STAT 530 – APPLIED REGRESSION

DATA ANALYSIS AND FORECASTING OF WALMART SALES DATA

Group 6

DEVYANI DEORE: 47390452
HRITIK GHORPADE: 96961612
VEDIKA SONTAKKE: 15227998

PROBLEM STATEMENT AND OBJECTIVE

In the retail industry, accurate sales forecasting is critical for driving operational efficiency and customer satisfaction. Walmart, operating across 45 stores, faces the intricate challenge of predicting weekly department-level sales in a dynamic environment influenced by diverse factors. These include store-specific attributes (e.g., size, type), regional economic indicators (e.g., fuel prices, unemployment rates, consumer price index), and external variables like holidays and weather conditions.

The complexity of the task is amplified by non-linear relationships, missing data, and outliers within the dataset. To address these challenges, this project aims to leverage machine learning techniques to analyze historical sales data, uncover the influence of various features, and develop models capable of producing accurate and actionable forecasts. By enhancing the reliability of predictions, Walmart can optimize inventory management, promotional strategies, and overall store performance, ensuring data-driven decisions at scale.

Objectives

1. Forecasting Department-Level Sales: Build a predictive model to accurately estimate sales for individual departments across Walmart stores, facilitating customized strategies for inventory management and marketing.
2. Enhancing Store Efficiency: Deliver precise sales predictions to aid data-driven decisions on promotions, pricing, and stock replenishment, thereby improving store performance and operations.
3. Holistic Sales Analysis: Examine various influencing factors, including local market dynamics and seasonal trends, to generate reliable sales forecasts for each department.
4. Personalized Forecasting Solutions: Adapt sales predictions to reflect the unique characteristics of different store types and geographic regions, ensuring practical and relevant insights for each store.

ABSTRACT

This project explores the application of machine learning to predict Walmart's weekly sales, addressing the complexities of retail forecasting through a structured and data-driven approach. The dataset integrates historical sales data with features such as store attributes, regional activity metrics (e.g., temperature, fuel prices, unemployment rates), and promotional markdowns. Comprehensive preprocessing steps were undertaken, including data cleaning, merging datasets, handling missing values, and enriching the data with external factors to ensure its reliability and readiness for analysis.

To address the multifaceted nature of the data, multiple machine learning models were implemented, including Linear Regression, Decision Trees, Gradient Boosting Machines (GBM), XGBoost, Support Vector Regression (SVR), and Random Forests. Each model was trained and evaluated on its ability to predict sales accurately while handling complexities like outliers, heteroscedasticity, and non-linear relationships. Visualizations such as residual plots and QQ plots were used to assess model performance and identify areas for improvement.

This report provides a comprehensive overview of the methodology, from preprocessing to model evaluation, highlighting the challenges and insights gained during the project. By leveraging machine learning, the study demonstrates how data-driven techniques can effectively address the complexities of retail sales forecasting, supporting Walmart in optimizing inventory management, strategic planning, and operational decision-making.

INTRODUCTION

Walmart operates an extensive network of stores across the United States, offering various store formats tailored to meet the diverse needs of its customers. For this analysis, we focus on three key types of Walmart stores: Supercenters, Discount Stores, and Neighborhood Markets.

Walmart Supercenters, introduced in 1988, are the largest format, averaging around 182,000 square feet and employing approximately 300 associates per location. These stores offer an extensive range of products, including groceries, electronics, apparel, and home furnishings. Many Supercenters operate 24 hours a day and often include additional services such as banks, salons, and other specialty shops, providing a comprehensive shopping experience.

Walmart Discount Stores, launched in 1962, are smaller in size, averaging about 106,000 square feet and employing around 200 associates. While more compact than Supercenters, they still offer a wide variety of products, including electronics, apparel, and home furnishings, catering to customers seeking value in a streamlined shopping environment.

Walmart Neighborhood Markets, introduced in 1998, are the smallest format among the three, averaging 38,000 square feet and staffed by up to 95 associates. These stores focus on groceries, household supplies, and pharmacy services, making them a convenient option for quick and essential shopping needs.

Situated in a variety of locations ranging from urban centers to rural areas, each store type is designed to cater to the unique demographics and market needs of its community, ensuring that Walmart remains a vital resource for customers across the nation.

DATASET OVERVIEW

The Walmart sales dataset provides historical data from 45 stores, distributed across several CSV files, each containing unique aspects of the stores, their characteristics, and sales performance. Below is a breakdown of the key datasets and the information they contain:

stores.csv

This file contains details about Walmart's store-specific characteristics, offering valuable insights into the factors that influence sales at each location:

- **Store (Store ID):** A unique identifier for each Walmart store, enabling granular analysis of sales trends and performance at individual locations.
- **Type:** Categorizes stores into formats such as Supercentres, Discount Stores, and Neighbourhood Markets, reflecting differences in customer demographics and shopping behaviors.
- **Size:** Indicates the physical size of the store, typically in square feet. Larger stores often offer a broader product range, potentially impacting customer footfall and sales.

walmart_data.csv

This file serves as the core historical dataset for sales forecasting, spanning from February 5, 2010, to November 1, 2012. It includes the following details:

- **Store:** The unique ID of the store.
- **Dept:** Department number within each store.
- **Date:** The specific week for which sales data is recorded.
- **Weekly_Sales:** Total sales for the given department and store during the specified week.
- **IsHoliday:** A flag indicating whether the week coincides with a holiday, which can significantly impact sales patterns.

features.csv

This dataset augments the sales and store data by providing external and regional factors that may influence sales performance. These features include:

- **Store:** The store number for linking with other datasets.
- **Date:** The corresponding week for the provided data.
- **Temperature:** Average temperature in the store's region.
- **Fuel_Price:** Regional fuel prices, which may impact customer purchasing behavior.
- **MarkDown-5:** Anonymized data related to promotional markdowns, reflecting sales strategies.
- **CPI (Consumer Price Index):** An economic indicator that measures changes in the cost of living.
- **Unemployment:** Regional unemployment rates, offering insights into local economic conditions.
- **IsHoliday:** A holiday indicator flag, signaling weeks with potential spikes in sales due to special events.

Description of Dataset

```
> str(stores)
'data.frame':   45 obs. of  3 variables:
 $ Store: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Type : chr  "A" "A" "B" "A" ...
 $ Size : int  151315 202307 37392 205863 34875 202505 70713 155078 125833 126512 ...
```

Data Preprossesing

The data preprocessing stage involved consolidating information from multiple CSV files to form a cohesive dataset for analysis. Specifically, the 'stores.csv' file was merged with the primary Walmart dataset based on the 'store type' attribute, while 'features.csv' was combined using 'store ID' and 'date of the week'. This integration enhanced the sales data by incorporating store-specific characteristics and external factors potentially influencing sales trends.

Following the merging process, data quality issues were addressed by eliminating null values from both 'stores.csv' and 'features.csv'. Since null entries can distort predictions and lead to inaccuracies, their removal was prioritized. Data imputation methods were applied selectively to retain critical information while preserving the dataset's reliability. In cases where null values were extensive or the data deemed non-essential, removal was considered to minimize information loss. The resulting cleaned dataset provides a reliable basis for developing robust analytical models.

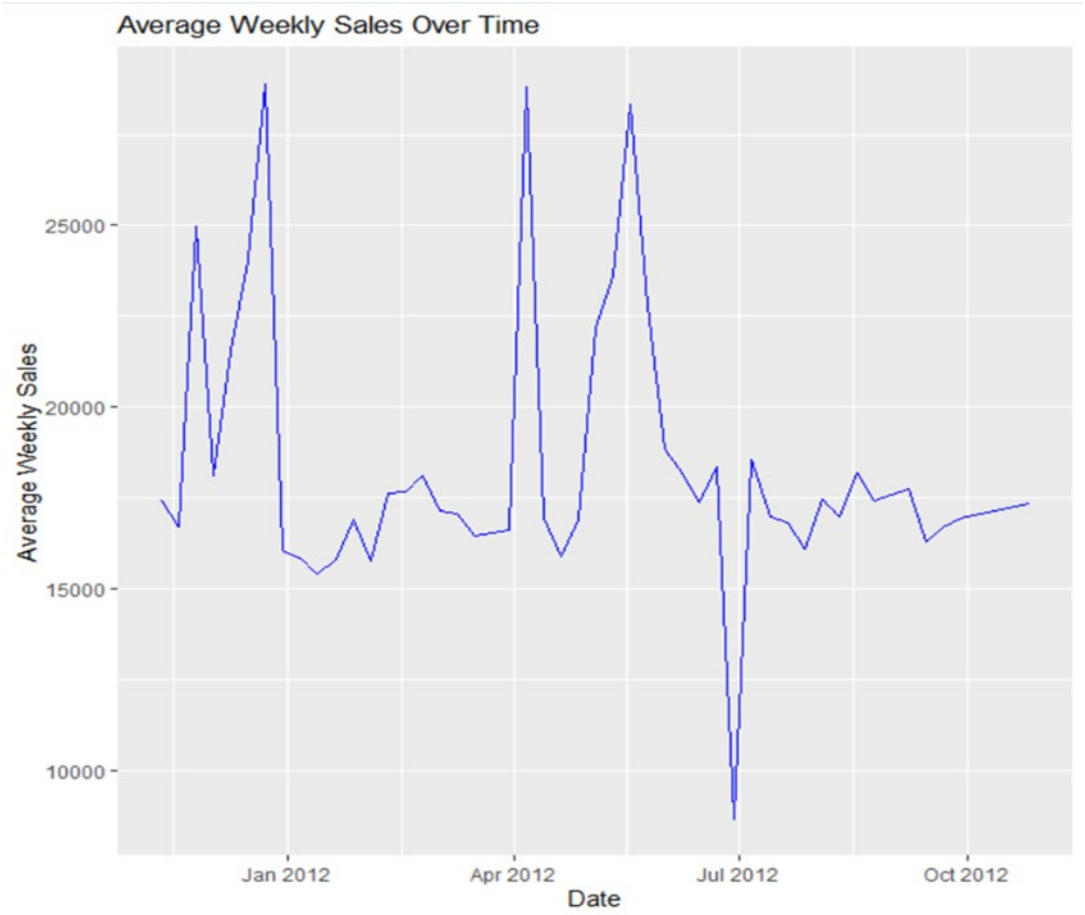
```
> # Summarize the merged dataset
> summary(train)
     Store            Dept            Date             Weekly_Sales
 Min.   : 1.00   Min.   : 1.00   Min.   :2011-11-11   Min.   : -1699
 1st Qu.:10.00   1st Qu.:19.00   1st Qu.:2012-01-20   1st Qu.:  2764
 Median :20.00   Median :37.00   Median :2012-03-30   Median :  8622
 Mean   :20.24   Mean   :44.28   Mean   :2012-04-16   Mean   : 17857
 3rd Qu.:29.00   3rd Qu.:72.00   3rd Qu.:2012-07-20   3rd Qu.: 22741
 Max.   :45.00   Max.   :99.00   Max.   :2012-10-26   Max.   :630999
 IsHoliday.x         Type                Size          Temperature
 Mode :logical   Length:97056       Min.   : 34875   Min.   : 7.46
 FALSE:87064     Class :character   1st Qu.:119557   1st Qu.:42.75
 TRUE :9992      Mode  :character   Median :155083   Median :57.95
                                    Mean   :155229   Mean   :57.35
                                    3rd Qu.:203742   3rd Qu.:72.66
                                    Max.   :219622   Max.   :95.91
   Fuel_Price      MarkDown1         MarkDown2           MarkDown3
 Min.   :3.031   Min.   :   32.5   Min.   :  -265.76   Min.   :   -29.10
 1st Qu.:3.413   1st Qu.: 3600.8   1st Qu.:    47.55   1st Qu.:     5.40
 Median :3.630   Median : 6264.2   Median :   192.00   Median :    30.46
 Mean   :3.619   Mean   : 8841.3   Mean   :  3693.53   Mean   :  1816.63
 3rd Qu.:3.820   3rd Qu.:10333.2   3rd Qu.:  2551.32   3rd Qu.:   123.42
 Max.   :4.301   Max.   :88646.8   Max.   :104519.54   Max.   :141630.61
   MarkDown4          MarkDown5             CPI          Unemployment
 Min.   :    0.46   Min.   :   170.6   Min.   :129.8   Min.   : 4.077
 1st Qu.:  605.88   1st Qu.:  2383.7   1st Qu.:136.9   1st Qu.: 6.392
 Median : 1739.83   Median :  3864.6   Median :189.2   Median : 7.280
 Mean   : 4025.92   Mean   :  5310.8   Mean   :174.8   Mean   : 7.415
 3rd Qu.: 4082.99   3rd Qu.:  6197.5   3rd Qu.:219.4   3rd Qu.: 8.256
 Max.   :67474.85   Max.   :108519.3   Max.   :227.0   Max.   :12.890
 IsHoliday.y
 Mode :logical
 FALSE:87064
 TRUE :9992
```

Data visualization

Data visualization is a fundamental step in understanding and analyzing the Walmart sales dataset, offering a visual representation of the patterns and relationships within the data. By employing techniques such as time series plots, box plots, and histograms, we were able to extract meaningful insights that guide our analysis and forecasting efforts. These visualizations not only validate our

assumptions but also provide actionable insights that enhance the interpretability of the dataset. Below, we delve deeper into each visualization technique and its significance:

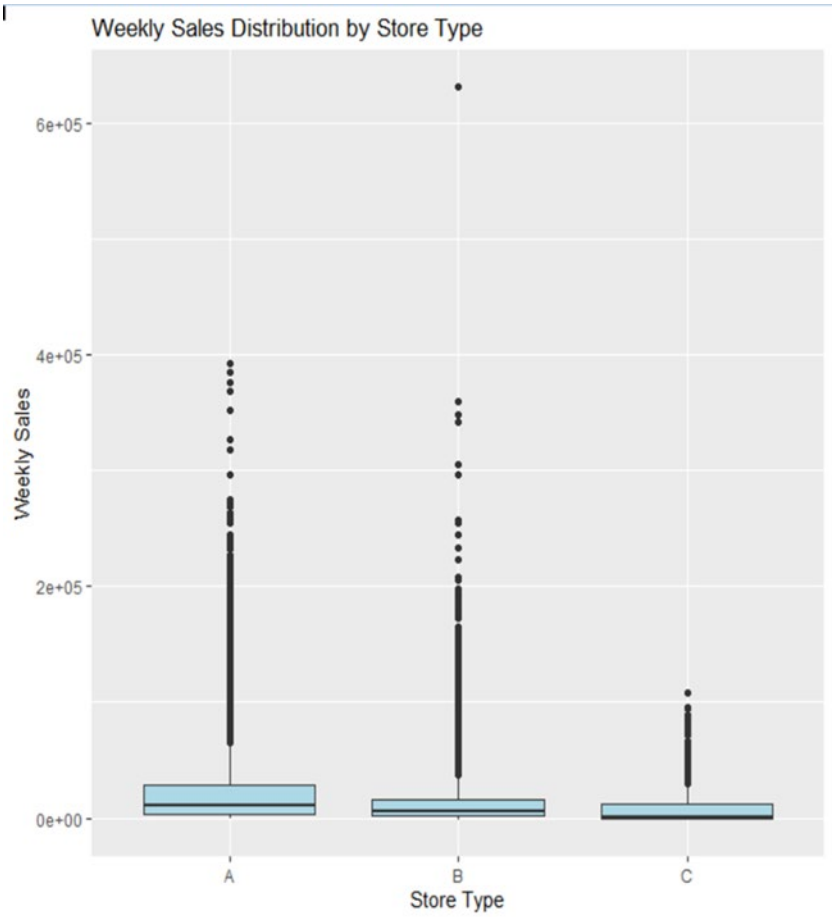
Time Series Plot:



The time series plot of Walmart's average monthly sales stands out as a pivotal visualization in this project. This graph reveals distinct seasonal patterns, with pronounced sales spikes during key periods, particularly around November and December, aligning with major holidays like Thanksgiving and Christmas. These spikes are followed by sharp declines, reflecting a return to typical consumer spending levels after the holiday shopping frenzy.

This visualization is invaluable for understanding the cyclical nature of sales trends, underscoring the need to account for seasonality in predictive models. By highlighting these patterns, the time series plot provides a compelling narrative of consumer behavior and validates assumptions about peak shopping periods. Additionally, it reinforces the importance of tailoring inventory and promotional strategies to capitalize on these seasonal peaks, ensuring that Walmart is well-prepared to meet consumer demand during high-traffic periods.

Box plot:



The boxplot illustrates the distribution of weekly sales across three store types: A, B, and C. For each store type, the box represents the interquartile range (IQR), capturing the middle 50% of the sales data, with the horizontal line inside the box indicating the median weekly sales. The "whiskers" extend to cover the range of data within 1.5 times the IQR, while any points beyond the whiskers are considered outliers and displayed as dots. Store Type A shows the widest distribution of weekly sales, with several outliers reaching values above 600,000. Store Type B has a similar distribution to Store Type A but with slightly less extreme outliers. Store Type C exhibits the narrowest range of sales and relatively fewer extreme outliers. This visualization highlights variability in sales performance, with Store Types A and B demonstrating higher variability compared to Store Type C.

Methodology

Models Implementation

The analysis of Walmart's sales data required the application of various machine learning models, each designed to capture different aspects of the data and address specific challenges. These models were implemented to forecast weekly sales while accounting for the complexities of seasonal trends, regional economic indicators, and store-specific characteristics.

The implemented models include -
1) Linear Regression
2) Decision Trees
3) Gradient Boosting Machines (GBM)
4) XGBoost
5) Support Vector Regression
6) Random Forests.

Each model brings unique strengths to the forecasting task, such as handling non-linear relationships, reducing overfitting, or capturing feature importance.

Through these models, we aimed to achieve accurate predictions, identify critical features influencing sales, and develop insights to support strategic decision-making. Visualizations and error metrics were used to assess their performance, ensuring a robust evaluation process. The subsequent sections delve into the implementation and evaluation of each model.

Model 1: Linear Regression

Linear regression is a statistical technique used to model the relationship between a dependent variable and one or more independent variables. In this analysis, the goal is to predict the continuous variable 'Weekly_sales' based on various features.

Target Variable: 'Weekly_sales' is a continuous variable that can take any numerical value. The relationship is modelled using the equation:

$$Y=\beta_0+\beta_1X_1+\beta_2X_2+\cdots+\beta_nX_n+\epsilon$$

Here:

Y: Represents 'Weekly_sales', the variable being predicted.

β_0: The intercept, denoting the expected value of Y when all independent variables (X_1,X_2,…..) are zero.

β_1,β_2,…,β_n: Coefficients that quantify the effect of each feature (X1,X2,…X_1, X_2, \dots) on sales.

ϵ: The error term, accounting for discrepancies between the actual and predicted values.

This approach enables the identification of key features that significantly influence sales, providing a foundation for accurate forecasting and actionable insights.

```
> # Train Linear Regression Model
> lm_model <- lm(Weekly_Sales ~ Store + Dept + IsHoliday + Temperature + Fuel_Price + CPI + Unemployment + Type + Size,
+                data = train_data)
>
> # Summarize the model
> summary(lm_model)

Call:
lm(formula = Weekly_Sales ~ Store + Dept + IsHoliday + Temperature +
    Fuel_Price + CPI + Unemployment + Type + Size, data = train_data)

Residuals:
   Min     1Q Median     3Q    Max
-36296 -14365  -6733   6424 608791

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   15426.650    260.547  59.209  < 2e-16 ***
Store          -170.545      7.591 -22.466  < 2e-16 ***
Dept            111.147      2.835  39.210  < 2e-16 ***
IsHolidayTRUE  1308.695    286.478   4.568 4.93e-06 ***
Temperature       5.719     95.432   0.060    0.952
Fuel_Price     -423.545    101.560  -4.170 3.04e-05 ***
CPI           -1181.914     99.967 -11.823  < 2e-16 ***
Unemployment   -837.331     92.764  -9.026  < 2e-16 ***
TypeB          1826.633    351.025   5.204 1.96e-07 ***
TypeC         10187.751   1060.105   9.610  < 2e-16 ***
Size           6333.599    174.357  36.326  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 23790 on 77637 degrees of freedom
Multiple R-squared:  0.07318,   Adjusted R-squared:  0.07307
F-statistic: 613.1 on 10 and 77637 DF,  p-value: < 2.2e-16
```

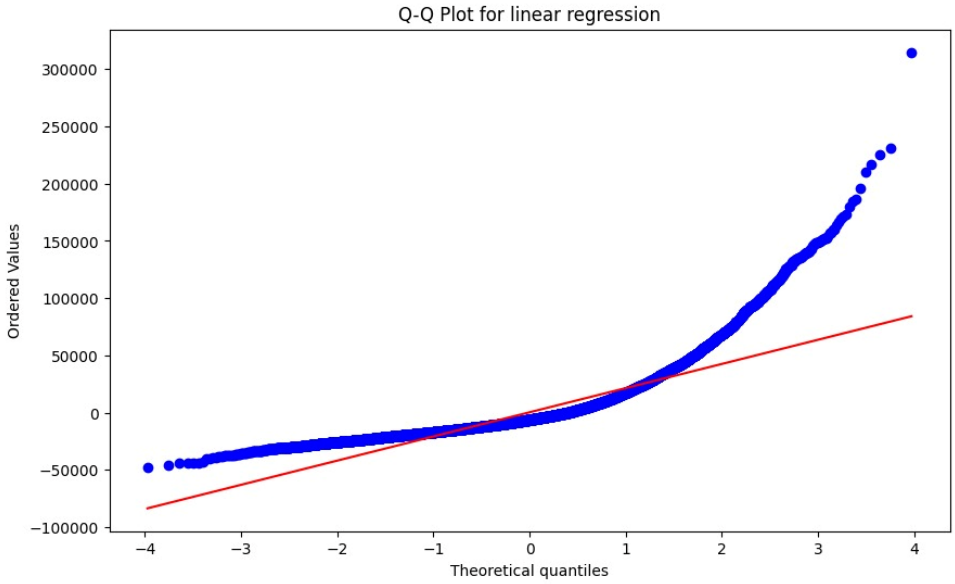
```
Analysis of Variance Table

Model 1: Weekly_Sales ~ Store + Dept + IsHoliday
Model 2: Weekly_Sales ~ X + Store + Dept + Date + IsHoliday + Type + Size +
    Temperature + Fuel_Price + MarkDown1 + MarkDown2 + MarkDown3 +
    MarkDown4 + MarkDown5 + CPI + Unemployment
  Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
1  97052 5.8059e+13
2  96993 5.4239e+13 59 3.8198e+12 115.77 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> qf(1-0.05, 59, 97993)
[1] 1.320997
```

Hypothesis Testing:

1) H0: $\beta 1 = \beta 4 = \beta 8 = \beta 10 = \beta 11 = \beta 12 = \beta 13 = \beta 14 = \beta 15 = \beta 16 = \beta 17 = 0$
2) Ha: not H0 (at least one non- zero)
3) Statistics: Fobs = 115.77
4) Rejection region: F>F0.0.5 (59,97993) = 1.320997
5) Conclusion: Since Fobs > F in the rejection region. We reject H0. So at least one among the features omitted is non-zero, making significant impact on the model

QQ-Plot



The Q-Q plot evaluates the normality of residuals from a linear regression model by comparing the observed quantiles of residuals (blue dots) against the theoretical quantiles of a normal distribution (red line). Ideally, the points should align closely with the red line if the residuals follow a normal distribution. However, in this plot, significant deviations are observed, particularly at the extremes, indicating heavy tails and suggesting that the residuals are not normally distributed. This pattern highlights potential skewness or kurtosis in the data. Such violations of normality can affect the validity of statistical tests and confidence intervals in the regression analysis. To address these issues, it may be necessary to consider transformations of the dependent variable, robust regression techniques, or further diagnostics for heteroscedasticity and influential points.

Model 2: Decision Trees

A Decision Tree is a supervised learning algorithm used for regression tasks to predict continuous variables like Weekly_Sales. It splits data based on feature significance, creating a tree structure where nodes represent features, and leaves represent predicted values.

The decision tree model has a tree-like structure, with nodes standing in for decision points and branches for potential outcomes of these decisions, which eventually result in predictions.

The root node, which stands for the complete dataset, is where the process starts. The data is then separated into subgroups, or nodes, according to particular feature values. The algorithm chooses a feature and a threshold value at each node that minimize the variation in the target variable (in this case, Weekly_Sales) across the subsets that are produced. The final predictions are represented by terminal nodes, or leaves, which are produced after this splitting process is completed and a predetermined stopping criterion is satisfied.

A strong method for forecasting numerical results like Weekly_Sales is to use a decision tree regression model, particularly with the part package in R. It is a useful tool for analysis because of its interpretability and visualization capabilities.

Advantages this model are Captures non-linear relationships. Interpretable structure. Limitations for this model are Prone to overfitting without pruning and Challenges with extreme predictions, as indicated in residual plots.

```
> summary(tree_model)
Call:
rpart(formula = Weekly_Sales ~ Store + Dept + IsHoliday + Temperature +
    Fuel_Price + CPI + Unemployment + Type + Size, data = train_data)
  n= 77648

          CP nsplit rel error    xerror       xstd
1 0.11579646      0 1.0000000 1.0000139 0.01663774
2 0.10043399      1 0.8842035 0.8842542 0.01515802
3 0.04180331      3 0.6833356 0.6834568 0.01374458
4 0.03639573      4 0.6415322 0.6416804 0.01377109
5 0.02258611      9 0.4595536 0.4597392 0.01090790
6 0.01000000     12 0.3917953 0.3920264 0.01038197

Variable importance
 Dept  Size  Type Store
   67    15    15     1
```

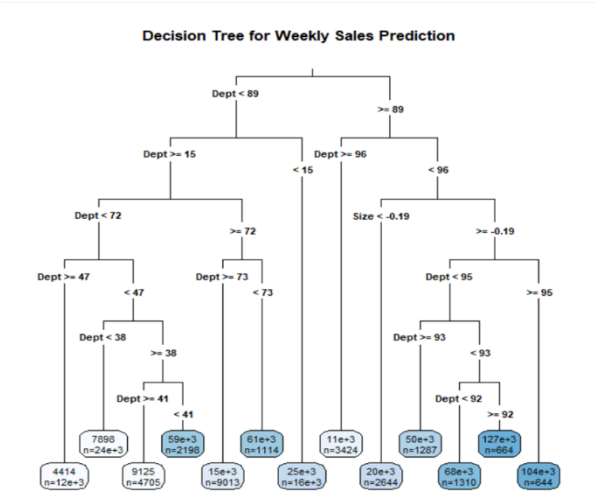
```
> # Output Metrics
> cat("Decision Tree MAE:", tree_mae, "\n")
Decision Tree MAE: 9986.829
> cat("Decision Tree MSE:", tree_mse, "\n")
Decision Tree MSE: 250941772
>
```

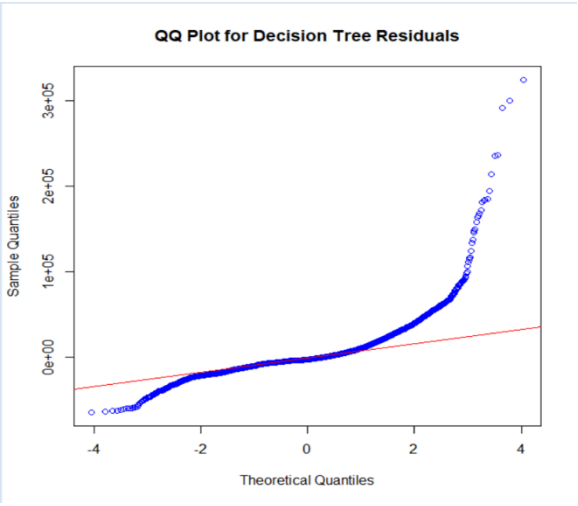
Decision Tree for Weekly Sales Prediction

The decision tree displayed in the first figure is used to predict weekly sales, with the root node representing the entire dataset. The tree structure comprises decision nodes, where the data is split based on specific features such as Dept and Size, and terminal nodes (leaves) that provide the predicted sales values. The splitting criteria at each node aim to minimize variance in the target variable (Weekly_Sales). The terminal nodes also show the predicted sales values and the number of data points associated with each prediction.

Residual Analysis of the Decision Tree Model

The QQ plot in the second figure visualizes the residuals of the decision tree model. Ideally, residuals should align with the red reference line, representing a normal distribution of errors. However, deviations observed, particularly at the extremes, suggest that the model has challenges in capturing the variance in certain segments of the data. This indicates potential areas for model refinement or the need for additional features to improve predictive performance.


Decision Tree for Weekly Sales Prediction
Dept < 89
>= 89
Dept >= 15
< 15
Dept >= 96
< 96
Dept < 72
>= 72
Size < -0.19
>= -0.19
Dept >= 47
< 47
Dept >= 73
< 73
Dept < 95
>= 95
Dept < 38
>= 38
Dept >= 93
< 93
Dept >= 41
< 41
Dept < 92
>= 92
7898 n=24e+3
59e+3 n=2198
61e+3 n=1114
11e+3 n=3424
50e+3 n=1287
127e+3 n=664
4414 n=12e+3
9125 n=4705
15e+3 n=9013
25e+3 n=16e+3
20e+3 n=2644
68e+3 n=1310
104e+3 n=644


QQ Plot for Decision Tree Residuals
Sample Quantiles
0e+00
1e+05
2e+05
3e+05
-4
-2
0
2
4
Theoretical Quantiles

Model 3: Gradient Boosting Machines (GBM)

Gradient Boosting Machine (GBM) is an ensemble learning method that builds multiple weak learners (typically Decision Trees) sequentially to correct the errors of previous models and improve overall accuracy. GBM is widely used for regression tasks due to its robustness and ability to handle non-linear relationships.

To implement GBM using R, libraries such as gbm or caret can be used. After loading the package and importing the dataset, the model is trained by specifying key parameters, such as the number of trees (n.trees) and the learning rate. The model iteratively adjusts weights to minimize the residual errors of prior trees. Once trained, predictions are generated using the predict() function. The aggregation of sequential trees helps in creating a strong predictive model.

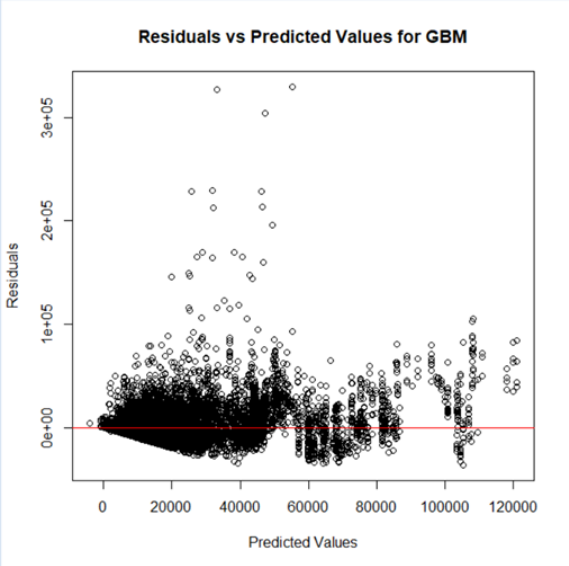
```
> # Train the Gradient Boosting Model
> gbm_model <- gbm(Weekly_Sales ~ Store + Dept + IsHoliday + Temperature + Fuel_Price + CPI + Unemployment + Type + Size,
+                  data = train_data, distribution = "gaussian", n.trees = 100, interaction.depth = 3, shrinkage = 0.1)
>
> # Make Predictions
> gbm_predictions <- predict(gbm_model, test_data, n.trees = 100)
>
> # Evaluate the Model
> gbm_mae <- mae(test_data$Weekly_Sales, gbm_predictions)
> gbm_mse <- mse(test_data$Weekly_Sales, gbm_predictions)
>
> # Output Metrics
> cat("GBM MAE:", gbm_mae, "\n")
GBM MAE: 8614.918
> cat("GBM MSE:", gbm_mse, "\n")
GBM MSE: 209257471
> summary(gbm_model)
                       var    rel.inf
Dept                  Dept 79.1844538
Size                  Size 16.6906757
Store                Store  1.8910078
CPI                    CPI  1.0513665
Type                  Type  0.6920627
Unemployment Unemployment  0.4904333
IsHoliday       IsHoliday  0.0000000
Temperature   Temperature  0.0000000
Fuel_Price     Fuel_Price  0.0000000
>
```

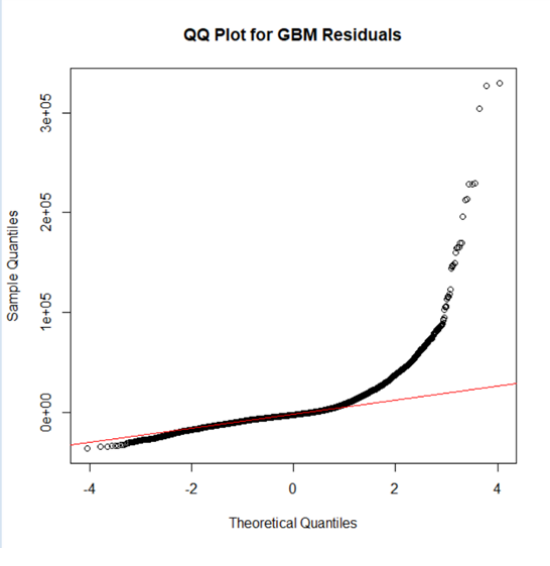
GBM provides strong predictive power and flexibility, making it well-suited for complex regression problems while offering interpretability through feature importance and residual analysis. GBM is an ensemble learning method that iteratively builds decision trees to minimize prediction errors, making it effective for predicting Weekly_Sales.

Model Parameters:
Trees = 100 trees, depth = 3, shrinkage = 0.1

Performance Metrics:
Mean Absolute Error (MAE): 6,614.92
Mean Squared Error (MSE): 209,257,471

Feature Importance:
Dept (79.18%): Dominates prediction.
Size (16.69%): Secondary contributor.
Minor impact from other features (e.g., CPI: 1.05%).





The Residual vs. Predicted Values plot shows that residuals are closer to zero for moderate predictions, indicating the model performs well for these values. As predicted values increase, residuals spread wider, suggesting the model struggles with higher sales predictions and generates larger errors.

The QQ Plot compares the residuals to a theoretical normal distribution. Significant deviations from the red diagonal line at both ends suggest:

- The presence of outliers in the data.
- Residuals are not normally distributed, indicating unmodeled patterns or heteroscedasticity.

GBM effectively captures complex patterns, with high accuracy for moderate predictions but struggles with extreme values. Tuning can improve performance further.

Model 4: XGBoost

XGBoost (Extreme Gradient Boosting) is a powerful and scalable ensemble learning algorithm based on gradient boosting. It enhances traditional gradient boosting by incorporating advanced features such as regularization, parallel processing, and tree pruning to improve performance and reduce overfitting. XGBoost is widely used for both regression and classification tasks due to its high efficiency and accuracy.

To implement XGBoost for regression in R, the xgboost package is used. The dataset is first preprocessed and converted into a DMatrix format, which optimizes memory and computation. Key parameters such as nrounds (number of boosting iterations) and eta (learning rate) are set to control the model training. The model minimizes a specified loss function (e.g., mean squared error) to improve predictions iteratively. After training, predictions are generated using the predict() function, which combines results from all boosted trees.

XGBoost's ability to handle missing data, regularization features, and parallel processing makes it a highly robust and efficient choice for regression tasks. Its interpretability is enhanced through tools like feature importance and residual analysis, providing insights into model performance and influential predictors.

XGBoost builds multiple decision trees sequentially to minimize the objective function, defined as:
Objective Function = Loss Function + Regularization Term
Loss Function: Mean Squared Error (MSE) for regression tasks:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Here, y_i represents the actual values, $\hat{y}_i$ represents the predicted values, and n is the number of observations.

Regularization Term: Penalizes complex models to prevent overfitting.

```
> # Make Predictions
> xgb_predictions <- predict(xgb_model, test_matrix)
>
> # Evaluate the Model
> xgb_mae <- mae(test_data$Weekly_Sales, xgb_predictions)
> xgb_mse <- mse(test_data$Weekly_Sales, xgb_predictions)
>
> # Output Metrics
> cat("XGBoost MAE:", xgb_mae, "\n")
XGBoost MAE: 3595.233
> cat("XGBoost MSE:", xgb_mse, "\n")
XGBoost MSE: 47535205
>
```

Performance Metrics:

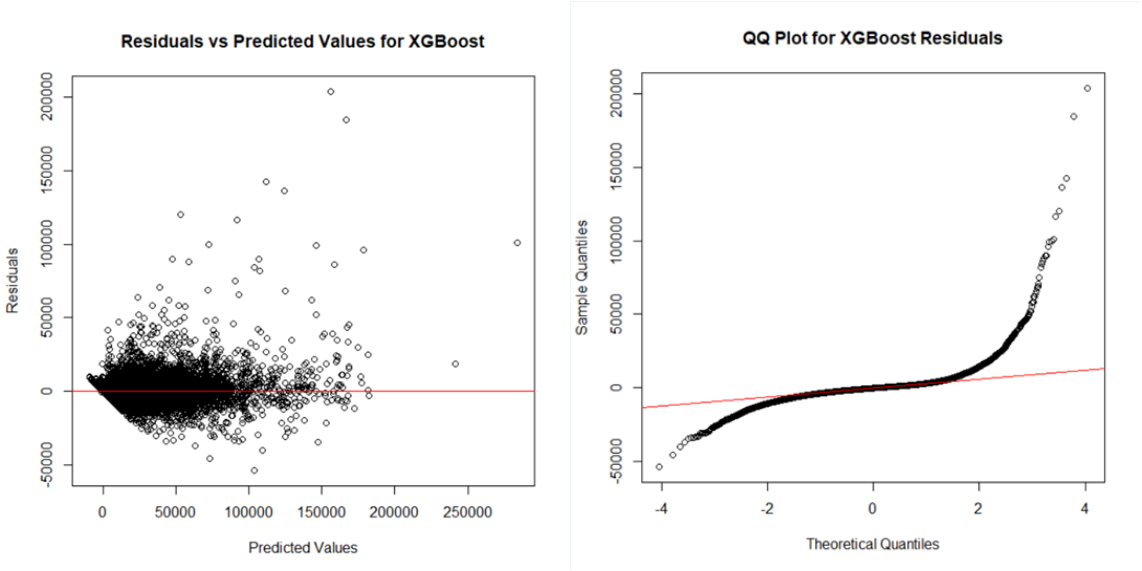
Mean Absolute Error (MAE): 3,595.233
Mean Squared Error (MSE): 47,535,205

Residual Plot:

- Most residuals cluster near zero, indicating accurate predictions for average cases.
- Outliers are present, particularly for extreme predicted values, highlighting areas for improvement.

QQ Plot:

- Residuals deviate from the red diagonal line at the extremes, indicating non-normality and unmodeled patterns.
- Suggests potential challenges in handling outliers or extreme values.



Model 5: Support Vector Regression

Support Vector Regression (SVR) is a machine learning algorithm designed to handle non-linear relationships and outliers effectively.

By using the kernel trick, it maps input features to a higher-dimensional space, allowing it to model complex patterns in data. In this analysis, SVR is applied to predict Weekly_Sales.

SVR aims to fit a hyperplane in the high-dimensional space to predict sales values while minimizing deviations within a specified margin (ε). The model uses the Radial Basis Function (RBF) kernel for non-linear mapping:

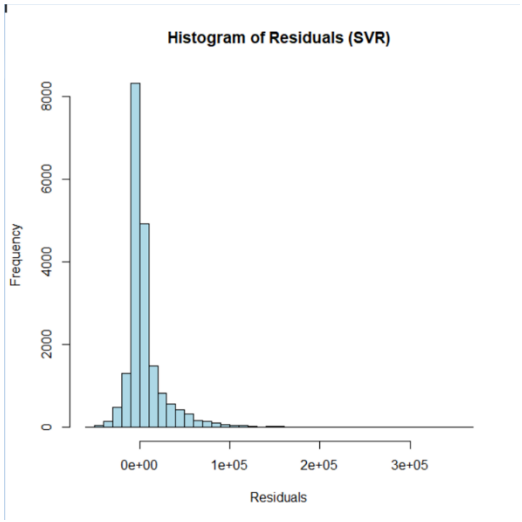
$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + b$$

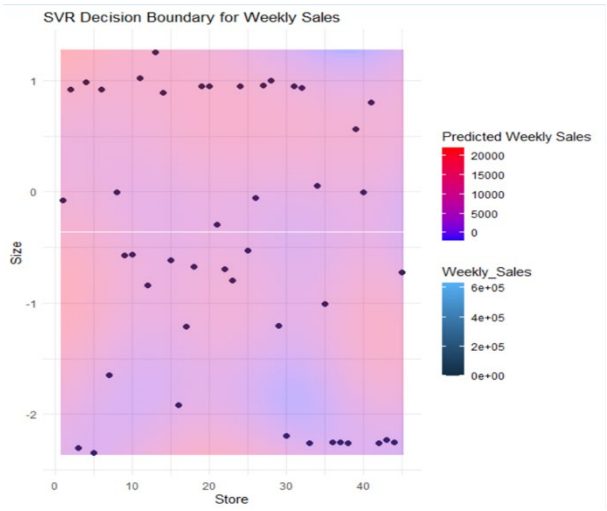
Where:

- $K(x_i, x)$: RBF kernel function.
- α_i, α_i^*: Lagrange multipliers.
- b: Bias term.

```
> # Train the Support Vector Regression Model
> svr_model <- svm(Weekly_Sales ~ Store + Dept + IsHoliday + Temperature + Fuel_Price + CPI + Unemployment + Type + Size,
+                  data = train_data, type = "eps-regression", kernel = "radial")
>
> # Make Predictions
> svr_predictions <- predict(svr_model, test_data)
>
> # Evaluate the Model
> svr_mae <- mae(test_data$Weekly_Sales, svr_predictions)
> svr_mse <- mse(test_data$Weekly_Sales, svr_predictions)
>
> # Output Metrics
> cat("SVR MAE:", svr_mae, "\n")
SVR MAE: 12222.02
> cat("SVR MSE:", svr_mse, "\n")
SVR MSE: 512806350
```

Performance Metrics:
Mean Absolute Error (MAE): 12,222.02
Mean Squared Error (MSE): 51,280,635





Histogram of Residuals:
Residuals are concentrated near zero, indicating good performance for most predictions.
Long tails suggest challenges in capturing extreme values.

Decision Boundary Visualization:
The color gradient represents predicted sales across combinations of Store and Size. Clear decision boundaries highlight how the model maps predictions to the feature space.

SVR excels in handling non-linear relationships when combined with kernels. Its margin-based approach ensures robust predictions and good generalization for regression tasks. However, it may require careful tuning of hyperparameters for optimal performance.

Model 6: Random Forests.

Random Forests is an ensemble learning method that builds multiple decision trees and aggregates their predictions to improve accuracy and reduce overfitting. It works effectively for regression tasks like predicting Weekly_Sales by averaging the outcomes of individual trees.

How it Works

Bootstrapping and Feature Sampling: The algorithm creates multiple decision trees using bootstrapped subsets of the data and randomly selects a subset of features at each split to ensure diversity among the trees.

Training Individual Trees: Each tree is trained independently on its respective subset of data, focusing on minimizing variance within the target variable across splits.

Aggregation: For regression tasks, the predictions from all trees are averaged to generate the final output, ensuring a more accurate and stable prediction.

Random Forests reduces variance by combining predictions from multiple decision trees:

$$\hat{y} = \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t$$

Where:

- $\hat{y}_t$: Prediction from the t-th tree.
- T: Total number of trees.

Random Forest Regression builds multiple decision trees during training and combines their predictions by averaging the outputs for regression tasks. To implement it using the randomForest package in R, start by loading the package and importing your dataset. Set key parameters, such as n_estimators, to specify the number of trees (e.g., n_estimators = 100).

Once the model is trained, predictions on new data can be made using the predict() function. This ensemble approach enhances accuracy and robustness by aggregating the results of multiple decision trees, making it ideal for regression tasks. Additionally, tools like feature importance plots can be used to visualize and interpret the factors that influence predictions.

Random Forest

```
> # Make Predictions
> rf_predictions <- predict(rf_model, test_data)
>
> # Evaluate the Model
> rf_mae <- mae(test_data$Weekly_Sales, rf_predictions)
> rf_mse <- mse(test_data$Weekly_Sales, rf_predictions)
>
> # Output Metrics
> cat("Random Forest MAE:", rf_mae, "\n")
Random Forest MAE: 6049.58
> cat("Random Forest MSE:", rf_mse, "\n")
Random Forest MSE: 118756196
```

```
> # Residuals vs Predicted Plot
> plot(rf_predictions, rf_residuals,
+      xlab = "Predicted Values", ylab = "Residuals",
+      main = "Residuals vs Predicted Values for Random Forest")
> abline(h = 0, col = "red")  # Add a reference line at 0
>
```

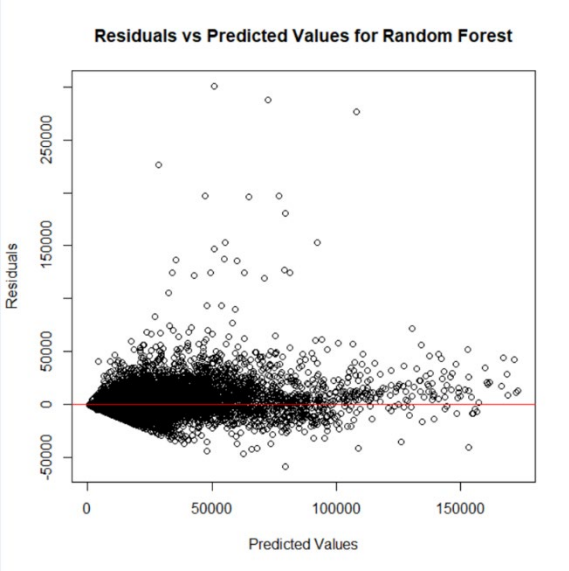
Performance Metrics:

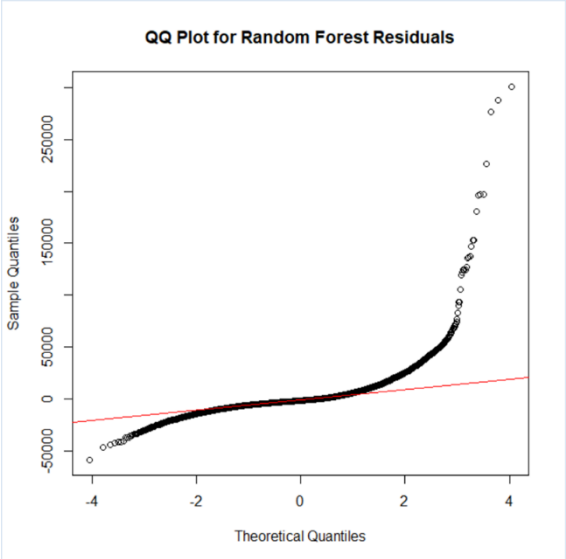
Mean Absolute Error (MAE): 6,049.58
Mean Squared Error (MSE): 118,756,196

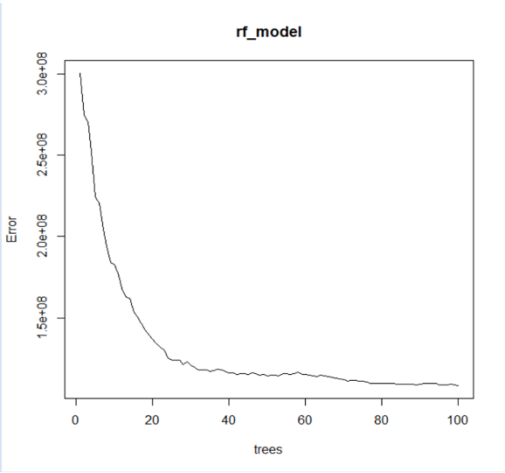
Residuals vs Predicted Values:
Residuals cluster around zero for moderate predictions, indicating good accuracy.
Errors increase for higher predicted values, suggesting challenges with extreme cases.

QQ Plot:
Residuals align well with the theoretical normal distribution for moderate values.
Deviations at the tails suggest the presence of outliers or difficulties with extreme predictions.

Error vs. Number of Trees:
The error decreases as more trees are added, stabilizing after around 50 trees.
This indicates diminishing returns for additional trees beyond this point.







The Random Forest algorithm was chosen for its ability to handle non-linear relationships, manage missing data effectively, and deliver accurate predictions for the target variable (Weekly_Sales).

Model Comparison:

Model	MAE	MSE
Linear Regression	16234.62	573582503
Decision Tree	9986.829	250941772
GBM	8614.918	209257471
XGBoost	3595.233	47535205
SVR	12,222.02	5,12,06,350
Random forest	6049.58	118756196

XGBoost: Best performance with the lowest MAE (3595.233) and MSE (47,535,205), making it the most accurate model.

Random Forests: Strong results with good accuracy for moderate predictions; slightly higher errors for extreme values.

GBM: Decent performance but struggled with extreme predictions compared to ensemble models like XGBoost.

Decision Tree: Reasonable accuracy but less robust than advanced ensemble methods.

SVR: Higher errors (MAE: 12,222.02) due to sensitivity to outliers and extreme residuals.

Linear Regression: Weakest performance; unable to handle non-linear relationships or complex patterns in the data.

Conclusion:

This project successfully explored and implemented various machine learning models to address the complexities of sales forecasting for Walmart stores. By leveraging advanced techniques like XGBoost, Random Forests, and Gradient Boosting Machines, the study demonstrated how ensemble methods outperform traditional models in accuracy and robustness, especially when handling non-linear relationships and diverse influencing factors.

The analysis revealed that XGBoost provided the most precise predictions, achieving the lowest mean absolute error and mean squared error among all models. Random Forests and Gradient Boosting Machines also delivered strong results, showcasing their capability to capture complex patterns and provide actionable insights. In contrast, simpler models like Linear Regression and Decision Trees were limited by their inability to handle the intricacies of the data, such as outliers and heteroscedasticity.

Through meticulous data preprocessing and visualization, the study uncovered key sales trends, seasonal patterns, and the importance of store-specific attributes. This comprehensive approach not only enhanced the reliability of the predictions but also highlighted critical areas for optimization in inventory management, promotional strategies, and operational planning.

Overall, this project underscores the value of data-driven methodologies in addressing real-world business challenges. By integrating advanced machine learning techniques, Walmart can make informed decisions that drive operational efficiency, improve customer satisfaction, and maintain a competitive edge in the retail industry. Future work could focus on refining model performance further, particularly in handling extreme values, and exploring additional features to enhance forecasting accuracy.