Puja Patidar

Walmart Retail Demand Forecasting

1.Introduction

Problem Statement

Retail organizations require accurate demand forecasts to effectively plan inventory levels, staffing, and supply chain operations. Inaccurate demand estimation can result in overstocking, stockouts, lost sales, and inefficient use of operational resources. Managing demand becomes particularly complex at the store–department level, where sales are influenced by promotions, seasonality, store characteristics, and external economic conditions.

Objective of the Project

The objective of this project is to develop a robust and reliable machine learning model capable of predicting weekly sales at the store–department level across Walmart stores using a regression-based machine learning approach. The model leverages historical sales data along with promotional activity, calendar effects, store-level attributes, and external economic indicators to estimate weekly demand. In addition to generating accurate sales forecasts, the analysis aims to identify key factors that drive variability in weekly sales.

Data Source

- **Kaggle:** Walmart Recruiting – Store Sales Forecasting https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data
- **Final dataset shape:** `(421,570 rows × 17 features)`
- **Target variable:** weekly sales (Dollar)
- Data includes sales, promotional markdowns, store metadata, calendar variables, and external indicators

Intended Users

- **Inventory Planning Teams**: Use the forecast to decide how much stock to keep and avoid overstocking or stockouts
- **Store Operations Managers**: Use weekly demand predictions to plan staffing and manage store-level operations efficiently.
- **Supply Chain and Logistics Teams**: Use demand estimates to plan shipments, warehouse capacity, and distribution schedules.
- **Merchandising and Category Managers**: Use model insights to assess the impact of promotions, seasonality, and pricing on sales.
- **Business and Data Analytics Teams**: Use forecast results and feature insights to monitor performance and refine forecasting strategies.

2. Data Cleaning and Wrangling:

The data cleaning process began by loading the raw datasets, including training, testing, store, and feature-level files, into pandas DataFrames. These datasets were merged to create a single analytical table by first joining the training data with store-level attributes using the Store identifier, followed by merging the resulting dataset with weekly feature data on both Store and Date. Column names were standardized by removing extra spaces, converting text to lowercase, and replacing spaces with underscores to ensure consistency across the dataset.

The Date column was converted from object format to datetime to enable time-based analysis. Missing values were systematically examined by calculating both the count and percentage of null values across columns. Missing values in the markdown variables (markdown1 through markdown5) were imputed with zero, based on the assumption that missing entries represent weeks without promotional markdown activity. Duplicate holiday indicator columns generated during the merge process were validated to be identical, after which one column was removed and the remaining column was renamed to isholiday.

Exploratory analysis of numeric feature distributions was conducted using histograms to assess skewness, spread, and the presence of outliers. This analysis revealed that weekly sales and markdown variables were highly right-skewed, while environmental and economic variables such as temperature, fuel price, CPI, and unemployment exhibited more stable distributions. Negative values in weekly_sales, likely arising from returns or accounting adjustments, were identified and addressed by creating a log-transformed feature (log_weekly_sales) using a log1p transformation, with values clipped to avoid invalid results.

To better understand the effect of promotional activity, a markdown indicator was created to distinguish weeks with promotional activity from those without. Sales distributions between these two groups were compared using summary statistics and visualizations. Finally, the cleaned and transformed dataset was exported to CSV format for downstream modeling and saved as a pickle file to support faster data loading in subsequent analysis. The final shape of the dataset after cleaning was (421,570 rows × 17 features)

3. Exploratory Data Analysis:

- Exploratory Data Analysis (EDA) was performed to understand the structure of the dataset, examine relationships between variables, and identify patterns relevant to weekly sales forecasting. The analysis started with basic inspection of the dataset to validate dimensions, data types, and summary statistics, ensuring the data was suitable for further analysis.

- Correlation analysis was conducted on numerical features to assess linear relationships and identify potential multicollinearity. The correlation heatmap revealed moderate association between store size and weekly sales, strong correlations among markdown variables, and relatively weak direct relationships between macroeconomic indicators and weekly sales. These findings highlight the presence of multicollinearity within promotional features and suggest that sales are influenced by multiple interacting factors rather than any single economic variable.

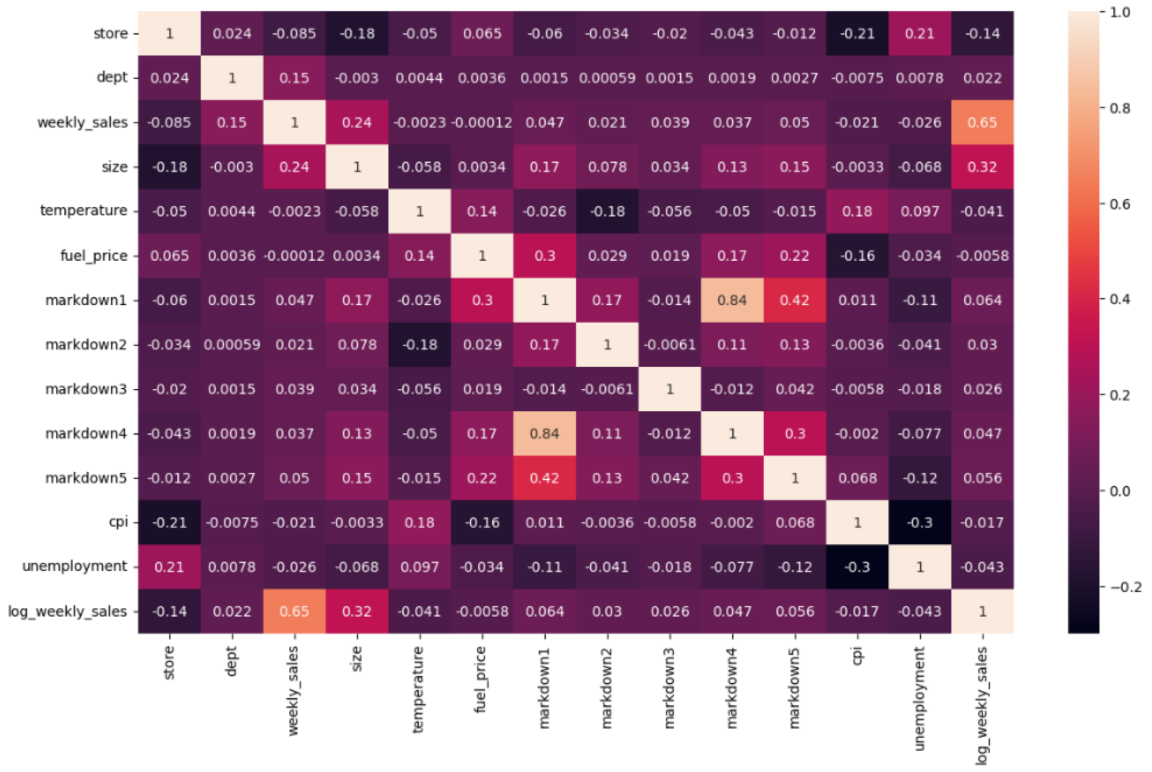


Figure 1 Correlation between numerical features

- To examine the spread and presence of extreme values, boxplots were used to analyze weekly sales with and without outliers. The results show substantial variability and the presence of extreme sales values, which can distort model performance if left unaddressed. A log transformation of weekly sales was visualized to assess distributional improvement and variance stabilization.

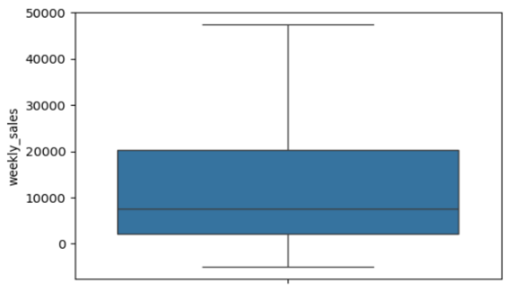


Figure 2 weekly sale boxplot w/o outliers

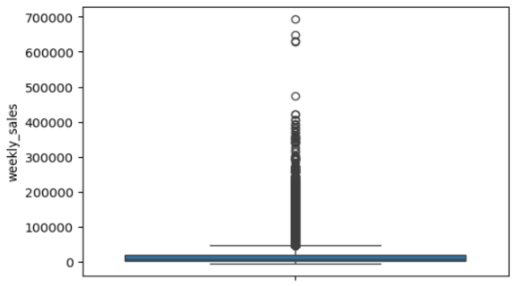


Figure 3 weekly sale boxplot with outliers

- Categorical and group-level analyses
  were performed to understand sales behavior across store types and holiday periods. Average weekly sales were compared across store types, revealing meaningful differences in sales performance. Holiday effects were further examined by comparing sales during holiday and non-holiday weeks across store types, indicating that the impact of holidays varies by store category.

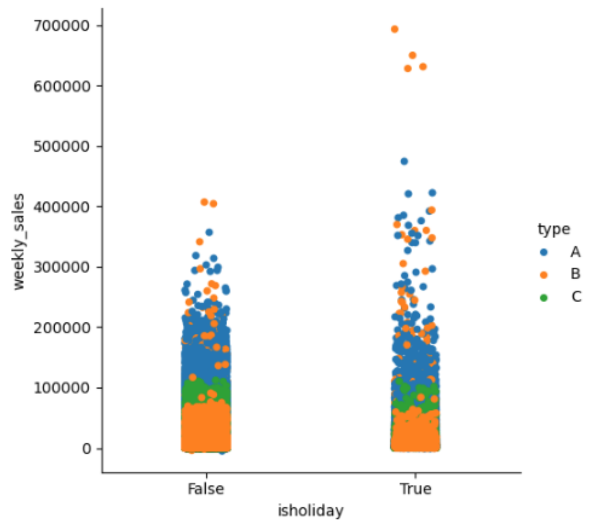


Figure 5 holiday vs weekly sales comparing across store types

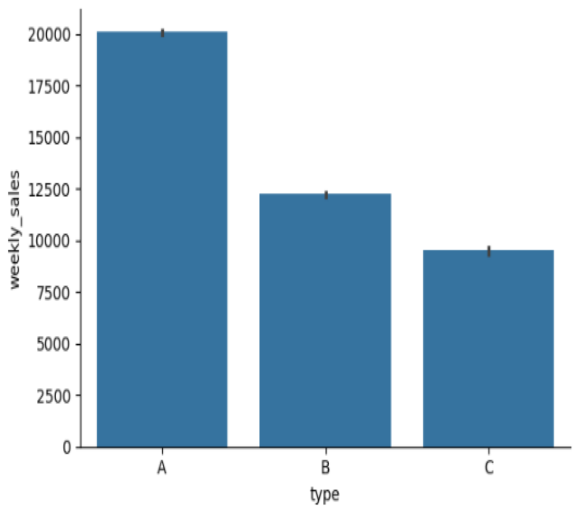


Figure 6 average weekly sales vs store types

- Store size was segmented into quartiles to assess its relationship with weekly sales. Sales distributions across size quartiles show that larger stores generally experience higher weekly sales, though variability remains within each group.

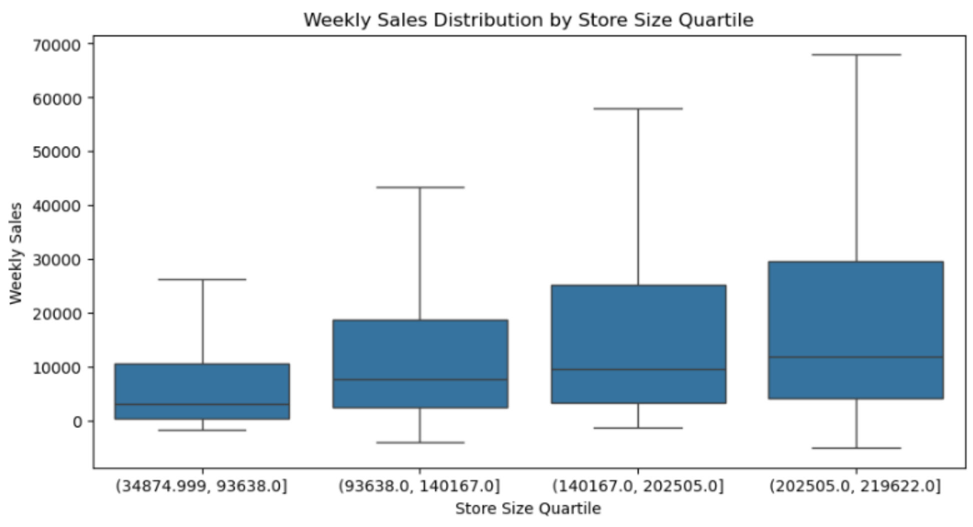
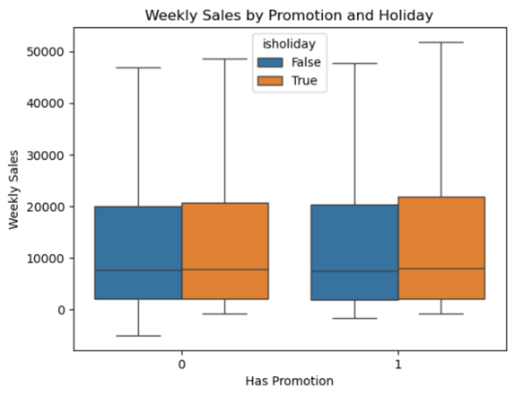
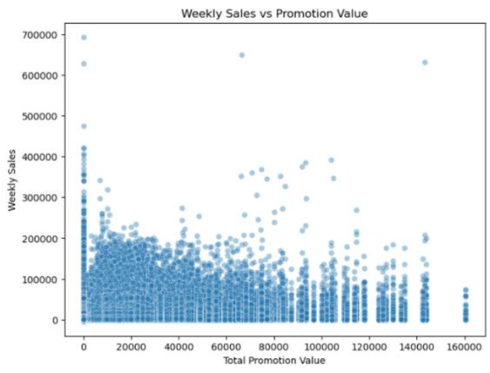


Figure 7 Weekly Sales Distribution by Store Size Quartile

- Promotional activity was analyzed by creating an indicator for weeks with active markdowns. Weekly sales distributions were compared between promotion and non-promotion weeks, further segmented by holiday status.





- Statistical testing using a two-sample t-test confirmed that sales differ significantly between promotional and non-promotional periods and There is a significant difference in average weekly sales between holidays and non-holidays periods. Additionally, a chi-square test was conducted to evaluate the association between holiday weeks and the presence of promotions.
- Overall, the EDA highlights the presence of outliers, promotional effects, store-level heterogeneity, holiday impacts, and strong temporal dependence, all of which informed subsequent feature engineering and modeling decisions.

4. Preprocessing and Modelling

Data Preprocessing

The preprocessing phase focused on preparing the cleaned Walmart sales dataset for time-series–aware modeling. The dataset was first loaded and inspected to ensure consistency in column names, data types, and the presence of all required features. The data was sorted chronologically to preserve the natural time sequence, which is critical for forecasting tasks.

Outlier treatment

Weekly sales contained extreme values, including high promotional spikes and negative values from returns or adjustments. Instead of removing records, percentile-based capping at the 1st and 99th percentiles was applied to reduce the impact of outliers while preserving all observations and maintaining the temporal structure of the data.

Feature Engineering, Encoding

- Calendar features (**year, month, day of week, weekend**) were derived from the date to capture seasonal patterns. Sales were normalized using **sales per square foot** to account for store size differences. Promotional impact was represented using a **promotion indicator** and **total promotion value**, while **fuel prices were bucketed** to capture non-linear effects. These features, along with store, holiday, and economic variables, were used for modelling, after feature engineering the shape of dataset was (421570,24)
- Following feature engineering, the dataset was prepared for modeling by defining the target variable and separating predictor features. Weekly sales was used as target variable, while all remaining engineered and contextual variables were treated as

predictors. Features were categorized into numerical and categorical groups to enable appropriate preprocessing and encoding.

- Categorical variables were encoded using **one-hot encoding and target encoding** to convert non-numeric categories into a machine-readable format without introducing ordinal assumptions. Numerical variables were passed through without scaling, as tree-based models do not require feature normalization. To ensure consistency, preprocessing was implemented using a ColumnTransformer, allowing numerical and categorical transformations to be applied within a single unified workflow.
- A baseline **Dummy Regressor** was trained to establish a minimum performance benchmark. A basic regression model using the same preprocessing pipeline was then evaluated to confirm that the engineered features provided meaningful improvement over the baseline.
- **TimeSeriesSplit cross-validation** was used to preserve temporal order and avoid look-ahead bias, with performance averaged across folds to assess generalization.

Modeling

- Time-based splitting using TimeSeriesSplit was used to preserve temporal order and avoid look-ahead bias during model evaluation**.**
- Multiple regression models, including **Linear Regression**, **Ridge Regression**, **Random Forest**, and **XGBoost**, were trained and evaluated using the same preprocessing pipeline and cross-validation strategy. Model performance was compared using **RMSE**, **MAE**, and **R²**, with **RMSE used as the primary metric** since it penalizes large forecasting errors and is well-suited for sales prediction tasks.

Model	Train RMSE	Test RMSE	Test MAE	Test R²
Dummy Regressor	N/A	20,802.57	14,698.98	0
Linear Regression	9,706.43	9,504.70	6,530.92	0.7865
Ridge Regression	9,706.43	9,504.69	6,530.92	0.7865
Random Forest	7,797.00	7,372.86	4,593.07	0.8715
XGBoost (Base)	6,865.70	7,100.56	4,096.63	0.8809
XGBoost (Tuned)	6,872.90	6,993.19	4,159.42	0.8844

Table 1Train and Test performance of all models

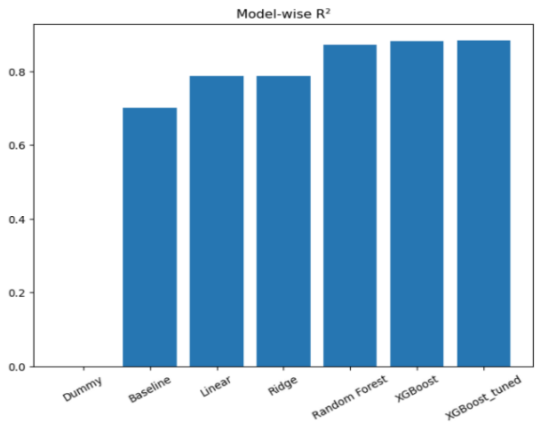


Figure 10 R2 of all models

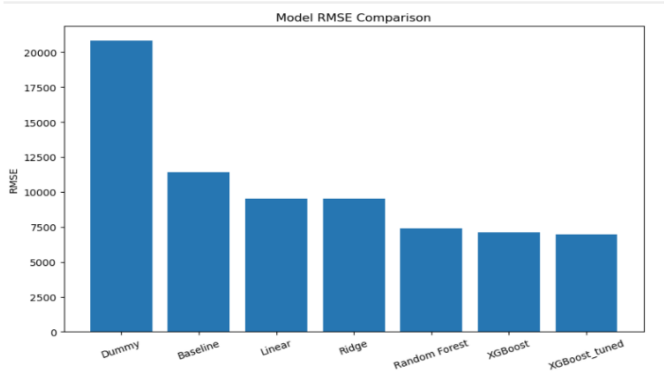


Figure 11 RMSE of all models

- Among all evaluated models, **XGBoost** consistently achieved the lowest RMSE, indicating superior performance in capturing non-linear relationships and complex interactions among features. Based on this result, XGBoost was selected for further optimization. Hyperparameter tuning was performed to improve model performance by adjusting parameters such as tree depth, learning rate, number of estimators, and subsampling ratios using cross-validation.
- Based on cross-validated results, the tuned XGBoost model was selected as the final model due to its consistently lower RMSE compared to other candidate models. Its robustness to feature interactions and non-linear effects makes it well-suited for predicting weekly store–department–level sales.

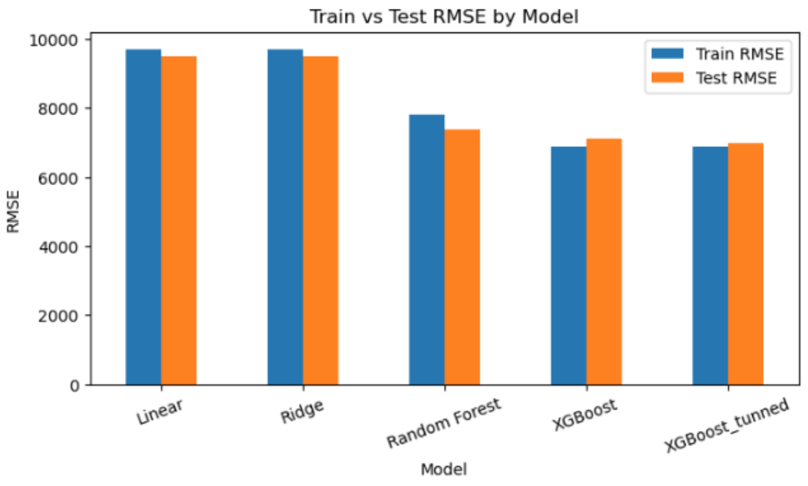
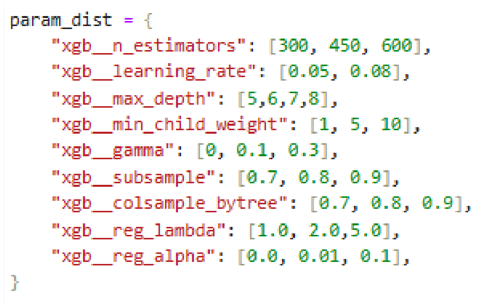


Figure 12 Train and Test RMSE of all models



```
param_dist = {
    "xgb__n_estimators": [300, 450, 600],
    "xgb__learning_rate": [0.05, 0.08],
    "xgb__max_depth": [5,6,7,8],
    "xgb__min_child_weight": [1, 5, 10],
    "xgb__gamma": [0, 0.1, 0.3],
    "xgb__subsample": [0.7, 0.8, 0.9],
    "xgb__colsample_bytree": [0.7, 0.8, 0.9],
    "xgb__reg_lambda": [1.0, 2.0,5.0],
    "xgb__reg_alpha": [0.0, 0.01, 0.1],
}
```

Figure 13 Parameter selected for XG-boost

Parameter selection for XG-boost Regressor, hyper parameter tuning by using Random Search as in figure 5

Best configuration after Random Search:

XGBoost parameters were selected using cross-validation, with the optimal configuration including n_estimators=300, learning_rate=0.05, max_depth=7, min_child_weight=1, gamma=0.1, subsample=0.7, colsample_bytree=0.7, reg_lambda=2.0, and reg_alpha=0.0, achieving the lowest RMSE.

Conclusion

This project built a time-aware machine learning framework to forecast weekly store–department–level sales. Among all evaluated models, the tuned **XGBoost** model achieved the lowest RMSE and best overall performance. Feature importance analysis showed that **store and department characteristics**, along with **store size, store type, month and calendar**

effects, are the primary drivers of sales, while promotions and economic variables have a smaller direct impact. The final model provides a reliable foundation for data-driven demand planning and operational decision-making.

Future Work

Future work can include adding **lag-based features**, improving **seasonality modelling**, and exploring **hierarchical approaches** at the store–department level to further enhance forecast accuracy.