

FORECASTING UNIT SALES OF WALMART RETAIL GOODS

Project thesis submitted in partial fulfillment of industry oriented mini project for VI
semester

Bachelor of Technology
In
Computer Science and Engineering

BY

N. SANDEEP (17131A05E1)
SAYYAD AZEEZ (17131A05J4)

Under the Esteemed Guidance of

Dr. D.N.D. Harini
Associate Professor
Department Of CSE
GVPCOE (A)



Department of Computer Science and Engineering

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (Autonomous)

Approved by AICTE, New Delhi and Affiliated to JNTU-Kakinada
Re-accredited by NAAC with “A” Grade with a CGPA of 3.47/4.00
Madhurawada, Visakapathanam-530 048

Year of Submission: 2020

Certificate

This is to certify that the project thesis entitled “**Forecasting Unit Sales of Walmart Retail Goods**” being submitted by

N.Sandeep (17131A05E1)
Sayyad Azeez (17131A05J4)

In partial fulfillment for the mini project in the Department of Computer Science and Engineering to the Jawaharlal Nehru Technological University Kakinada, Kakinada is a record of bonafied work carried out under my guidance and supervision.

The results embodied in this project thesis have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Project Guide
Dr. D.N.D. Harini
Associate Professor
Department Of CSE
GVPCOE (A)

Head of the Department
Dr. P. Krishna Subba Rao
Professor
Department Of CSE
GVPCOE (A)

ACKNOWLEDGEMENT

We would like to express our deep sense of gratitude to our esteemed institute Gayatri Vidya Parishad College of Engineering (Autonomous), which has provided us an opportunity to fulfil our cherished desire.

We thank **Prof. A. B. K. Rao**, Principal, Gayatri Vidya Parishad College of Engineering (A) for extending his utmost support and cooperation in providing all the provisions for the successful completion of the project.

We consider it our privilege to express our deepest gratitude to **Dr. P.K. Subba Rao**, Professor, Head of the Department, Department Computer Science and Engineering for his valuable suggestions and constant motivation that greatly helped the project work to get successfully completed.

We express our profound gratitude and our deep indebtedness to our guide **Dr. D.N.D. Harini**, Associate Professor whose valuable suggestions, guidance and comprehensive assistance helped us a lot in realizing our present project.

We also thank **Dr. Omprakash Tembhurne**, Associate Professor, Project coordinator, for guiding us throughout the project and lead us in completing our project efficiently.

We also thank all the members of the staff in Computer Science Engineering for their sustained help in our pursuits.

We thank all those who contributed directly or indirectly in successfully carrying out his work.

Project Members:

N. SANDEEP (17131A05E1)

SAYYAD AZZEZ (17131A05J4)

ABSTRACT

Many household products are sold by various branches of the retail store network, which are geographically located at multiple locations. Supply chain inefficiencies will occur at different places when the market potential is not evaluated by the retailers correctly. Many times it is not easy for retailers to understand the market condition at various locations. The organization of the retail store network has to understand the market conditions to raise its goods to be bought and sold so that many customers get attracted in that direction. Business forecast helps retailers to visualize the forecasting of the sales to get a general idea of the coming years if any changes are needed then those changes are done in the retail store's goal so that success is achieved more profitably. It also helps the customers to be happy by providing the products wanted by them in the wanted time when the customers are happy then they prefer the store that provides all the resources they need to their satisfaction by this the sales in the particular store in which the customers purchase more items increases causing more profit. The forecasting of sales helps to know the retailers the demand for the prod

INDEX

Contents

1. Introduction	3
1.1 Objective	4
1.2 Problem Statement	4
2. Literature Survey	5
2.1 Relevant Work:	5
2.1.1 Linear Regression	5
2.1.2 KNN Regression	6
2.1.3 Random Forest Regressor	7
2.1.4 Extra tree Regressor	9
2.1.5 XGBoost Regressor	9
2.1.6 CatBoostRegressor	9
3. Software Requirement Analysis	10
3.1 NumPy	10
3.1.1 Introduction	10
3.1.2 Operations Using Numpy	10
3.1.3 Nddarray Object	10
3.2 Pandas	10
3.2.1 Introduction	10
3.2.2 read_csv	11
3.2.3 get_dummies	11
3.3 Matplotlib.Pyplot	11
3.4 Seaborn	12
3.5 Sklearn	12
4. Software Design	13
4.1 Architecture Design	13
4.2 The Forecasting Process	13
4.3 Introduction to UML	15
4.4 Building blocks of UML	15
4.5 UML Diagram	16
5. Software and Hardware Requirements	18
Software Requirements	18
Hardware Requirements	18
6. Coding	19

7. Testing	25
8. Result.....	26
9. Conclusion.....	34
10. Further Enhancements	35
11. References	36

1. Introduction

Everyone is very interested in the future! Very excited to know what will happen with all of us the very next moment, tomorrow, similarly the merchants also curious about their business, its development, reducing factors of it. By adopting certain steps the aspects that can cause damage or reduce the profit can be avoided. In this goal of forecasting the sales business the data from the various sectors is collected and the data analytics is done the efficient understanding of the observed data by common steps is not practically possible because the data is very huge the masses of data of an organization is formed in such a way that it's having meaning by understanding deeply the suitable actions can be taken.

Let us consider the retail store's network example Walmart it is an example for giant stores like Dmart, big bazaars, etc. The retail store offers fresh produce, meat, dairy products, bakery, household supplies, health and beauty aids, and a pharmacy and obtains a profit by that. There are different branches of the retail store network whose locations are variously located at multiple geographical locations most of the time retailers will not be successful in knowing the customer's needs because they need to be able to evaluate the market potential at that location, during special occasions or special events like Christmas, Black Friday, thanksgiving day, etc. The rate of sales or shopping is more sometimes this may cause inefficiency in the products.

The relationship between the customers and the stores needs to be analysed and the changes need to be made in order to obtain more profit. The history of the purchase of each product in each store and department is to be maintained and observing these sales, forecasts to be made which enables the knowledge of profit and loss that occurred during that year. In a particular department during the particular session let us consider example Christmas. During the Christmas festival, the sales are more in a department like clothing, footwear, etc., then during summer, the sales of cotton clothes are more, during winter the sales for Sweaters are more. The sales of products changes as per the session by observing this history of sales, the sales can be predicted for the future. That finds the solution of uneasiness in the business of the retail store chain. Supply chain management is the advantage of competition, the main majors of supply change management are to increase the profit of sales and to manage the inventory turnovers when the supply change is observed properly then a clear picture is obtained about a particular store whether there is a profit from that store are its under loss. Then accordingly suitable operations are done to be successful. Here the retailers observe the customers and they target them by some attractive offers. So that they will be back to the store and spend a long time and more money.

In this project, the model forecasts the sales by using different ML Algorithms. Machine learning algorithm which provides efficient results are applied so that huge data can be modelled, python is a programming language and software environment for statistic computing and graphs. The python programming is used by the statisticians and data miners widely to develop the statistical software and data analysis. The various algorithm is used to predict sales. The seasonality, trend, and randomness are observed in the algorithm. The algorithm is used for train data sets and then the sales prediction is done.

1.1 Objective

- This model is an untidy place for predicting the cost of sales in each department and each store on a given date.
- The main aim of the project is to predict the sales in the store.

1.2 Problem Statement

The model is provided with historical sales data for 45 Walmart stores located in different regions. Each store contains several departments, and the main task of the model predicting the department-wide sales for each store.

Also, Walmart runs several promotional markdown events throughout the year. These markdowns precede prominent holidays, the four largest of which are the Super Bowl, Labor Day, Thanksgiving, and Christmas. The weeks including these holidays are weighted five times higher in the evaluation than non-holiday weeks. Part of the challenge presented by this competition is modeling the effects of markdowns on these holiday weeks in the absence of complete/ideal historical data.

2. Literature Survey

The present problem statement is based on the analysis of the sales and prediction of the sales. In this algorithm, model build by using some ML algorithms to analyze the sales of Walmart.

The data is gathered from the data set which was available on the Kaggle website where the data set for machine learning is found. In this project first, the data is collected, then cleaned the data, further preprocess the data then the data is available for the analysis and prediction. In the last few years, analyzing and forecasting the sales of Walmart has become an important field for researchers and retailers to grow their market chain. In most of the cases, the researchers had attempted to establish a linear relationship between the input macroeconomic variables and the Sales Analysis. After the discovery of nonlinearity in the Sales analysis, much important literature has come up in nonlinear statistical modeling of sales analysis, most of them required that the nonlinear model be specified before the estimation is done. But since sales analysis is noisy, uncertain, chaotic, and nonlinear. ML has evolved out to be a better technique in capturing the structural relationship between a sales performance and its determination factors more accurately than many other statistical techniques.

In literature, different sets of variables are used to predict sales. Different input variables are used to predict the same set of sales data. Some researchers used input data from a single time series where others considered the inclusion of heterogeneous market information and macro-economic variables. Some researchers even preprocessed these input data sets before feeding it to the machine learning for analysis.

2.1 Relevant Work:

2.1.1 Linear Regression

Linear Regression is a machine learning algorithm based on **supervised learning**. It performs a **regression task**. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between the dependent and independent variables, they are considering and the number of independent variables being used.

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

$$Y = \theta_1 + \theta_2.x$$

While training the model, model takes:

x: input training data (univariate – one input variable (parameter))

y: labels to data (supervised learning)

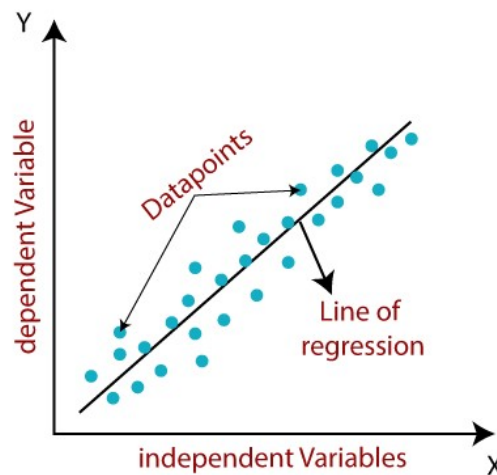


Fig 3: Linear Regression

2.1.2 KNN Regression

K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining, and intrusion detection.

The K-Nearest Neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

The KNN algorithm assumes that similar things exist nearby. In other words, similar things are near to each other. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics — calculating the distance between points on a graph. There are other ways of calculating distance, and one way might be preferable depending on the problem that is being solved. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice.

There are two major types of supervised machine learning problems, called *classification* and *regression*. In classification, the goal is to predict a class label, which is a choice from a predefined list of possibilities. Classification is sometimes separated into two types namely: binary classification or multi-class classification. Binary classification is similar to answer a yes or no question.

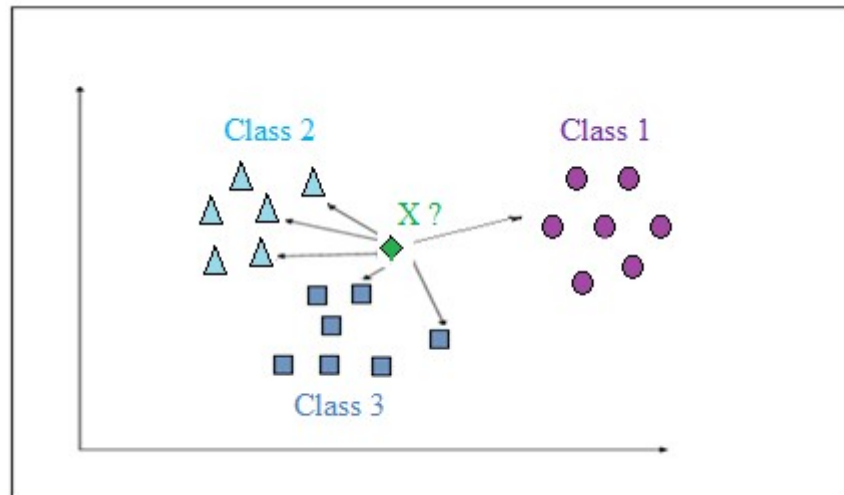


Fig 4: KNN Classifier

For the regression task, the goal is to predict a floating number in programming terms or perhaps a real number in terms of mathematics. An easy way to distinguish between classification and regression problem is to ask if there is some kind of continuity in the output. If there is continuity between possible outcomes, then the problem is a regression problem. The more complex our model will be, the better model will be able to predict the training data. However, if our model becomes too complex, the model starts focusing too much on each individual data point in our training set, and our model will not generalize well with our new data. There is always a sweet spot in between that will generalize well.

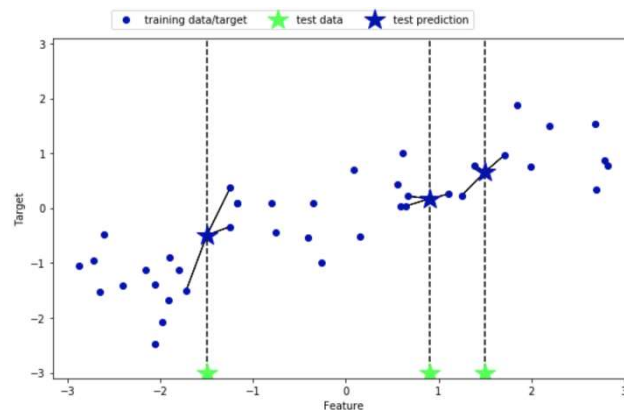


Fig 5: KNN Regression

2.1.3 Random Forest Regressor

Random Forest is an ensemble machine learning technique capable of performing both regression and classification tasks using multiple decision trees and a statistical technique called **bagging**. Bagging along with boosting are two of the most popular ensemble techniques which aim to tackle high variance and high bias. An RF instead of just averaging the prediction of trees it uses two key concepts that give it the name *random*.

1. A Random sampling of training observations when building trees

2. Random subsets of features for splitting nodes

Random forest builds multiple decision trees and merges their predictions together to get a more **accurate** and **stable** prediction rather than relying on individual decision trees.

Algorithm Steps

Step 1: Samples are taken repeatedly from the training data so that each data point is having an equal probability of getting selected, and all the samples have the same size as the original training set.

$X = 0.1, 0.5, 0.4, 0.8, 0.6$, $y = 0.1, 0.2, 0.15, 0.11, 0.13$ where x is an independent variable with 5 data points and y is dependent variable.

Now Bootstrap samples are taken with replacement from the above data set. **N_estimators** is set to 3 (no of tree in the random forest), then:

The first tree will have a bootstrap sample of size 5 (same as the original dataset), assuming it to be: $x_1 = \{0.5, 0.1, 0.1, 0.6, 0.6\}$ likewise

$x_2 = \{0.4, 0.8, 0.6, 0.8, 0.1\}$

$x_3 = \{0.1, 0.5, 0.4, 0.8, 0.8\}$

Step 2: A Random Forest Regressor model is trained at each bootstrap sample drawn in the above step, and a prediction is recorded for each sample.

Step 3: Now the ensemble prediction is calculated by averaging the predictions of the above trees producing the final prediction.

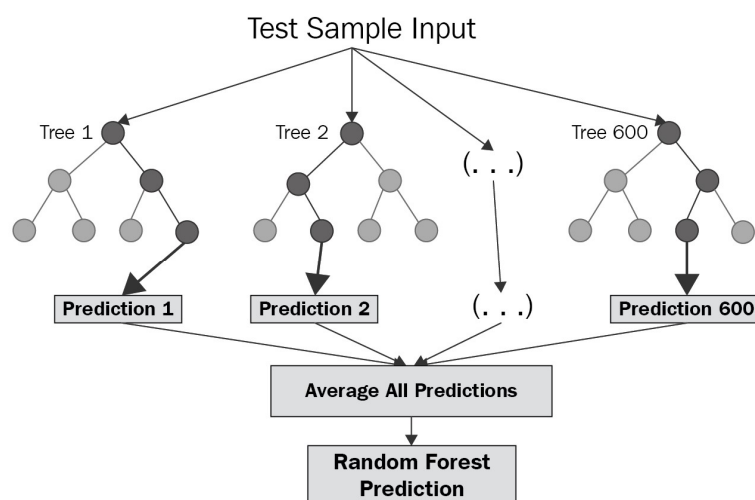


Fig 6: Random Forest Regression

2.1.4 Extra tree Regressor

The Extra Trees and Random Forest algorithms are almost the same. In the Random Forest algorithm, the tree splitting phenomenon is deterministic whereas, in the case of Extremely Randomized Trees, the split of the trees is completely random. In other words, during the process of splitting, the algorithm chooses the best split among random splits in the selected variable for the current decision tree. The features employed are like the ones used in the previous algorithms. Python's ExtraTreesRegressor function from the scikit-learn class was used to execute the algorithm, and the various performance metrics calculated for the previous methods are evaluated and reported.

The Extra-Tree method (standing for **extremely randomized trees**) was proposed with the main objective of further randomizing tree building in the context of numerical input features, where the choice of the optimal cut-point is responsible for a large proportion of the variance of the induced tree. Concerning random forests, the method drops the idea of using bootstrap copies of the learning sample, and instead of trying to find an optimal cut-point for each one of the K randomly chosen features at each node, it selects a cut-point at random.

2.1.5 XGBoost Regressor

XGBoost (Extreme Gradient Boosting) belongs to a family of boosting algorithms and uses the gradient boosting (GBM) framework at its core. It is an optimized distributed gradient boosting library.

Boosting is a sequential technique which works on the principle of an ensemble. It combines a set of weak learners and delivers improved prediction accuracy. At any instant t, the model outcomes are weighed based on the outcomes of previous instant t-1. The outcomes predicted correctly are given a lower weight and the ones miss-classified are weighted higher. Note that a weak learner is one that is slightly better than random guessing.

2.1.6 CatBoostRegressor

CatBoost is a recently open-sourced machine learning algorithm from Yandex. It can easily integrate with deep learning frameworks like Google's TensorFlow and Apple's Core ML. It can work with diverse data types to help solve a wide range of problems that businesses face today. To top it up, it provides best-in-class accuracy.

"CatBoost" name comes from two words "**C**ategory" and "**B**oosting". "**B**oost" comes from a gradient boosting machine learning algorithm as this library is based on a gradient boosting library. Gradient boosting is a powerful machine learning algorithm that is widely applied to multiple types of business challenges like fraud detection, recommendation items, forecasting and it performs well also. It can also return very good results with relatively less data, unlike DL models that need to learn from a massive amount of data.

3. Software Requirement Analysis

3.1 NumPy

3.1.1 Introduction

NumPy is a Python package. It stands for '**Numerical Python**'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created the NumPy package by incorporating the features of Numarray into the Numeric package. There are many contributors to this open-source project.

3.1.2 Operations Using Numpy

The following operations are performed using NumPy

1. Mathematical and logical operations on arrays.
2. Fourier transforms and routines for shape manipulation.
3. Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

3.1.3 Nddarray Object

The most important object defined in NumPy is an N-dimensional array type called **ndarray**. It describes the collection of items of the same type. Items in the collection can be accessed using a zero-based index.

Every item in an ndarray takes the same size as the block in the memory. Each element in ndarray is an object of the data-type object (called **dtype**).

SYNTAX

numpy.array

It creates a ndarray from any object exposing an array interface, or from any method that returns an array.

3.2 Pandas

3.2.1 Introduction

Pandas is an open-source Python library providing high-performance data manipulation and analysis tools using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data. Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, statistics, analytics, etc.

Key Features of Pandas

- Fast and efficient DataFrame object with the default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Columns from a data structure can be deleted or inserted. Group by data for aggregation and transformations.

- High-performance merging and joining of data.
- Time Series functionality.

3.2.2 read_csv

`read_csv` is an important pandas function to read CSV files and do operations on it. Opening a CSV file through this is easy. But there are many other things one can do through this function only to change the returned object completely. For instance, one can read a CSV file not only locally, but from a URL through `read_csv` or one can choose what columns needed to export.

```
[2] import pandas as pd
features = pd.read_csv('/content/sample_data/features.csv')
train = pd.read_csv('/content/sample_data/train.csv')
stores = pd.read_csv('/content/sample_data/stores.csv')
test = pd.read_csv('/content/sample_data/test.csv')
sample_submission = pd.read_csv('/content/sample_data/sampleSubmission.csv')
```

Fig 1: Load Data with Pandas

3.2.3 get_dummies

The `get_dummies()` function is used to convert a categorical variable into dummy/indicator variables.

```
import pandas as pd
df = pd.DataFrame({'Type': ['a', 'b', 'a']})
type = pd.get_dummies(df, prefix = 'Type')
```

Type				
	Type		Type_a	Type_b
0	a		1	0
1	b		0	1
2	a		1	0

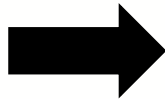


Fig 2: Converting the Type data type from object to int using `get_dummies` method

3.3 Matplotlib.Pyplot

Matplotlib is a plotting library for creating static, animated, and interactive visualizations in Python. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits like Tkinter, awxPython, etc.

Pyplot is a Matplotlib module which provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python and the advantage of being free and open-source. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. The various plots that can be utilized using Pyplot are **Line Plot**, **Histogram**, **Scatter**, **3D Plot**, **Image**, **Contour**, and **Polar**.

3.4 Seaborn

Seaborn is a library for making statistical graphics in Python. It is built on top of Matplotlib and closely integrated with pandas data structures.

Here is some of the functionality that seaborn offers:

- A dataset-oriented API for examining relationships between multiple variables
- Specialized support for using categorical variables to show observations or aggregate statistics
- Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data
- Automatic estimation and plotting of linear regression models for different kinds of dependent variables
- Convenient views onto the overall structure of complex datasets
- High-level abstractions for structuring multi-plot grids that let you easily build complex visualizations
- Concise control over matplotlib figure styling with several built-in themes
- Tools for choosing color palettes that faithfully reveal patterns in your data

Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

3.5 Sklearn

scikit-learn is an open-source Python library that implements a range of machine learning, preprocessing, cross-validation, and visualization algorithms using a unified interface.

Important features of scikit-learn:

- Simple and efficient tools for data mining and data analysis. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, etc.
- Accessible to everybody and reusable in various contexts.
- Built on the top of NumPy, SciPy, and matplotlib.
- Open source, commercially usable – BSD license.

-

4. Software Design

4.1 Architecture Design

The below figure, gives an overview of the Walmart sales analysis. The system makes use of an existing dataset as an input. The first frame is considered as the reference frame. The subsequent frames are taken as the input frames. All the null data points in the dataset has to been cleaned. The modified data is used to analysis the data and also used to predict the data.

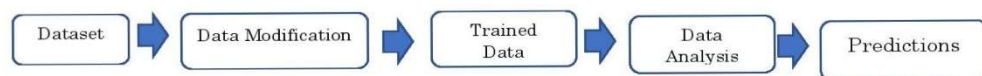


Figure 7: Overview of Walmart Sales Forecast.

In the prediction, assume that the trained dataset as input. It considered two or three flied and made comparison on the data.it processes the data and give the required output.

4.2 The Forecasting Process

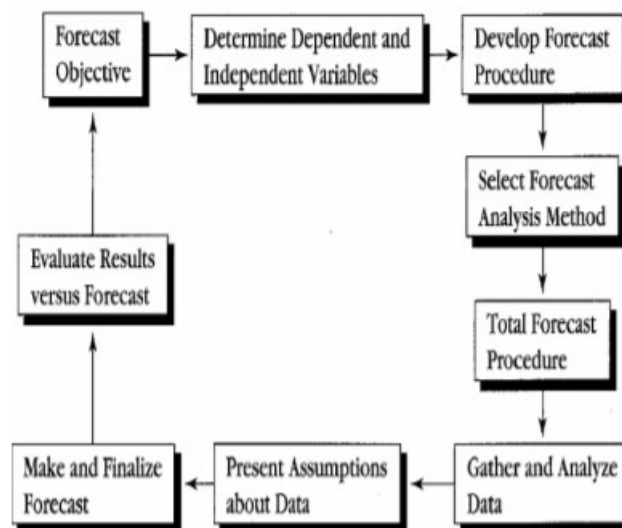


Fig 8: Forecasting Process of Walmart Sales

The process of forecasting is a group of methods to predict sales. It is initiated after determining the objective. It may include the sales amount in dollars, the number of employees to be appointed. The selection of dependent and independent variables is done.

The forecasting results like sales data or the number of employees to be appointed in the upcoming year. The market factor includes factors like product existence in the store, its quality, and the demand of the item. The market index is a market factor that is expressed as the amount of percentage relatively with some base content. When the market index is increased then the industry sales are increased. The index consists of many market factors like price, the population of the area, disposable personal income. Then in the forecasting process, the procedures of forecast and methods that are useful for data analysis are determined. If the procedures were not used prior then the firm may want to test the procedures. Then gathering and analyzing data is done. Certain assumptions are made about the forecasted sales. Then the sales forecast is finalized as time passes and the results are evaluated.

Below figure as shows read the data set as the input and read the data is converted into trained data. The trained data is different from dataset and trained dataset. The data differences processing stage executed on the trained dataset by the machine learning algorithms. Initially the raw data is to be merged to form the final training data. Once the training data is prepared the unnecessary columns are dropped, after this our training data is ready for analysis and prediction.

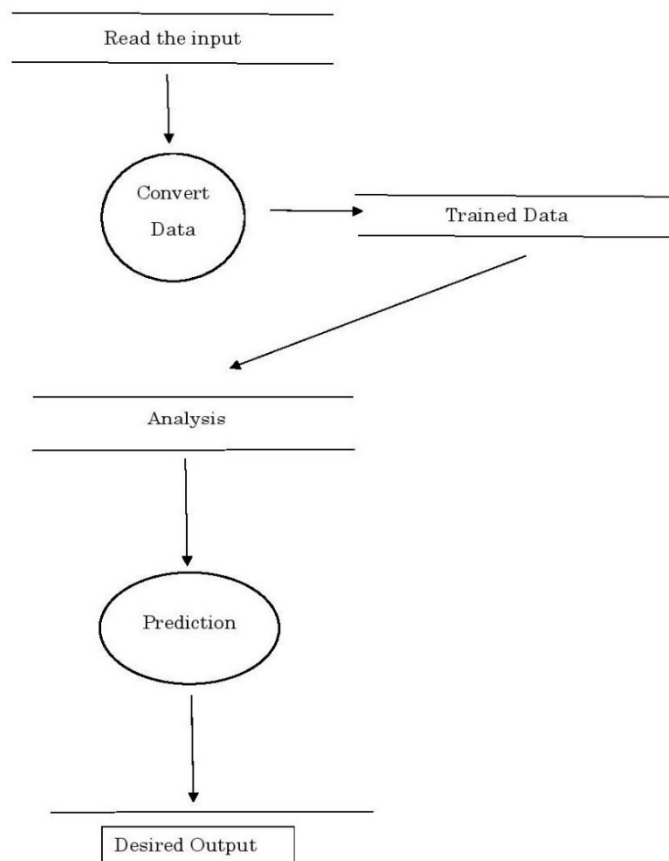


Figure 9: Data Flow Diagram for Walmart Sales Forecast.

4.3 Introduction to UML

A UML diagram is a diagram based on the UML(Unified Modeling Language) to visually represent a system along with its main actors, roles, actions, artifacts or classes, to better understand, alter, maintain, or document information about the system.

Benefits of UML Diagrams:

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So UML becomes essential to communicate with non-programmer's essential requirements, functionalities, and processes of the system.
- A lot of time is saved down the line when teams are able to visualize processes, user interactions and static structure of the system.

Object Oriented Concepts Used in UML:

1. **Class** – A class defines the blueprint i.e. structure and functions of an object.
2. **Objects** – Objects help us to decompose large systems and help us to modularize our system. Modularity helps to divide our system into understandable components so that it can build system piece by piece. An object is the fundamental unit (building block) of a system that is used to depict an entity.
3. **Inheritance** – Inheritance is a mechanism by which child classes inherit the properties of their parent classes.
4. **Abstraction** – Mechanism by which implementation details are hidden from the user.
5. **Encapsulation** – Binding data together and protecting it from the outer world is referred to as encapsulation.
6. **Polymorphism** – Mechanism by which functions or entities are able to exist in different forms

4.4 Building blocks of UML

UML is composed of three main building blocks, i.e., things, relationships, and diagrams. Building blocks generate one complete UML model diagram by rotating around several different blocks. It plays an essential role in developing UML diagrams.

The building blocks of UML can be defined as –

- Things
- Relationships
- Diagram

Things are the most important building blocks of UML. Things can be :

- Structural
- Behavioural
- Grouping
- Annotational

4.5 UML Diagram

Use Case Diagram

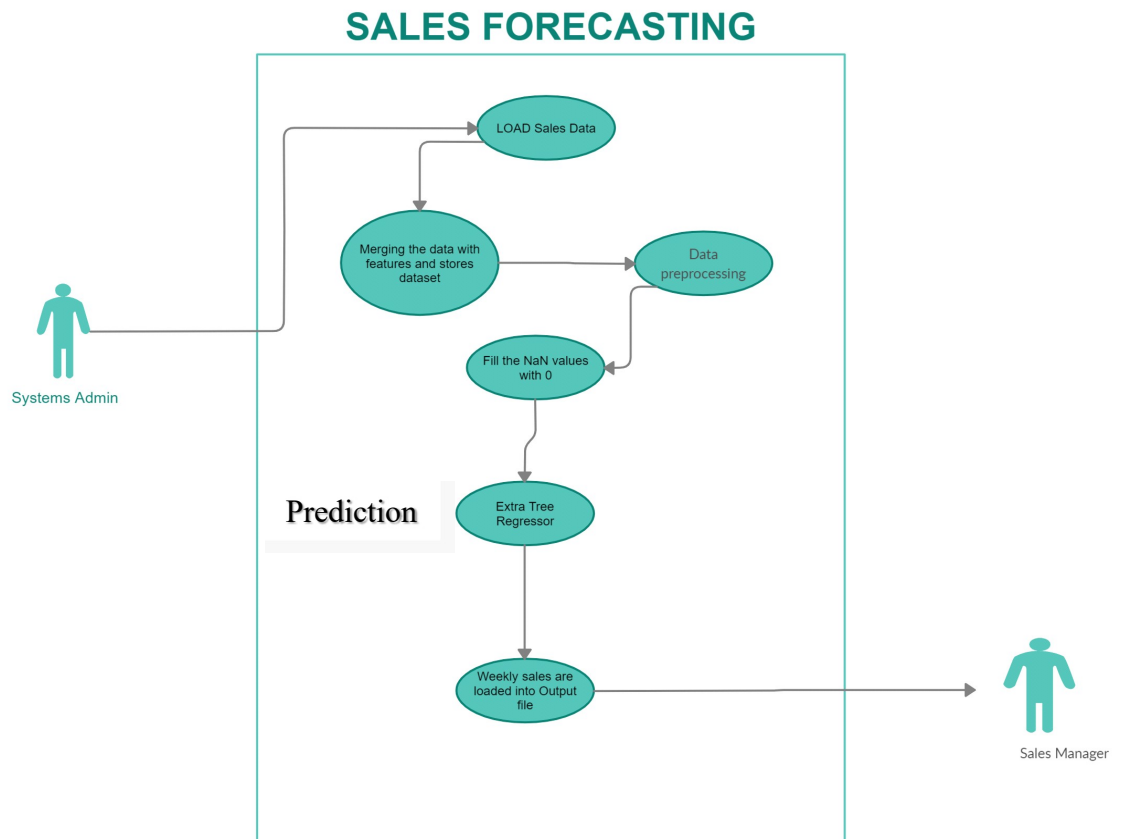


Fig 10: Use Case Diagram

Sequence Diagram

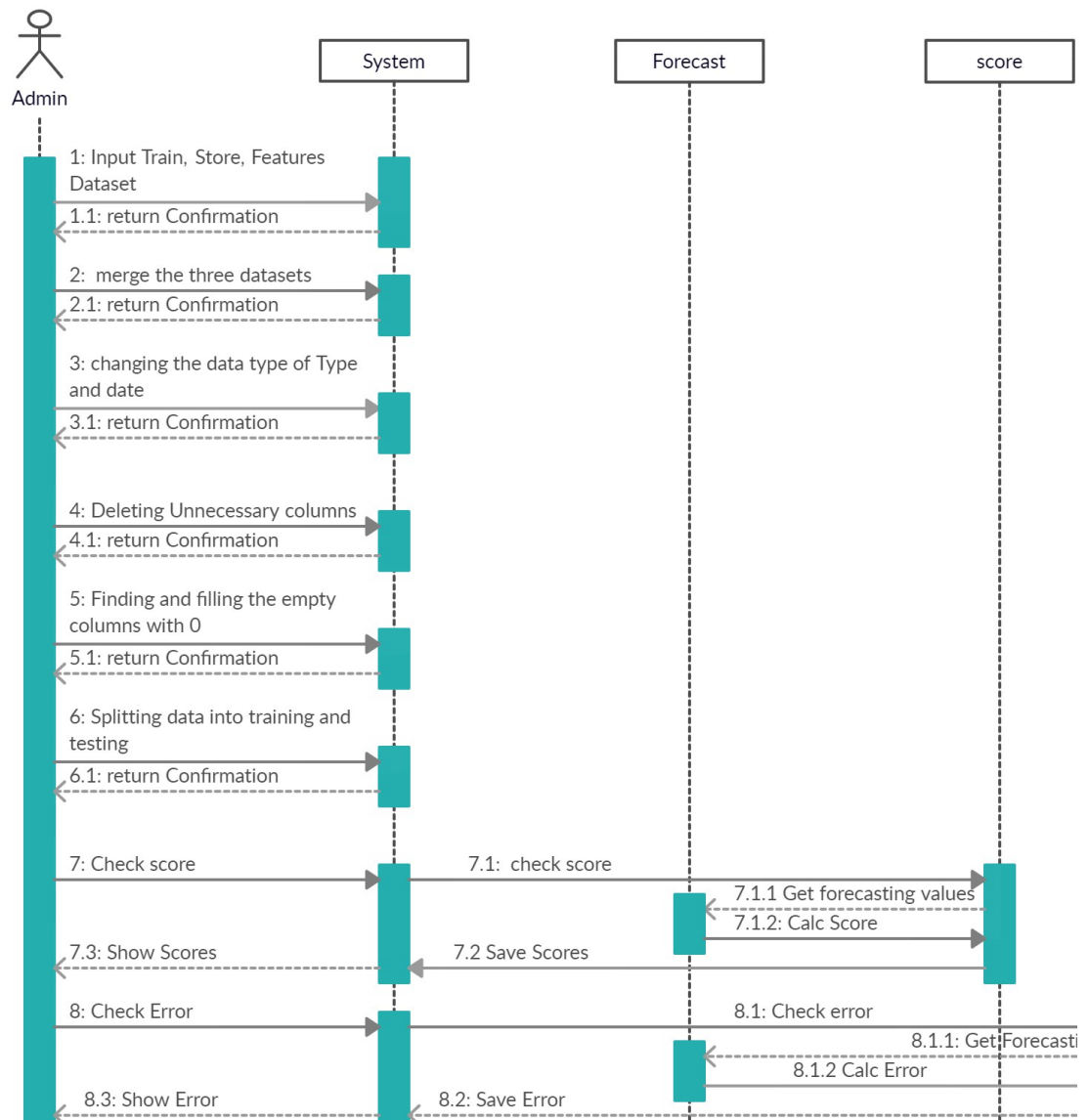


Fig 11: Sequence Diagram

5. Software and Hardware Requirements

The software and hardware requirement for Walmart sales forecast is as follows:

Software Requirements

Operating System: Windows/ Linux

Language: Python

Libraries: Numpy, Pandas, matplotlib, Scikit-Learn

Editors: Google Collab

Hardware Requirements

Minimum Processor: Intel I5 or more

Hardware: 8GB RAM or more

Disk Space: 1TB

6. Coding

#The Numpy and Pandas files are imported which helps to read the CSV files

```
import numpy as np
import pandas as pd
```

#The datasets of the train, test, feature and store are loaded using read_csv.

```
train = pd.read_csv('/content/sample_data/train.csv')
test = pd.read_csv('/content/sample_data/test.csv')
feature = pd.read_csv('/content/sample_data/features.csv')
store = pd.read_csv('/content/sample_data/stores.csv')
```

#The below three lines of the code is to see the sample data of the dataset.

```
train.head(2)
store.head(2)
feature.head(2)
```

#The train dataset is merged with features and stores dataset to prepare the complete train dataset.

```
train = train.merge(feature).merge(store)
train.head(2)
```

#train.info() command gives the information about the train dataset which includes the column data type, no of non-null content

```
train.info()
```

#matplotlib is imported which supports the plots like barblots, correlation matrix

```
import matplotlib.pyplot as plt
```

```
plot = plt.figure(figsize=(18, 14))
corr = train.corr()
c = plt.pcolor(corr)
plt.yticks(np.arange(0.5, len(corr.index), 1), corr.index)
plt.xticks(np.arange(0.5, len(corr.columns), 1), corr.columns)
plot.colorbar(c)
```

```
import seaborn as sns
```

```
sns.pairplot(train, vars=['Weekly_Sales', 'Fuel_Price', 'Size', 'CPI', 'Dept', 'Temperature',
'Unemployment'])
```

```
plt.figure(figsize=(12,6))
sns.distplot(train['Weekly_Sales'], color='y')
```

```

plt.show()

sns.countplot(train['IsHoliday'])

sns.countplot(train['Type'])

corr_matrix = train.corr()
corr_df = corr_matrix['Weekly_Sales'].sort_values(ascending = False)
corr_df

#here the dummies are eliminated like type column has a,b,c values which is not
supported so we created the 3 additional columns like type_a, type_b, type_c

type = pd.get_dummies(train['Type'], prefix = 'Type')
type.head(2)

#concatenating the new 3 columns with the existing train dataset.

train = pd.concat([train, type], axis=1)
train.head(3)

#date data type is modified into day, month, and year which are later concatenated into
train dataset

train['Date'] = pd.to_datetime(train['Date'])
train['Day'] = train['Date'].dt.day
train['Month'] = train['Date'].dt.month
train['Year'] = train['Date'].dt.year
train.info()

#dropped the date and type columns because the columns are modified earlier and new
columns are added
train.drop('Date', axis=1, inplace=True)
train.drop('Type', axis=1, inplace=True)

#the below line of code is to give a total number of null vales in each column

train.isna().sum()
#the fillna will allows you to fill the null values with 0
train.fillna(value=0, inplace=True)

train.isna().sum()

corr_matrix = train.corr()
corr_df = corr_matrix['Weekly_Sales'].sort_values(ascending = False)
corr_df

plt.figure(figsize=(16,8))

```



```

sns.heatmap(train.corr(), annot=True)
plt.show()

#dropped the unemployment and fuel price column because they have the low corr value
wrt weekly sales.

train.drop(['Unemployment','Fuel_Price'], axis=1, inplace=True)

#divided the x, y as the x is the input training data and y is the labels to data
x= train.drop('Weekly_Sales', axis=1).values
y = train['Weekly_Sales'].values

#the train data is splitted according to test size of 0.3 rate.
from sklearn.model_selection import train_test_split
train_x,test_x,train_y,test_y = train_test_split(x,y,test_size=0.3,random_state=100)
train_x
train_y

from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.linear_model import LinearRegression

linear = LinearRegression()
linear.fit(train_x, train_y)
pred_linear_y = linear.predict(test_x)
linear_score=linear.score(test_x, test_y)
linear_score*100
mse_linear = mean_squared_error(test_y,pred_linear_y)
mse_linear
mae_linear = mean_absolute_error(test_y,pred_linear_y)
mae_linear

from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsRegressor(n_neighbors=9)
knn.fit(train_x, train_y)
pred_knn_y = knn.predict(test_x)
score_knn=knn.score(test_x, test_y)
score_knn=score_knn*100
mse_knn = mean_squared_error(test_y,pred_knn_y)
mse_knn
mae_knn = mean_absolute_error(test_y,pred_knn_y)
mae_knn

from sklearn.ensemble import ExtraTreesRegressor, RandomForestRegressor
etg = ExtraTreesRegressor(n_estimators=100,max_features='auto', verbose=1,
n_jobs=1)
etg.fit(train_x, train_y)
pred_etg_y = etg.predict(test_x)

```

```

score_etg=etg.score(test_x, test_y)
score_etg=score_etg*100
mse_etg = mean_squared_error(test_y,pred_etg_y)
mae_etg = mean_absolute_error(test_y,pred_etg_y)
score_etg

#RANDOM FOREST REGRESSOR

from sklearn.ensemble import RandomForestRegressor
rfg = RandomForestRegressor(n_estimators=100,max_features='log2', verbose=1)
rfg.fit(train_x, train_y)
pred_rfg_y = rfg.predict(test_x)
score_rfg=rfg.score(test_x, test_y)
score_rfg=score_rfg*100
mse_rfg = mean_squared_error(test_y,pred_rfg_y)
mae_rfg = mean_absolute_error(test_y,pred_rfg_y)
score_rfg

#XGBoost Regressor

import xgboost as xgb
reg = xgb.XGBRegressor(n_estimators=500)
reg.fit(train_x, train_y)
pred_reg_y = reg.predict(test_x)
score_reg=reg.score(test_x, test_y)
score_reg=score_reg*100
mse_reg = mean_squared_error(test_y,pred_reg_y)
mae_reg = mean_absolute_error(test_y,pred_reg_y)

#CatBoostRegressor

from catboost import CatBoostRegressor
cbr = CatBoostRegressor()
cbr.fit(train_x, train_y)
pred_cbr_y = cbr.predict(test_x)
score_cbr=cbr.score(test_x, test_y)
score_cbr=score_cbr*100
mse_cbr = mean_squared_error(test_y,pred_reg_y)
mae_cbr = mean_absolute_error(test_y,pred_reg_y)

algorithms = ['Linear Regression', 'KNN', 'Extra Tree Regressor', 'Random Forest
Regressor', 'XGBoost', 'catBoostRegresor']
mse = [mse_linear, mse_knn, mse_etg, mse_rfg, mse_reg, mse_cbr]
mae = [mae_linear, mae_knn, mae_etg, mae_rfg, mae_reg, mae_cbr]
score = [linear_score, score_knn, score_etg, score_rfg, score_reg, score_cbr]
plt.figure(figsize=(16, 6))

```

```

sns.barplot(algorithms, mse)

plt.figure(figsize=(16, 6))
sns.barplot(algorithms, mae)

plt.figure(figsize=(16, 6))
sns.barplot(algorithms, score)

#from here the forecasting started, initially loaded test dataset the merged the other dataset
to form complete test dataset.
test = pd.read_csv('/content/sample_data/test.csv')
test=test.merge(feature, how='left').merge(store, how='left')

#eliminate the dummies similar to train dataset
type_test = pd.get_dummies(test['Type'], prefix = 'Type')
type_test.head(2)

#filled all the NA values with 0
test[['MarkDown1','MarkDown2','MarkDown3','MarkDown4', 'MarkDown5']] =
test[['MarkDown1','MarkDown2','MarkDown3','MarkDown4','MarkDown5']].fillna(0)
test = test.fillna(0)

#converted all date data type into int data type
column_date = test['Date']
test['Date'] = pd.to_datetime(test['Date'])
test['Day'] = test['Date'].dt.day
test['Month'] = test['Date'].dt.month
test['Year'] = test['Date'].dt.year
test.info()
#concatinating the type column
test = pd.concat([test, type_test], axis=1)
test.head(3)
test.info()

#dropping the unnecessary columns
test.drop('Date', axis=1, inplace=True)
test.drop('Type', axis=1, inplace=True)
test.drop(['Unemployment','Fuel_Price'], axis=1, inplace=True)

#prediction is done by using the extra tree regressor because of high accuracy and low
RMSE
prediction = etg.predict(test)
test['Weekly_Sales'] = prediction
test['Date'] = column_date
test['id'] = test['Store'].astype(str) + '_' + test['Dept'].astype(str) + '_' +
test['Date'].astype(str)
test = test[['id', 'Weekly_Sales']]
test = test.rename(columns={'id': 'Id', 'Weekly_Sales': 'Weekly_Sales'})

```

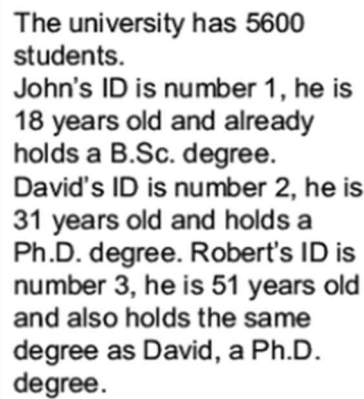
```
#output data is loaded into the output.csv  
test.to_csv('/content/sample_data/output.csv', index=False)  
op = pd.read_csv('/content/sample_data/output.csv')
```

7. Testing

Testing is the process of detecting errors. Testing performs a very critical role for quality assurance and for ensuring the reliability of software. The results of testing are used later on during maintenance also. Purpose of Testing: The aim of testing is often to demonstrate that a program works by showing that it has no errors. The basic purpose of testing phase is to detect the errors that may be present in the program. Hence one should not start testing with the intent of showing that a program works, but the intent should be to show that a program doesn't work. Testing Objectives: The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time.

In this project, row data like unstructured data will not be detected. The data which is in the form of separated filled link dataset only works.

The row data like unstructured data which is in the form will no be executed



The university has 5600 students.
John's ID is number 1, he is 18 years old and already holds a B.Sc. degree.
David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Figure 12: Unstructured data

The structure data link data set which in the form will be executed

ID	Name	Age	Degree
1	John	18	B.Sc.
2	David	31	Ph.D.
3	Robert	51	Ph.D.
4	Rick	26	M.Sc.
5	Michael	19	B.Sc.

Figure 13: Structured data

8. Result

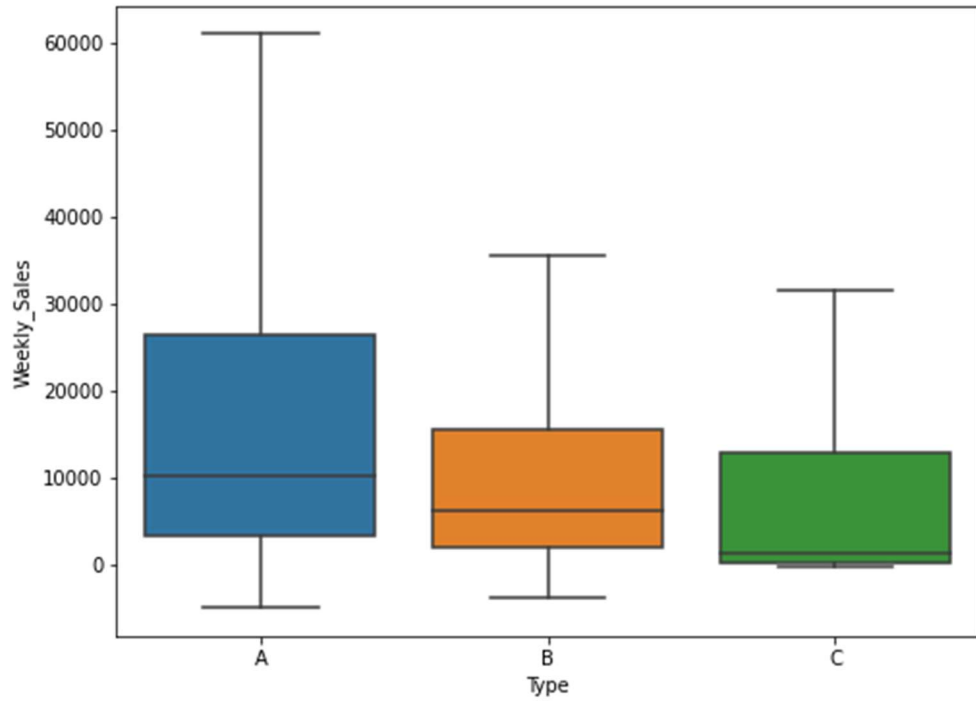


Fig 14: Type of Store vs. Weekly Sales

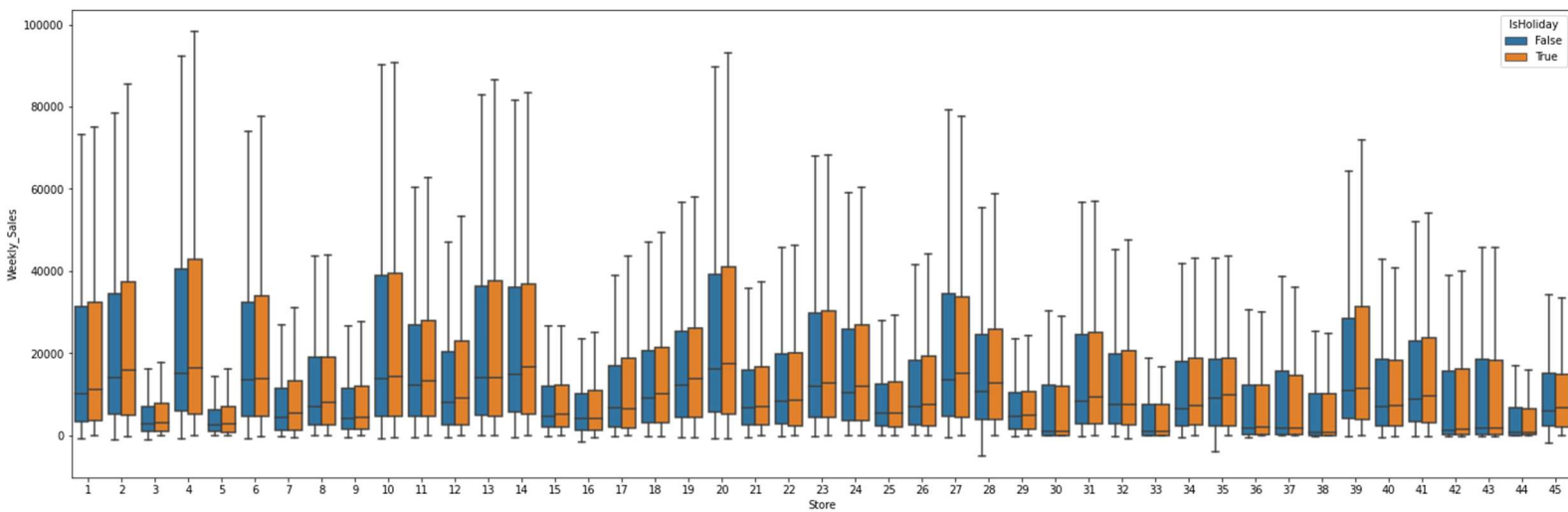


Fig 15: store vs. Weekly Sales

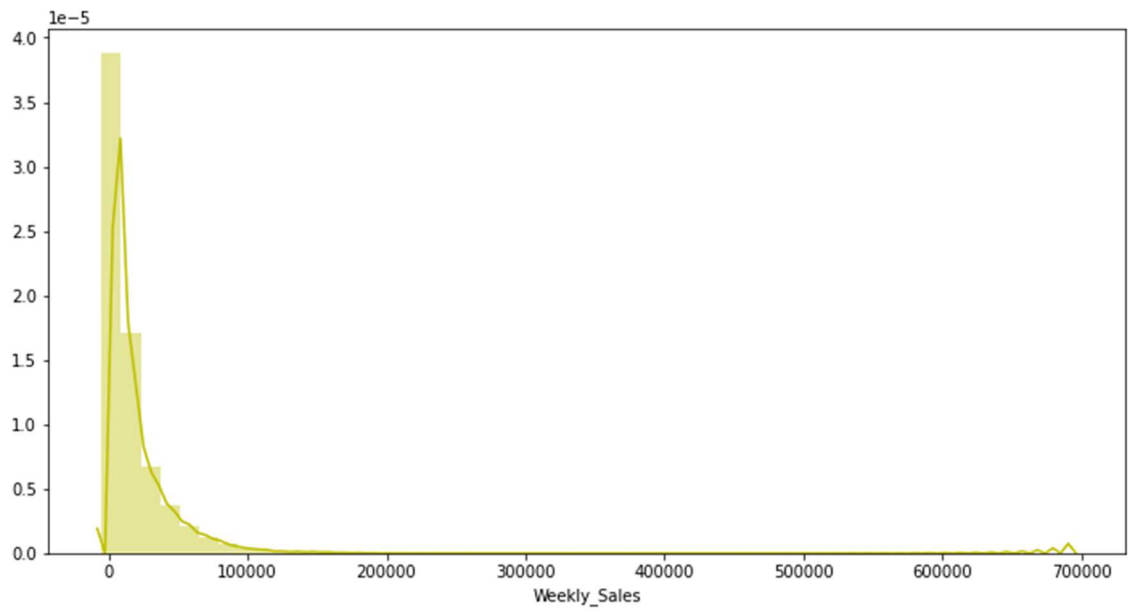


Fig 16: Weekly Sales of the stores

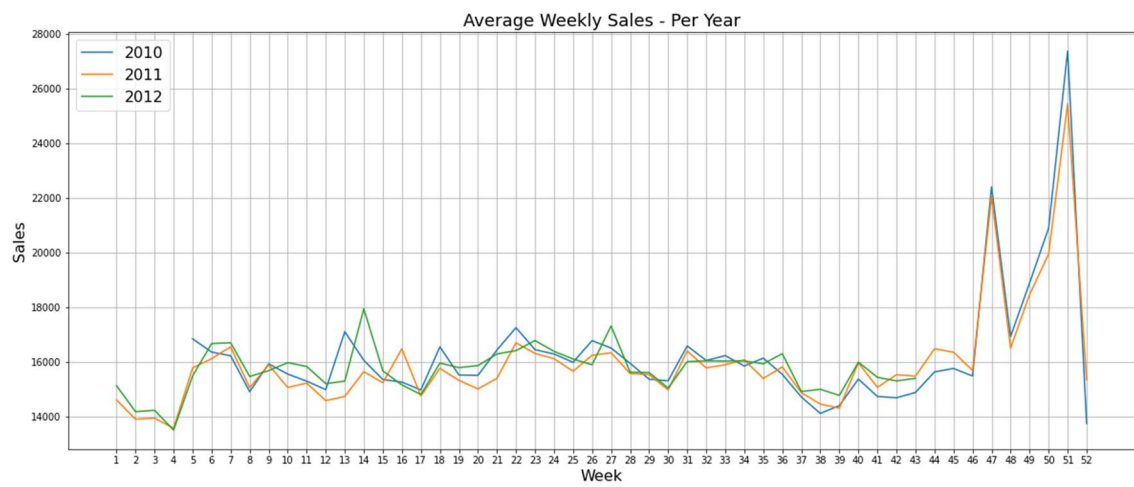


Fig 17: Avg. Weekly Sales per Year

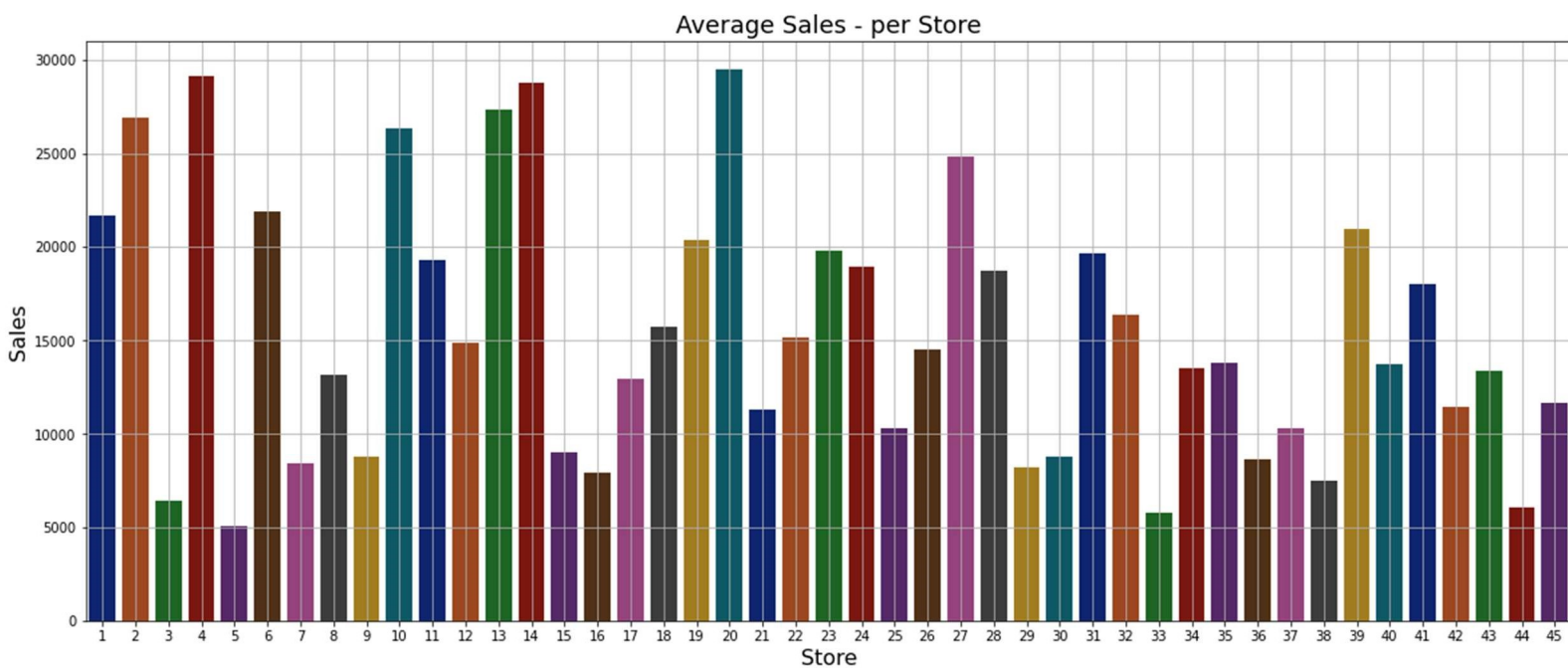


Fig 18: Avg. Sales per Store

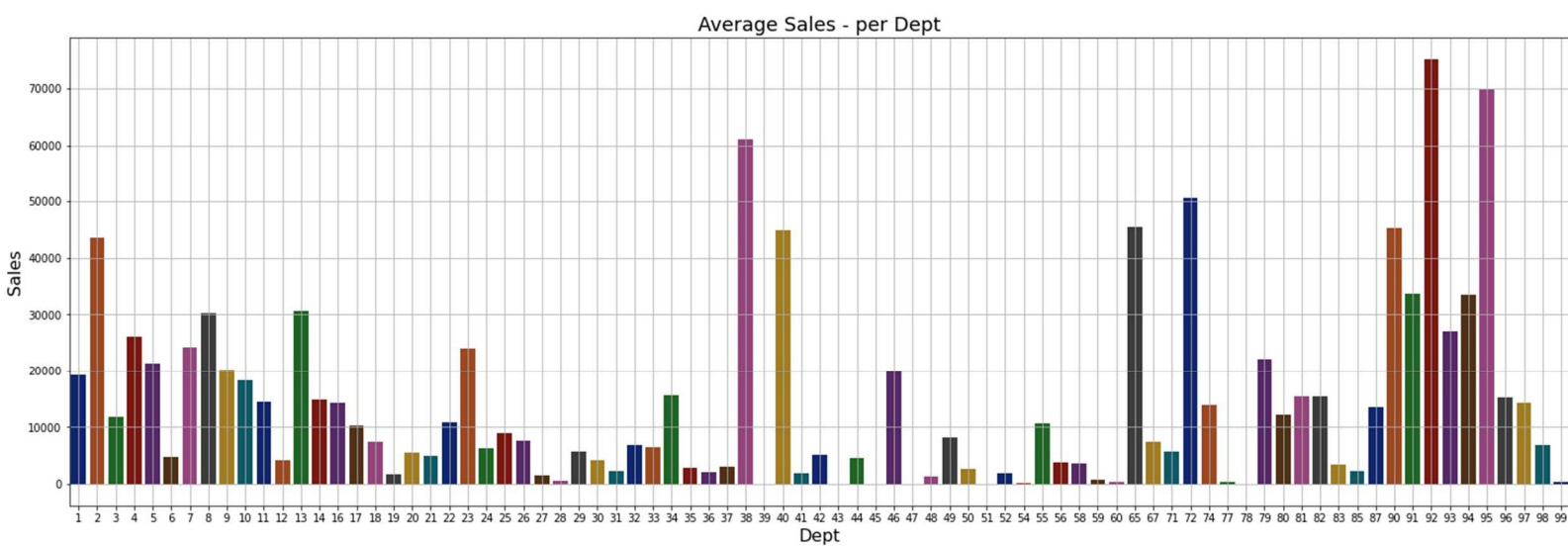


Fig 19: Avg. Sales per dept

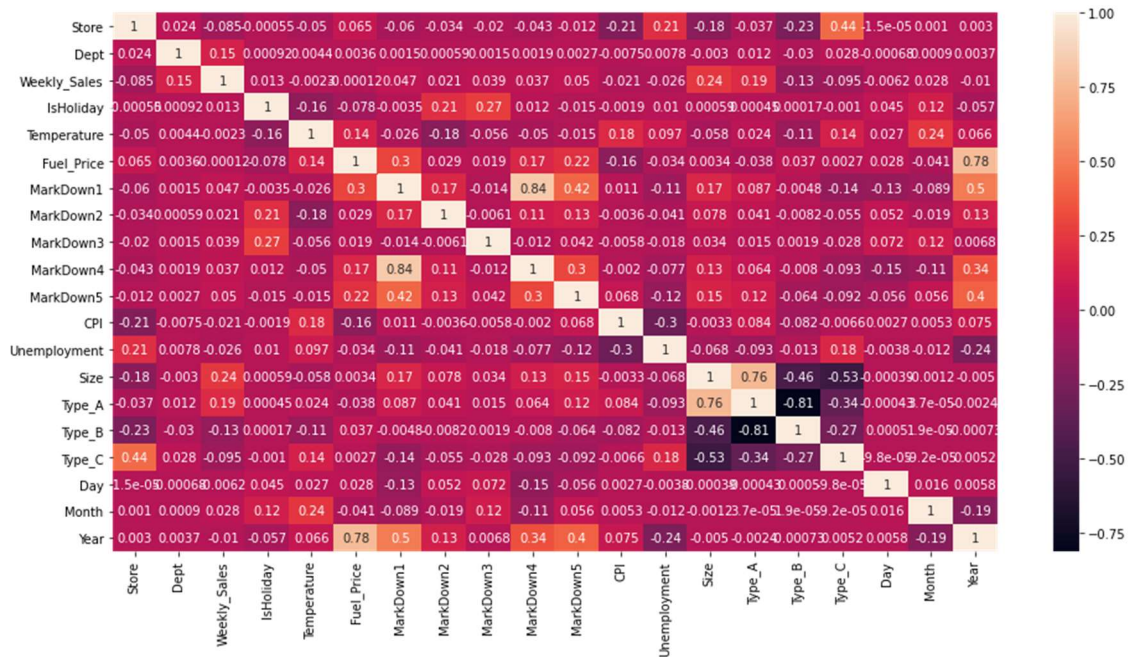


Fig 20: Correlation matrix with values

Here is the observation made from the plots which are plotted using matplotlib and seaborn libraries. As shown in fig 14, stores with more sizes have higher sales records, like a store of type A record more sales than the other two types of store, at the same time type C stores record low sales during a given period. It is also observed that 48.9% of type A stores and 37.8% of type B stores are present. Similarly, fig 15 shows the relation between weekly sales of the 45 stores on holidays and non-holidays. It is observed that Holidays and Stores do not show significant relations but just small higher sales soaring when there is a holiday. And it is also came to know that each department shows the different levels of sales, the Department may be a powerful variable to predict sales. When the department and type of store are considered together, generally the department in type A shows the highest sales record. 72 department shows the highest surge in sales during holiday and Sales on holiday is a little bit more than sales in not-holiday. From fig 16, it can be said that the maximum sales of the stores are in the range of 0 to 100,000 and there are very few stores whose sales are more than 100,000. Fig 17 explains about the avg. sales in a week in the 3 years. In 2010 last week of December i.e. week 51 registered highest sales which are the avg. of 45 stores because of Christmas, similarly in the year of 2011 week 51 record the highest avg. sales. In contrast to the other two years in 2012, week 14 in which Good Friday occurs has the highest avg. sales this is because the data is up to 43rd week of 2012 is available. In Fig 18 it shows us that the avg. sales of stores 4 and 20 are high compared to other store sales. In contrast, it shows that the avg. sales of store 5 and 33 are low. In fig 19 it shows avg. sales per department, in department 92 and 95 record highest sales and some departments like department 39, 43, 45, 47, 51 and 79 record very minimal sales. Fig 19 which shows the correlation matrix is very important for feature selection and to eliminate the unnecessary columns WRT our Label.

Algorithm	RMSE	MAE	Accuracy
Linear Regression	21698.512	14582.234	9.203
KNN Regression	18634.043	11473.039	33.038
XGBoost Regressor	3429.664	1739.421	97.732
Random Forest Regression	6785.958	3599.671	91.120
Cat Boost Regressor	8689.227	4996.594	94.069
Extra Tree Regression	3600.117	1411.430	97.501

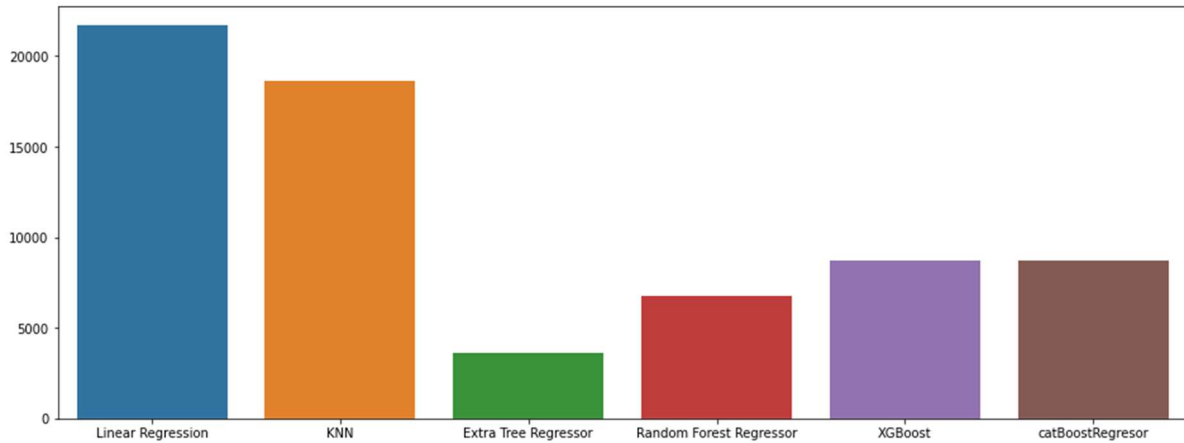


Fig 21: Algorithm vs. Root Mean Square Error

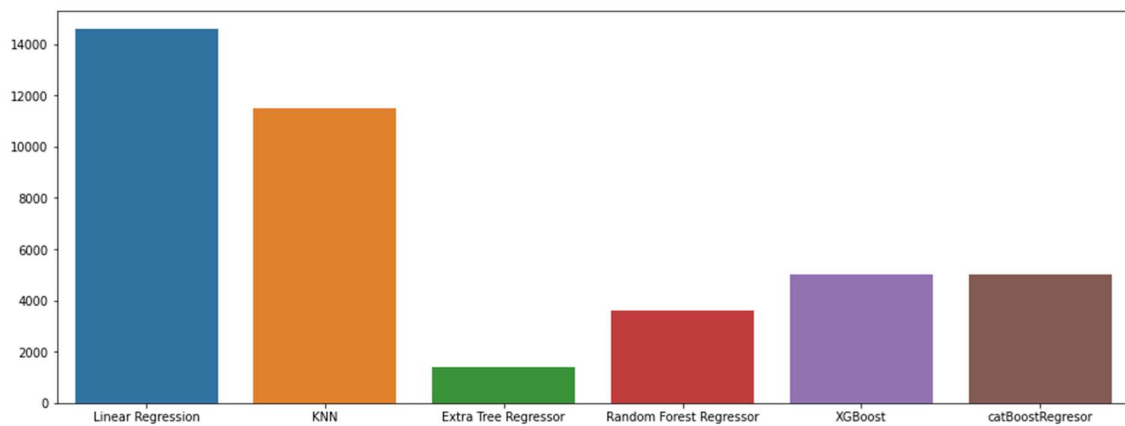


Fig 22: Algorithm vs. Mean Absolute Error

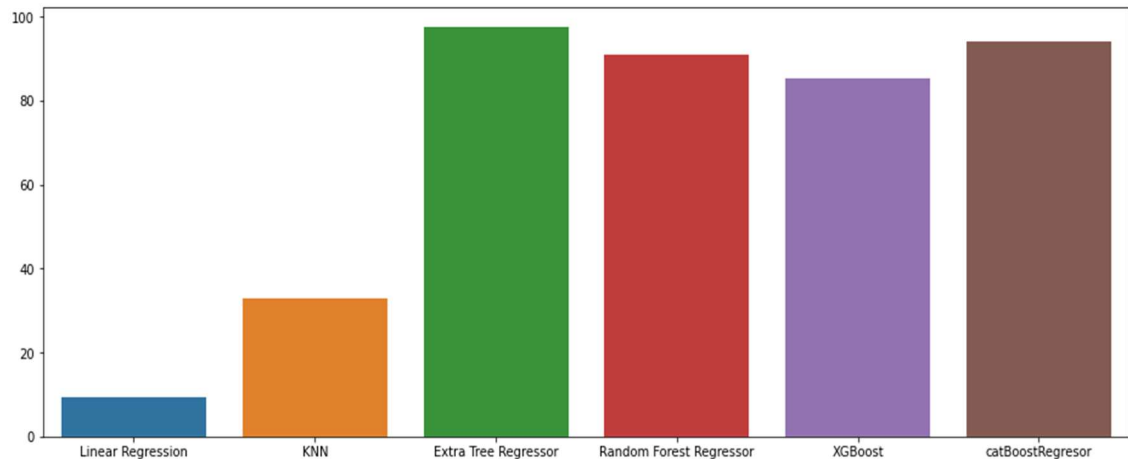


Fig 23: Algorithm vs. Accuracy Score

The forecasting of sales begins with the loading of the train data into the Linear regression while using linear regression the model predicts the sales value with an accuracy of 9.203% which is very low. The accuracy is low because when dealing with datasets having a large number of features Linear Regression may outperform. Linear regression is parametric as linear regression supports best for linear solutions only. Secondly performed KNN regression KNN is a non-parametric model. KNN is slow in real-time because it has to keep track of all training data and need to find the neighbor nodes, but the linear regression model can easily extract output for tuned coefficients. Knn is better than linear regression when the data have high SNR (Signal to Noise Ratio). XGBoost requires depends on 3 parameters which control the overfitting.

1) Learning rate: Shrinks the contribution of each successive tree in the ensemble. Makes the model more robust by shrinking the weights on each step. Typical final values to be used: 0.01-0.2

2) max depth: Maximum depth of the Decision Tree, controls the complexity of the decision tree. Used to control over-fitting as higher depth will allow the model to learn relations very specific to a particular sample. Typical values: 3-10.

3) min child length: Defines the minimum sum of weights of all observations required in a child. Used to control over-fitting. Higher values prevent a model from learning relations which might be highly specific to the particular sample selected for a tree. Too high values can lead to under-fitting hence, it should be tuned using CV.

Whereas cat boost depends on

1) Learning Rate: Learning Rate is used for reducing the gradient step. It affects the overall time of training: the smaller the value, the more iterations are required for training. Overfitting is detected — decrease the learning rate.

2) Depth: The optimal depth ranges from 4 to 10. Values in the range from 6 to 10 are recommended. The range of supported values depends on the processing unit type and the type of the selected loss function.

3) L2 leaf reg (L2 Leaf regularization): Coefficient at the L2 regularization term of the cost function. It is used to overcome overfitting

XGBoost doesn't support categorical features but Cat boost does, which improves training results. CatBoost allows the usage of non-numeric factors instead of preprocessing data or time spent and effort turning it into a number.

Dealing with data sets having a large number of Features Random Forest will give the best results. Because it is a non-parametric ensemble model. RF can handle co-linearity better than linear regression. RF can support automatic feature interaction where KNN cannot have this feature and also RF is faster than KNN in real-time. So random Forest Performs best when comparing to other algorithms. But Extra Tree regressor fit more accurately than Random Forest. Extra tree Regressor is an extension of Random Forest. Random Forest has splitting nodes on the best split among a random subset of features selected at every node. But extra tree split nodes with two key differences:

- i) It does not bootstrap observations which means it take samples without replacement.
- ii) Nodes are split randomly but not on the best split.

Algorithm	HyperParameters	Accuracy
Linear Regression	Default	9.203
KNN	n_neighbors = 11	33.038
Extra Tree Regressor	n_estimators=100 max_features='auto' verbose=1 n_jobs= 1	97.501
Random Forest Regressor	n_estimators=58 max_depth=27 max_features=6 min_samples_split=3 min_samples_leaf=1	91.120
XGBoost	n_estimators=500 learning_rate= 0.05 max_depth = 10 subsample = 1.0 colsample_bytree = 0.7 objective = 'reg:linear' eval_metric = 'rmse' silent = 1	97.732

	Id	Weekly_Sales
0	1_1_2012-11-02	19895.35
1	1_1_2012-11-09	20280.85
2	1_1_2012-11-16	23534.44
3	1_1_2012-11-23	31982.29
4	1_1_2012-11-30	23481.57

Fig 24: Output file sample

Forecasted Sales are stored in a new CSV file. It consists of Id and weekly Sales. Weekly sales are the forecasted sales on a given date. The id is the combination of store number, department number, and date.

9. Conclusion

This project dealt with the implementation of six algorithms namely, Linear Regression, XGBoost, KNN, Random Forest, cat Boost and Extra Trees, on the Walmart dataset and a comparative analysis was carried out to determine the best algorithm. Extra Trees was confirmed to be a very effective model in forecasting sales data. Extra Trees, an extension of Random Forest, which showed very good accuracy for the best implementations. These algorithms could possibly produce even better results if they are provided with better hardware electronics like Graphics Processing Units (GPUs), and their use would be beneficial in providing valuable insight, thus leading to better decision-making.

Most of the shopping malls / shopping centers plan to attract the customers to the store and make profit to the maximum extent by them. Once the customers enter the stores they are attracted then definitely they shop more by the special offers and obtain the desired items which are available in the favorable cost and satisfy them. If the products as per the needs of the customers then it can make maximum profit the retailers can also make the changes in the operations, objectives of the store that cause loss and efficient methods can be applied to gain more profit by observing the history of data the existing stores a clear idea of sales can be known like seasonality trend and randomness. The advantage of forecasting is to know the number of employees should be appointed to meet the production level. Sales drop is bad thing forecasting sales helps to analyze it and it can overcome through the sales drop to remain in the competition forecast plays a vital role.

10. Further Enhancements

Future work would include the Extra Trees model being developed to consider sparse promotional markdown data and moving holidays. It would also involve the fine-tuning of the hyperparameters of the models to improve the accuracy of prediction. Future work could also entail combining the models to produce an ensemble training model that could represent even the tiniest details present in the data. With the development of deep learning techniques, the results of this research could be further improved in the near future through the use of more complex and multilayer ANNs. This work shows that there are highly efficient algorithms to forecast sales in big, medium or small organizations.

11. References

1. Ezgigm, Walmart Store Sales Forecasting, 2020
2. Geurts, Pierre, Damien Ernst, and Louis Wehenkel. "Extremely randomized trees." *Machine learning* 63.1 (2006).
3. Hai Rong Lv Xin Xin Bai Wen Jun Yin Jin Dong, Simulation Based Sales Forecasting On Retail Small Stores.
4. Mark Lutz and David Ascher, "*Learning Python*", O'Reilly Publications, 5th edition.
5. Oliver Theobald "Machine Learning For Absolute Beginners: A Plain English Introduction", 2nd Edition.
6. Xinxin Bai, Wei Shang, Wenjun Yin, Jin Dong, A Service-Oriented Solution for Retail Store Network Planning.