Sales Forecasting using XGBoost

Dr K. Alice
Dept. of Computing Technologies
SRM Institute of Science and Technology, Kattankulathur, Kancheepuram, Tamil Nadu, India
alicek@srmist.edu.in

Syed Hamad ul Haq Andrabi
Dept. of Computing Technologies
SRM Institute of Science and Technology, Kattankulathur, Kancheepuram, Tamil Nadu, India
sa3184@srmist.edu.in

Siddharth Anoop Srivastava
Dept. of Computing Technologies
SRM Institute of Science and Technology, Kattankulathur, Kancheepuram, Tamil Nadu, India
ss8341@srmist.edu.in

***Abstract*—This study intends to investigate several machine learning algorithms for sales forecasting strategies. A retailer can use this to predict future market demand and adjust its inventory levels accordingly. The accuracy of these predictions will determine whether the retailer profits or suffers losses. In this paper, we worked on the Walmart Sales dataset from Kaggle. It has over 400,000 rows and about 20 columns. After cleaning and performing the necessary feature engineering of the data, we used machine learning algorithms such as eXtreme Gradient Boosting (with and without tuned hyperparameters), Linear Regression, Ridge Regression, Decision Tree Regressor and Random Forest Regressor (with and without tuned hyperparameters). The most effective algorithm out of all the others was XGBoost when the hyperparameters were tuned. This model performs well on sales prediction by utilising less processing power and memory.**

Keywords—Sales Forecasting, Data Preprocessing, Feature Engineering, XGBoost, Machine Learning,

I. Introduction

Sales Forecasting issues that are both challenging and intriguing are rather typical. Accurate projections can aid businesses in risk mitigation, investment optimization, inventory cost reduction, and sales and profit growth.. The data that we need in our daily lives has been increasing as each day is passing. With Sales Forecasting, we can increase the sales of the products that can produce more profits meanwhile dropping the less used products. We have taken a historical dataset which depicts sales of Walmart supermarkets to predict the upcoming sales for each of the stores. After performing the required data preprocessing, EDA and making the data suitable for the ML modelling.

II. Problem Statement

Our goal is to develop an accurate and reliable sales forecasting system that businesses can use to predict future customer behaviour. By doing this, they will be able to gauge market demand and prepare their inventory accordingly, resulting in maximum profits and minimal losses.

We must look for any missing values in the dataset. EDA of the data is another step we must take to determine the degree of feature correlation. The success of this stage will greatly affect the model's accuracy.

III. Proposed Methodology

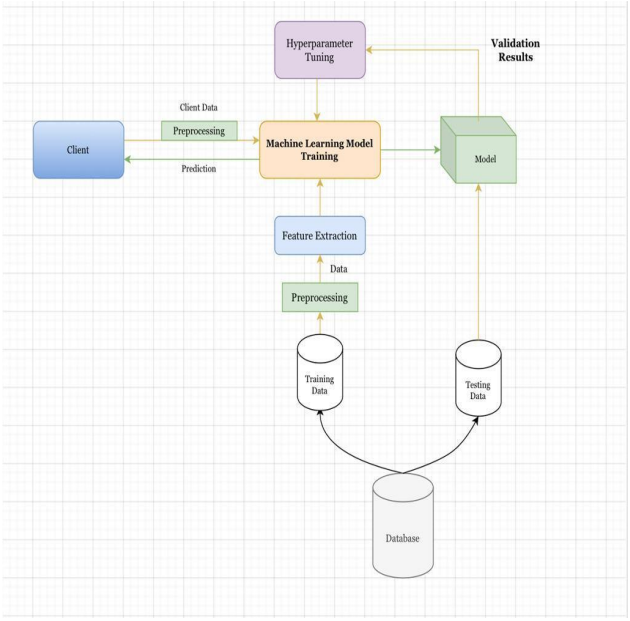
The data preparation will be the first thing we do. This entails determining the dataset's entry count and the datatypes for each column. Datetime columns must be transformed into a format that meets our requirements.

With the use of EDA, we can narrow down the features that should be taken into account for model training. This can be accomplished by removing columns that have a tenuous connection to our target variable. Input and target data frames must be created from the data.

We need to do feature scaling in cases where data points are relatively far apart to generalise the points and reduce the distance between them.
It is necessary to further partition the dataset into training and testing sets. Our machine-learning model will use the training set as input, and the testing set will be used to check the final trained model's correctness. Data splitting should be carried out before feature selection to prevent train-test leakage.

The regression-based machine learning techniques that use these features will be tested, and the best algorithm will be selected.

IV. Architecture Diagram



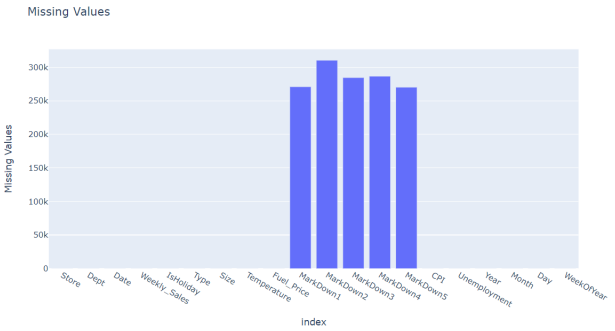
V. Data Preprocessing

We are going to use the Walmart Sales Dataset from Kaggle. It has 421,570 rows and 20 columns as shown in Table I.

#	Column	Non-Null Count	Dtype
0	Store	421570 non-null	int64
1	Dept	421570 non-null	int64
2	Date	421570 non-null	datetime64[ns]
3	Weekly_Sales	421570 non-null	float64
4	IsHoliday	421570 non-null	bool
5	Type	421570 non-null	object
6	Size	421570 non-null	int64
7	Temperature	421570 non-null	float64
8	Fuel_Price	421570 non-null	float64
9	MarkDown1	150681 non-null	float64
10	MarkDown2	111248 non-null	float64
11	MarkDown3	137091 non-null	float64
12	MarkDown4	134967 non-null	float64
13	MarkDown5	151432 non-null	float64
14	CPI	421570 non-null	float64
15	Unemployment	421570 non-null	float64
16	Year	421570 non-null	int64
17	Month	421570 non-null	int64
18	Day	421570 non-null	int64
19	WeekOfYear	421570 non-null	Float64

A. *Checking for NULL Values:*

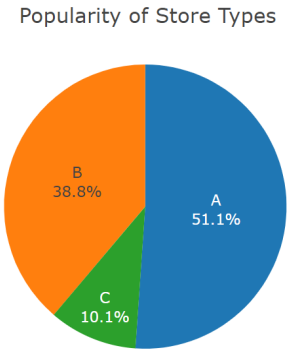
Missing values are present only in columns Markdown 1-5 columns. These amount to more than 250,000 missing values in each column. These columns represent the promotional activities being carried out in different stores. These generally occur after November and are not always available throughout the year, hence the missing values.



VI. Exploratory Data Analysis (EDA)

A. *Popularity of Store Type*

We begin by checking the popularity of different store types given in our data.

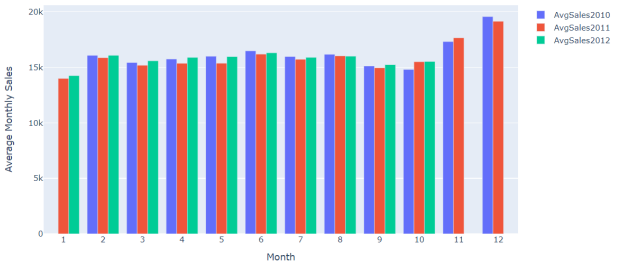


Type A stores account for more than half the total sales followed by type B and C respectively.

B. *Average Sales Based on Store Type*

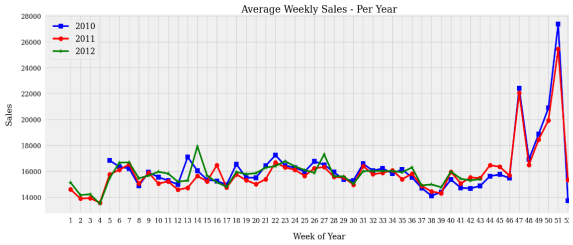
Store type A shows higher average sales than types B and C.

C. *Average Monthly Sales - Per Year*



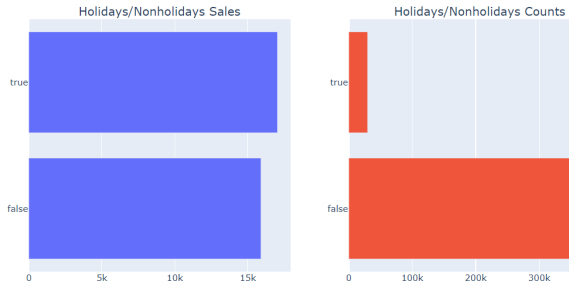
The lowest sales were seen in January of both 2011 and 2012. While sales reached their high in December 2010 and 2011. Data was not available for January 2010 and December 2012. From February through October in each of the three years, sales remained consistent.

D. *Average Weekly Sales - Per Year*



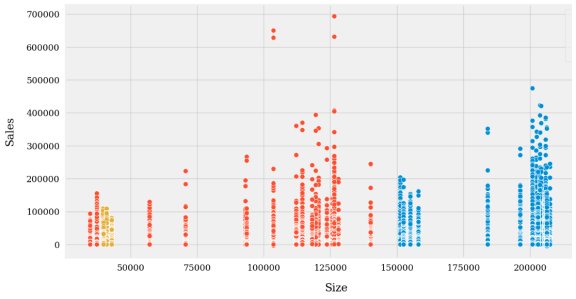
The first to fourth weeks of each year, or from January 2011 to 2012, show the lowest sales.
The week before Thanksgiving and the week before Christmas, respectively, in weeks 47 and 51, saw a rise in sales.

E. *Holiday Vs Non-Holiday Sales*



Although there are fewer holiday weeks (only 7%), on average, holiday week sales are higher than non-holiday week sales.

F. *Size of the Store Vs Sales*

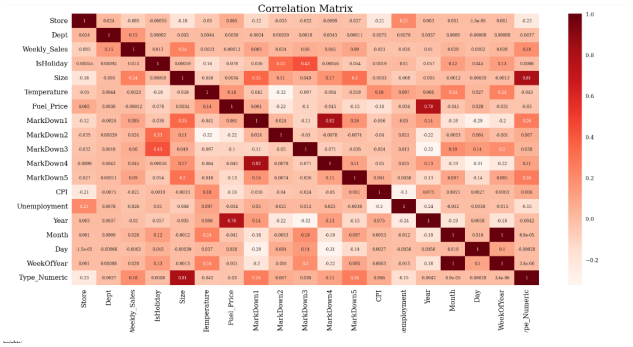


The weekly sales and store size exhibit a linear relationship. With a few exceptions, sales normally grow as store size increases.

G. *Others*

Features such as Unemployment, CPI, Fuel Price and Temperature don't show any clear relationship with our target variable.

H. *Correlation Matrix*



Department, Store Type and size have a moderate correlation with weekly sales. Markdown 1-5, Temperature, Fuel price, CPI and Unemployment are discarded since they have an insignificant correlation with our target variable. IsHoliday column will be included in our model despite its low correlation with the target variable as the sales increased in weeks before holidays as shown above.

VII. Feature Selection

The exploratory data analysis and correlation study's findings led to the decision to exclude the columns that had a weak association with the target column. The number of chosen features was decreased from 19 to 6 (excluding the target variable). These features are 'Store', 'Dept', 'IsHoliday', 'Size', 'Year'and ' WeekOfYear'.

The dataset was divided between the input and target data frames.
The column Weekly_sales serves as our target variable.
To reduce the difference between the features, MinMaxScaling is used. This brings the value of each column between 0 to 1.

Data was finally split between the training and validation set. (**Note:** Before performing any data preprocessing or feature engineering, the testing set is isolated from the training set, and all alterations are made to both sets of data independently.)

VIII. Model Training

We are going to run our data set on the following machine learning algorithms:

- Ridge Regression
- Decision Tree Regressor
- Random Forrest Regressor
- eXtreme Gradient Boosting(XGBoost)

A. *Ridge Regression*

It is a regularized version of Linear Regression. It uses L2 Regulaziation: the regularization term is added to the cost function as shown below:

$$J(\theta) = \sum_{i=1}^{m} |y_i - h_\theta(x_i)|$$

The main objective here is to keep the model weights as small as possible. The regularization term is added to the cost function during training.

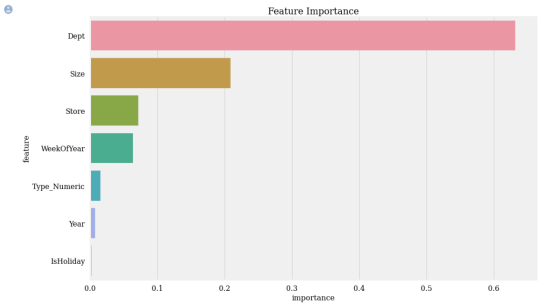
In our case, ridge regression produced really poor accuracy:

```
Training accuracy 0.0844
Validation accuracy 0.0845
```

B. *Decision Tree Regressor*

Decision tree Regressor uses a tree-like model of decisions to predict the target value. The splitting process begins at the root node and proceeds through a branching tree, ending at a leaf node (terminal node), which carries the prediction or algorithmic result. The process of building a decision tree typically proceeds top-down, with each step selecting the variable that best divides the set of objects. A binary tree can be used to represent each sub-tree of the decision tree model, where a decision node divides into two nodes depending on the circumstances.

The reduction in node impurity weighted by the probability of reaching that node is how we determine a feature's importance. The node probability can be computed by dividing the total number of samples by the number of samples that reach the node. Importance is directly proportional to this value. In our case, the feature importance for the given datasets for decision is shown below:

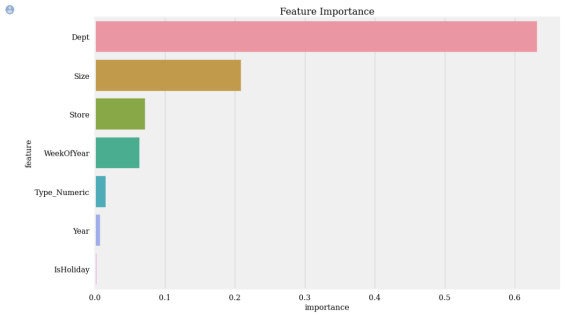


Here, ‘Department’, ‘Store Size’ and ‘Store Number’ are the top three features that determine the outcome.
Decision Tree Regressor performs well on our dataset.

```
Training accuracy 1.0000
Validation accuracy 0.9529
```

C. *Random Forest Regressor*

Random Forest is an ensemble of multiple Decision Trees trained using the bagging method. Being an ensemble technique means that they use a collection of results to draw a conclusion. For regression tasks, the mean of predictions is considered.
In our case, the feature importance for Random Forest Regressor is as follows:

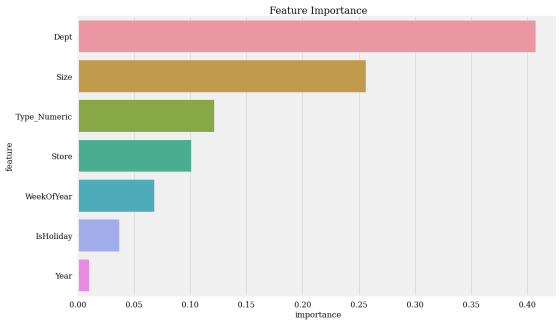


The order of feature importance is the same as Decision Tree Regressor with more importance given to the ‘Department’ feature and slightly less to the ‘Size’ feature.
Random Forest Regressor algorithm achieved training and validation accuracy values of 0.9967 and 0.9731, respectively. Even with hyperparameter tuning, we were unable to further raise the accuracy of this model.
Final Accuracy score:

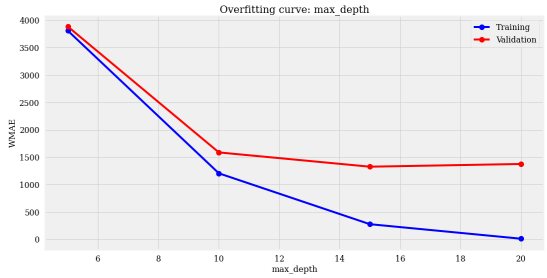
```
Training accuracy 0.9967
Validation accuracy 0.9732
```

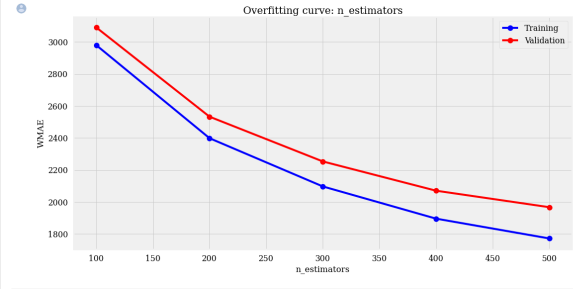
D. *eXtreme Gradient Boosting*

Gradient Boosting sequentially adds predictors to an ensemble. These predictors are constructed from Decision Trees. Each predictor's job is to correct the previous one. It does so by fitting the new predictor to the residual errors made by the previous predictors.
eXtreme Gradient Boosting (XGBoost) is an open-source implementation of the Gradient Boosting Algorithm. It is very fast, scalable, and portable. It offers features such as automatically taking care of early stopping. It is also space aware, which means missing data values are automatically handled. It also supports parallelization, i.e, the model is implemented to train with multiple CPU cores resulting in higher efficiency and speed. To avoid overfitting, XGboost also includes regularization penalties. This results in the ability of the model to generalize sufficiently. XGboost is also capable from
In our case, the feature importance for this algorithm is given below:



It is similar to Random Forest's Feature Importance. Here more importance is given to 'Type_Numeric' as compared to the Decision Tree and Random Forest.
This has a training accuracy of 0.9509 and a validation accuracy of 0.9460.
model. After studying the overfitting curves





the following values for the hyperparameters were chosen:

max_depth = 15
n_estimators = 400
learning_rate = 0.35.

The final accuracy achieved after carefully tuning the hyperparameters for XGBoost is as follows:

```
Training accuracy 1.0000
Testing accuracy 0.9834
```

IX. Conclusion

The following inferences and conclusions can be drawn from our study:

- Stores of type 'A' are more popular than types 'B' and 'C'.
- Type 'A' stores are bigger than 'B' and 'C' and perform better in terms of weekly sales.
- Sales are significantly affected by the week of the year. Weeks with holidays witnessed a greater surge in sales as compared to non-holiday weeks. These are the week leading to Thanksgiving and Christmas week.
- Weekly sales are also dependent on the size of the store.
- Different store departments showed varying amounts of sales, suggesting store departments are another determinant in sales.
- The most effective trained model for forecasting future sales is the eXtreme Gradient Boosting with tuned hyperparameters as it outperformed all other machine learning models.

Models	**Accuracy**
Ridge Regression	0.0844
Decision Tree	0.9529
Randon Forest	0.9732
XGBoost	**0.9834**

X. References

[1] E-commerce Sales Forecast Based on Ensemble Learning | IEEE Conference Publication | IEEE Xplore

[2] Walmart Sales Forecasting using XGBoost algorithm and Feature engineering | IEEE Conference Publication | IEEE Xplore

[3] Store-sales Forecasting Model to Determine Inventory Stock Levels using Machine Learning | IEEE Conference Publication | IEEE Xplore

[4] Predictive Analysis for Big Mart Sales Using Machine Learning Algorithms | IEEE Conference Publication | IEEE Xplore

[5] Grid Search Optimization (GSO) Based Future Sales Prediction for Big Mart | IEEE Conference Publication | IEEE Xplore

[6] A hybrid machine learning model for sales prediction | IEEE Conference Publication | IEEE Xplore

[7] Machine Learning Model for Sales Forecasting by Using XGBoost | IEEE Conference Publication | IEEE Xplore

[8] Forecasting of sales by using fusion of machine learning techniques | IEEE Conference Publication | IEEE Xplore

[9] For XGBoost parameter tuning [Online]. Available: http://www.analyticsvidhya.com

[10] XGBoost: Everything You Need to Know [Online]. Available on: neptune.ai