AUC.
   * Conduct cross-validation or holdout validation to assess the generalization ability of the models and detect overfitting or underfitting.
   * Compare the performance of different models and parameter configurations to identify the best-performing model for predicting Big Mart sales.
3. **Integration Testing**:
   * Test the integration between different components of the system, including data pipelines, machine learning models, and user interfaces.
   * Verify that data flows smoothly through the system, from data ingestion and preprocessing to model training, prediction, and result visualization.
   * Validate that data transformations, feature extraction, and model inference are performed accurately and consistently.
4. **User Interface Testing**:
   * Validate the usability and functionality of the user interface components, such as dashboards, reports, and interactive visualizations.
   * Test user interactions, such as data filtering, sorting, and drill-down, to ensure they are intuitive and responsive.
   * Verify that the user interface displays relevant and accurate information, such as sales trends, predictions, and performance metrics.
5. **Performance Testing**:
   * Assess the performance of the system under different load conditions, including varying data volumes, concurrent user interactions, and computational resource usage.
   * Measure the system's response time, throughput, and resource utilization to identify potential bottlenecks or performance issues.
   * Optimize the system's performance by tuning parameters, optimizing algorithms, and scaling resources as needed.
6. **Security Testing**:
   * Conduct security testing to identify and address potential vulnerabilities in the system, such as data breaches, unauthorized access, or malicious attacks.
   * Implement security measures, such as data encryption, access controls, and authentication mechanisms, to protect sensitive information and ensure compliance with regulatory requirements.
7. **End-to-End Testing**:
   * Perform end-to-end testing to validate the entire workflow of the system, from data ingestion to model deployment and result presentation.
   * Test real-world scenarios and use cases to ensure the system meets the needs and expectations of stakeholders and end-users.
   * Verify that the system operates correctly and reliably in production environments, handling unexpected inputs and edge cases gracefully.

**8**.**White Box Testing**

* White-box testing, also known as clear-box testing, glass-box testing, or structural testing, is a software testing technique that examines the internal structure, logic, and code of a software application. In the context of a machine learning project predicting Big Mart sales, white-box testing can be applied to verify the correctness of the underlying code, algorithms, and data processing pipelines.

**Black Box Testing**:

* Black-box testing, also known as behavioral testing or functional testing, is a software testing technique that focuses on testing the functionality of a software application without considering its internal implementation details. In the context of a machine learning project predicting Big Mart sales, black-box testing verifies the system's behavior and outputs based on specified input conditions

**TEST STRATEGY AND APPROACH**

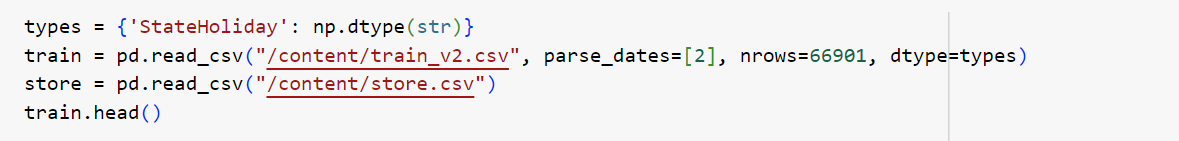
Developing a test strategy and approach for a machine learning project predicting Big Mart sales involves outlining the overall testing process, methodologies, tools, and resources needed to ensure the quality and reliability of the system. Here's a structured approach to defining a test strategy and approach for this project:

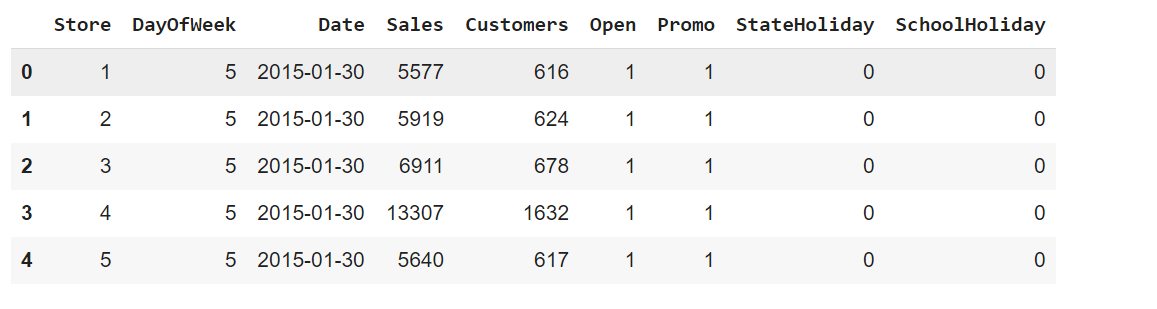
1. **Define Testing Objectives:**
   * Clearly outline the testing objectives, goals, and priorities for the project.
   * Determine what aspects of the system need to be tested, such as data preprocessing, model training, evaluation, integration, and deployment.
2. **Identify Testing Scope:**
   * Define the scope of testing, including the specific functionalities, features, and components of the system to be tested.
   * Identify the boundaries and limitations of testing, such as data availability, resource constraints, and project timelines.
3. **Select Testing Techniques:**
   * Choose appropriate testing techniques based on the nature of the system and the testing objectives.
   * Consider techniques such as black-box testing, white-box testing, integration testing, regression testing, and performance testing.
4. **Determine Test Environment**:
   * Establish the test environment, including hardware, software, and data requirements.
   * Set up testing infrastructure, tools, and resources needed to conduct testing effectively.
5. **Develop Test Cases:**
   * Create test cases and test scenarios based on the testing objectives and requirements.
   * Define input data, expected outputs, preconditions, and postconditions for each test case.
   * Prioritize test cases based on their criticality, complexity, and impact on the system.
6. **Identify Test Data:**
   * Gather or generate test data sets that represent different scenarios and use cases.
   * Ensure test data covers a wide range of input variations, including normal cases, edge cases, and boundary conditions.
7. **Plan Test Execution:**
   * Develop a test execution plan outlining the sequence, schedule, and resources needed to execute the test cases.
   * Assign responsibilities to team members for conducting tests, recording results, and reporting issues.
8. **Execute Test Cases:**
   * Execute test cases according to the test plan and schedule.
   * Record test results, including actual outcomes, observed behavior, and any deviations from expected results.
9. **Monitor and Manage Defects:**
   * Track defects and issues identified during testing using a defect tracking system or issue management tool.
   * Prioritize and triage defects based on severity, impact, and urgency.
   * Collaborate with development teams to investigate, resolve, and verify defect fixes.
10. **Review and Retest:**
    * Review test results and analyze trends, patterns, and areas of improvement.
    * Conduct test result reviews with stakeholders to gather feedback and validate test coverage.
    * Retest fixed defects and regression test affected areas to ensure issues have been resolved satisfactorily.
11. **Report Test Progress:**
    * Provide regular updates on test progress, status, and metrics to project stakeholders.
    * Prepare test summary reports, including test coverage, defect metrics, and quality indicators.
    * Communicate any risks, issues, or concerns that may impact the project timeline or deliverables.
12. **Continuous Improvement:**
    * Continuously evaluate and improve the testing process based on lessons learned and feedback from testing activities.
    * Incorporate automation, optimization, and best practices to enhance test efficiency and effectiveness.
    * Encourage collaboration, knowledge sharing, and skill development within the testing team to foster a culture of quality and excellence

**CHAPTER-10**

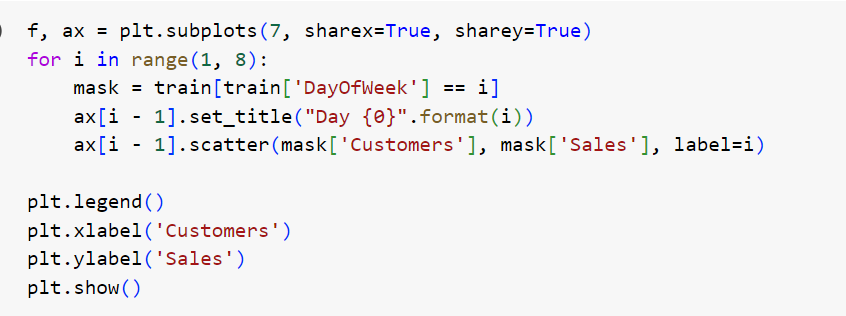
**OUTPUT SCREENSHOTS**

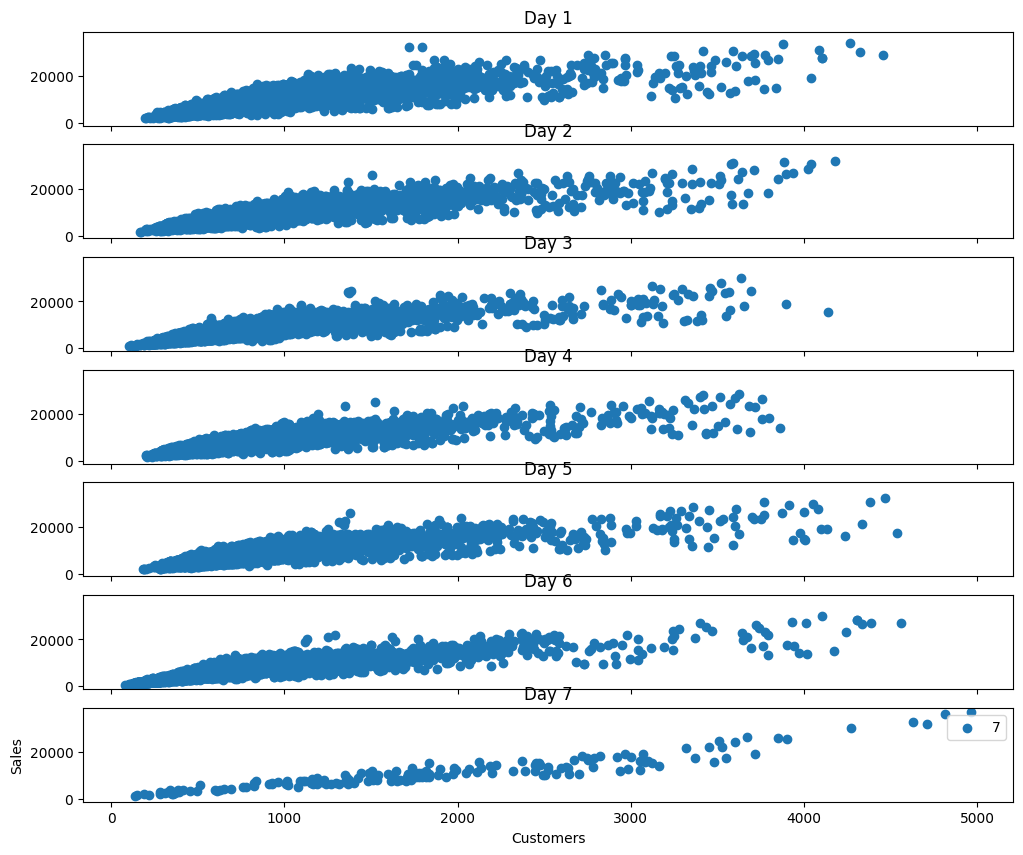
**10.1 Loading the Data Set**



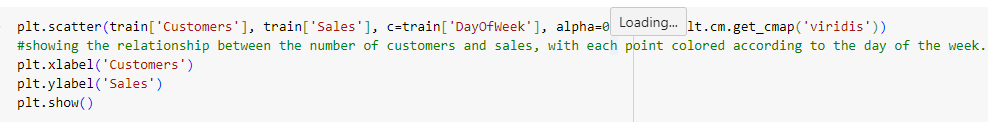


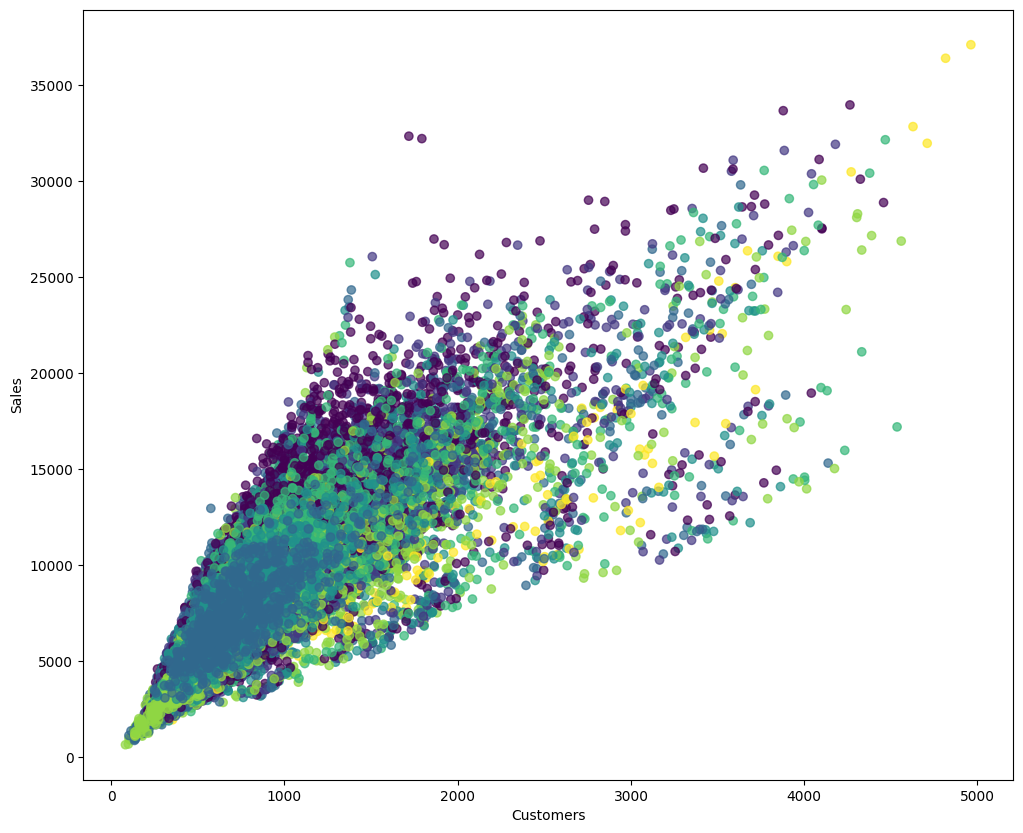
**10.2 Data Validation**





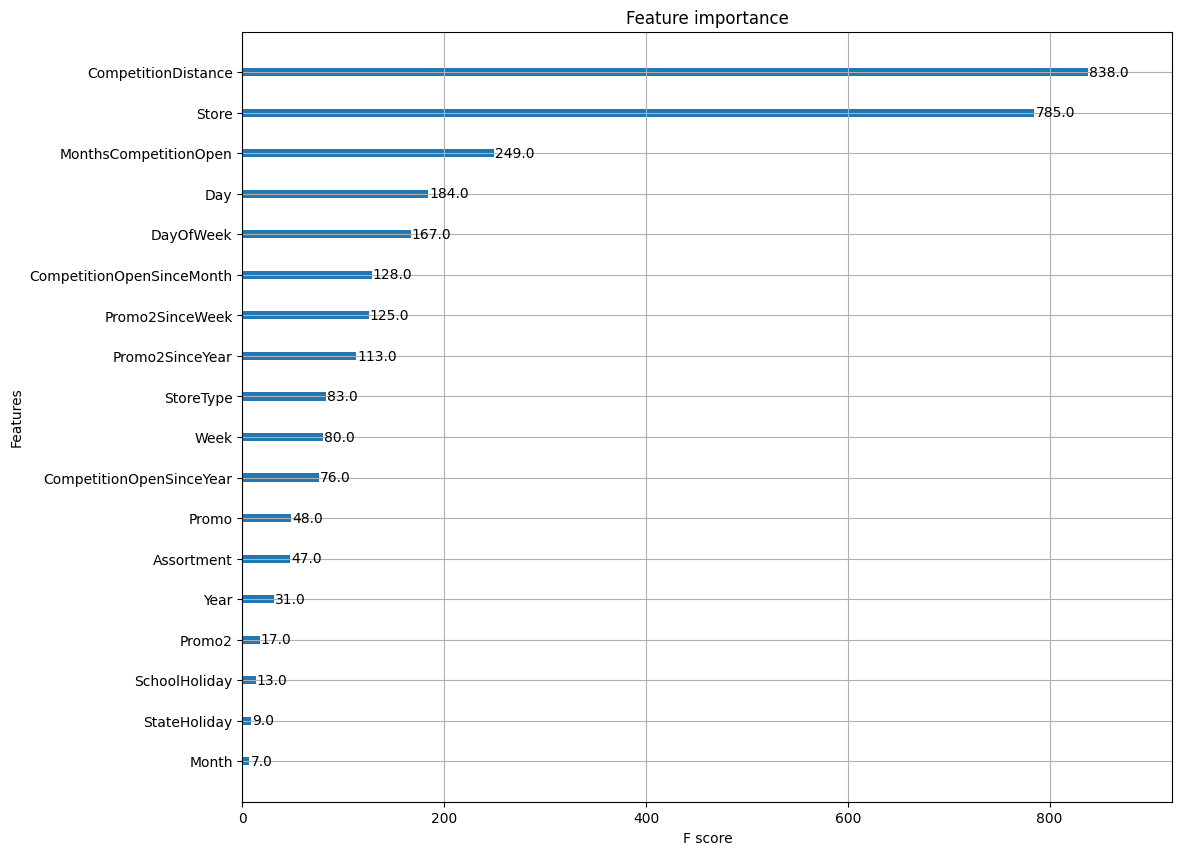
**10.3 Scatter Plot**





**10.4 Result**





**CHAPTER-11**

**CONCLUSION**

In this work, the effectiveness of Decision Tree Regression on the data on revenue and review of, best performance-algorithm, here propose software to using regression approach for predicting the sales centred on sales data from the past the accuracy of linear regression prediction can be enhanced with this method, and Decision Tree Regression can be determined. So, we can conclude Decision Tree Regression gives the better prediction with respect to Accuracy.

FUTURE WORK:

In future, the forecasting sales and building a sales plan can help to avoid unforeseen cash flow and manage production, staff and financing needs more effectively. In future work we can also consider with the ARIMA model which shows the time series graph.

**CHAPTER-12**

**REFERENCES**

[1] Ching Wu Chu and Guoqiang Peter Zhang, “A comparative study of linear and nonlinear models for aggregate retails sales forecasting”, Int. Journal Production Economics, vol. 86, pp. 217- 231, 2003.

[2] Wang, Haoxiang. "Sustainable development and management in consumer electronics using soft computation." Journal of Soft Computing Paradigm (JSCP) 1, no. 01 (2019): 56.- 2.

[3] Suma, V., and Shavige Malleshwara Hills. "Data Mining based Prediction of Demand in Indian Market for Refurbished Electronics." Journal of Soft Computing Paradigm (JSCP) 2, no. 02 (2020): 101- 110

[4] Giuseppe Nunnari, Valeria Nunnari, “Forecasting Monthly Sales Retail Time Series: A Case Study”, Proc. of IEEE Conf. on Business Informatics (CBI), July 2017.

[5] https://halobi.com/blog/sales-forecasting-five-uses/.

[6] Zone-Ching Lin, Wen-Jang Wu, “Multiple Linear Regression Analysis of the Overlay Accuracy Model Zone”, IEEE Trans. On Semiconductor Manufacturing, vol. 12, no. 2, pp. 229 – 237, May 1999.

[7] O. Ajao Isaac, A. Abdullahi Adedeji, I. Raji Ismail, “Polynomial Regression Model of Making Cost Prediction In Mixed Cost Analysis”, Int. Journal on Mathematical Theory and Modeling, vol. 2, no. 2, pp. 14 – 23, 2012.

[8] C. Saunders, A. Gammerman and V. Vovk, “Ridge Regression Learning Algorithm in Dual Variables”, Proc. of Int. Conf. on Machine Learning, pp.

515 – 521, July 1998.IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 56, NO. 7, JULY 2010 3561.

[9] ”Robust Regression and Lasso”. Huan Xu, Constantine Caramanis, Member, IEEE, and Shie Mannor, Senior Member, IEEE. 2015 International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration.”An improved Adaboost algorithm based on uncertain functions”.Shu Xinqing School of Automation Wuhan University of Technology.Wuhan, China Wang Pan School of the Automation Wuhan University of Technology Wuhan, China.

[10] Xinqing Shu, Pan Wang, “An Improved Adaboost Algorithm based on Uncertain Functions”, Proc. of Int. Conf. on Industrial Informatics – Computing Technology, Intelligent Technology, Industrial Information Integration, Dec. 2015.

[11] A. S. Weigend and N. A. Gershenfeld, “Time series prediction: Forecasting the future and understanding the past”, Addison-Wesley, 1994.

[12] N. S. Arunraj, D. Ahrens, A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting,

Int. J. Production Economics 170 (2015) 321-335P

[13] D. Fantazzini, Z. Toktamysova, Forecasting German car sales using Google data and multivariate models, Int. J. Production Economics 170 (2015) 97-135.

[14] X. Yua, Z. Qi, Y. Zhao, Support Vector Regression for Newspaper/Magazine Sales Forecasting, Procedia Computer Science 17 ( 2013) 1055–1062.

[15] E. Hadavandi, H. Shavandi, A. Ghanbari, An improved sales forecasting approach by the integration of genetic fuzzy systems and data clustering: a Case study of the printed circuit board, Expert Systems with Applications 38 (2011) 9392–9399.

[16] P. A. Castillo, A. Mora, H. Faris, J.J. Merelo, P. GarciaSanchez, A.J. Fernandez-Ares, P. De las Cuevas, M.I. Garcia-Arenas, Applying computational intelligence methods for predicting the sales of newly published books in a real editorial business management environment, Knowledge-Based Systems 115 (2017) 133-151.

[17] R. Majhi, G. Panda and G. Sahoo, “Development and performance evaluation of FLANN based model for forecasting of stock markets”. Expert Systems with Applications, vol. 36, issue 3, part 2, pp. 6800-6808, April 2009.

[18] Pei Chann Chang and Yen-Wen Wang, “Fuzzy Delphi and back propagation model for sales forecasting in PCB industry”, Expert systems with applications, vol. 30,pp. 715-726, 2006.

[19] R. J. Kuo, Tung Lai HU and Zhen Yao Chen “application of radial basis function neural networks for sales forecasting”, Proc. Of Int. Asian Conference on Informatics in control, automation, and robotics, pp. 325- 328, 2009.

[20] R. Majhi, G. Panda, G. Sahoo, and A. Panda, “On the development of Improved Adaptive Models for Efficient Prediction of Stock Indices using Clonal-PSO (CPSO) and PSO Techniques”, International Journal of Business Forecasting and Market Intelligence, vol. 1, no. 1, pp.50-67, 2008.

[21] Suresh K and Praveen O, "Extracting of Patterns Using Mining Methods Over Damped Window," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020, pp. 235-241, DOI:

10.1109/ICIRCA48905.2020.9182893.

[22] Shobha Rani, N., Kavyashree, S., & Harshitha, R. (2020). Object Detection in Natural Scene Images Using Thresholding Techniques. Proceedings of the International Conference on Intelligent Computing and Control Systems, ICICCS 2020, Iciccs, 509–515.

[23] https://www.kaggle.com/brijbhushannanda1979/bigmartsalesdata.