Forecasting Weekly Sales

Github Repo:https://github.com/annihu601/1030project_AnniHu.git

Anni HU

Brown University
Data Science Institute

INTRODUCTION

The goal of this project is to predict weekly sales at the Store–Department level for Walmart using historical sales data combined with external economic and environmental factors such as Consumer Price Index (CPI) and temperature. Accurate demand forecasting at this granular level is critical for large retailers, as it directly affects inventory management, supply chain planning, pricing decisions, and labor allocation. Poor forecasts may lead to stockouts or overstocking, both of which incur significant financial costs.

The dataset used in this project was originally released by Walmart as part of a Kaggle forecasting competition. It consists of four files: train.csv, features.csv, test.csv, and stores.csv. The training data contain historical weekly sales for each Store and Department , while the features dataset includes external variables such as CPI, unemployment rate, fuel price, temperature, and indicators for holiday weeks. The stores dataset provides metadata about each store, including store type and size. According to the data description, the data were collected from Walmart's internal Enterprise Resource Planning (ERP) systems, reflecting real-world retail operations. Previous work on the Walmart weekly sales forecasting problem has been widely shared through Kaggle, where tree-based machine learning models such as Random Forest and Gradient Boosting are commonly used together with calendar and economic features. These models typically outperform simple baselines that predict the historical mean weekly sales.Given this existing work, the predictive performance achieved in this project is expected to be comparable to previously reported results. Large deviations from this range would likely indicate issues such as data leakage, inappropriate data splitting, or implementation errors.

EDA

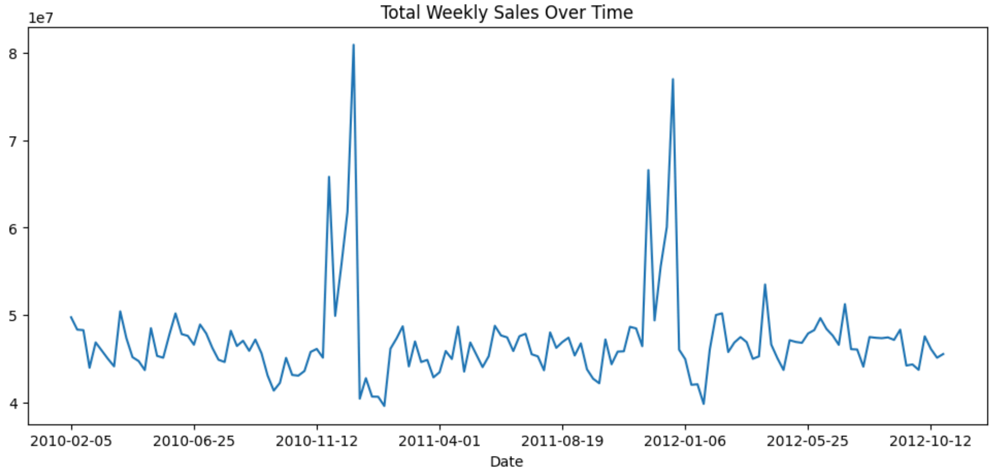


Figure1

Figure 1 shows total weekly sales aggregated over all stores and departments. Sales exhibit clear seasonal patterns, with sharp peaks during year-end holiday periods, while remaining relatively stable during non-holiday weeks. This highlights the importance of temporal and holiday-related features.

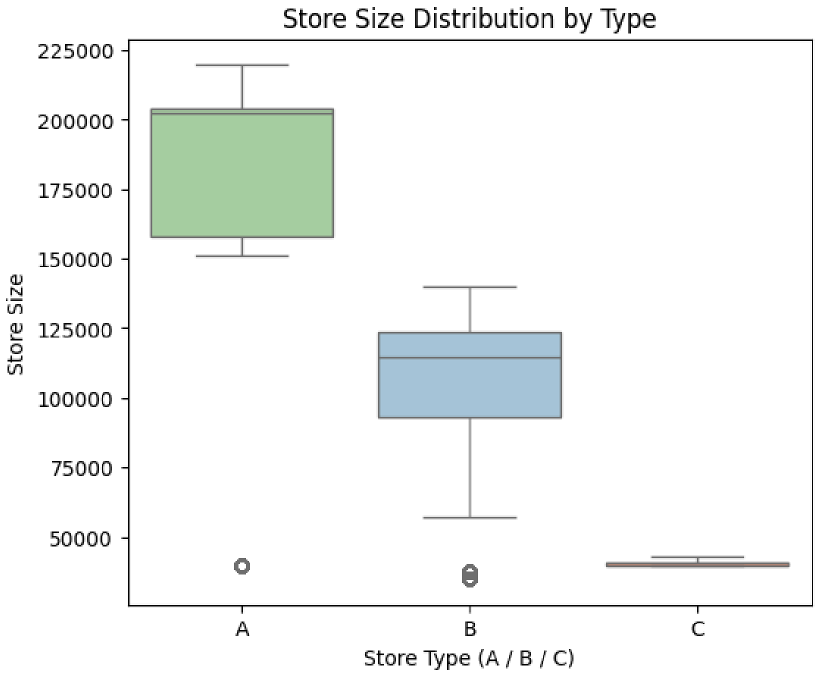
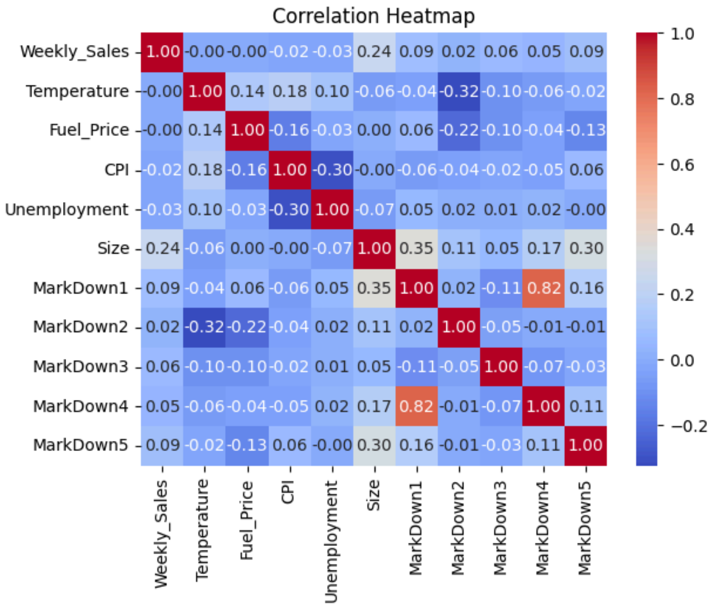


Figure2

Figure 2 displays the distribution of store sizes by store type. Store Type A is generally larger and more variable in size than Types B and C, while Type C stores are consistently small. This structural difference suggests that store type and size are important drivers of sales variation.



	Weekly_Sales	Temperature	Fuel_Price	CPI	Unemployment	Size	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5
Weekly_Sales	1.00	-0.00	-0.00	-0.02	-0.03	0.24	0.09	0.02	0.06	0.05	0.09
Temperature	-0.00	1.00	0.14	0.18	0.10	-0.06	-0.04	-0.32	-0.10	-0.06	-0.02
Fuel_Price	-0.00	0.14	1.00	-0.16	-0.03	0.00	0.06	-0.22	-0.10	-0.04	-0.13
CPI	-0.02	0.18	-0.16	1.00	-0.30	-0.00	-0.06	-0.04	-0.02	-0.05	0.06
Unemployment	-0.03	0.10	-0.03	-0.30	1.00	-0.07	0.05	0.02	0.01	0.02	-0.00
Size	0.24	-0.06	0.00	-0.00	-0.07	1.00	0.35	0.11	0.05	0.17	0.30
MarkDown1	0.09	-0.04	0.06	-0.06	0.05	0.35	1.00	0.02	-0.11	0.82	0.16
MarkDown2	0.02	-0.32	-0.22	-0.04	0.02	0.11	0.02	1.00	-0.05	-0.01	-0.01
MarkDown3	0.06	-0.10	-0.10	-0.02	0.01	0.05	-0.11	-0.05	1.00	-0.07	-0.03
MarkDown4	0.05	-0.06	-0.04	-0.05	0.02	0.17	0.82	-0.01	-0.07	1.00	0.11
MarkDown5	0.09	-0.02	-0.13	0.06	-0.00	0.30	0.16	-0.01	-0.03	0.11	1.00

Figure3

Figure 3 shows presents correlations among numerical variables. Weekly sales are moderately correlated with store size, whereas economic variables such as CPI, temperature, and unemployment show weak linear correlations with sales. The MarkDown variables are strongly correlated with each other, indicating overlapping promotional effects. These observations motivate the use of non-linear models to capture complex relationships and interactions.

Methods

To respect the temporal structure of the data, I used a time-series–based splitting strategy. Each Store–Department pair was treated as an independent time-series group. Within each group, observations were sorted by date, with the first 70% used for training and the remaining 30% for validation. This approach prevents information leakage from future observations. Although the dataset also includes an official Kaggle test set from 2013, it was not used for model evaluation since sales labels are unavailable.

Feature engineering focused on capturing temporal dynamics and external effects. I constructed lag-based features including last-week sales, last-year sales, rolling averages, as well as lags for temperature, CPI, and unemployment. Categorical variables were handled using one-hot encoding, while missing values in time-dependent variables such as CPI and unemployment were imputed using fallback strategies based on historical values. All features were then processed through a unified preprocessing pipeline to ensure consistent transformations across models.

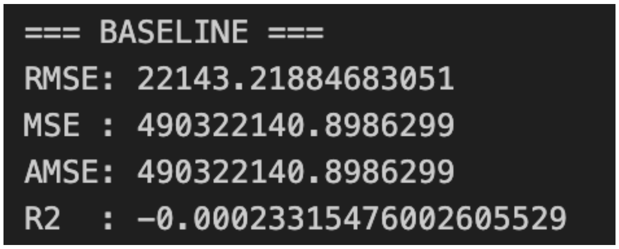
For the Machine Learning Model, I choose DummyRegressor as a baseline model, and also train Linear Regression, Elastic Net, Random Forest Regressor, XGBoost Regressor. Model performance was evaluated using RMSE, MSE, and R^2. RMSE and MSE were chosen because they penalize large prediction errors and remain in the original scale of weekly sales, which is important for assessing forecasting accuracy, while R^2 provides an interpretable measure of the proportion of variance explained by the model and allows for comparison across different models.

Hyperparameter tuning was conducted using GridSearchCV on a smaller data subset due to computational constraints. For Random Forest, the tuned parameters included n_estimators ∈ {400, 600}, max_depth ∈ {16, 18}, min_samples_leaf ∈ {5, 10}, min_samples_split ∈ {10, 30}, and max_features ∈ {0.3, "sqrt"}. For XGBoost, we tuned max_depth ∈ {5, 8}, learning_rate ∈ {0.02, 0.05}, subsample ∈ {0.6, 0.8}, colsample_bytree ∈ {0.8, 1.0}, reg_lambda ∈ {1, 2}, and reg_alpha ∈ {0, 0.1}. The selected hyperparameters were then used to train final models on the full dataset.

To assess uncertainty in model evaluation, I examined variability arising from both data splitting and model stochasticity. For time-series splitting, performance was compared across multiple temporal splits to ensure stability over time. For non-deterministic models such as Random Forest and XGBoost, I trained models with different random seeds and measured the variation in evaluation metrics. These analyses confirmed that observed performance differences between models were consistent and not driven by random variation. Overall, each step of the pipeline was designed to balance predictive performance, computational feasibility, and robustness while avoiding data leakage.

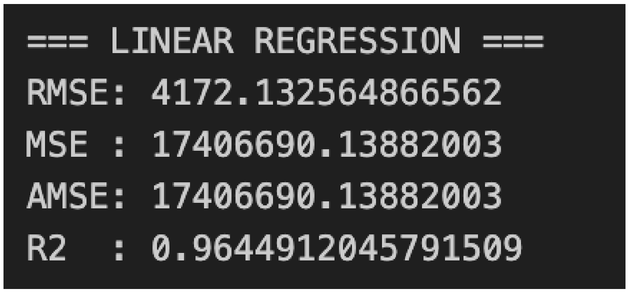
RESULTS

Baseline Model

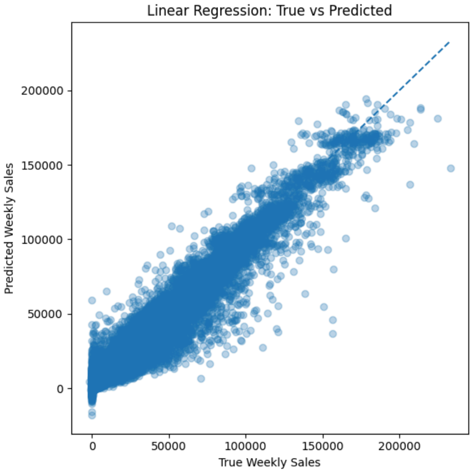


```
=== BASELINE ===
RMSE: 22143.21884683051
MSE : 490322140.8986299
AMSE: 490322140.8986299
R2  : -0.00023315476002605529
```

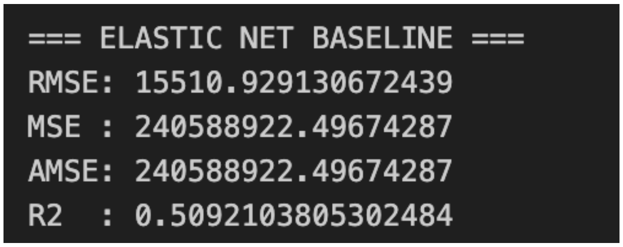
Linear Regression



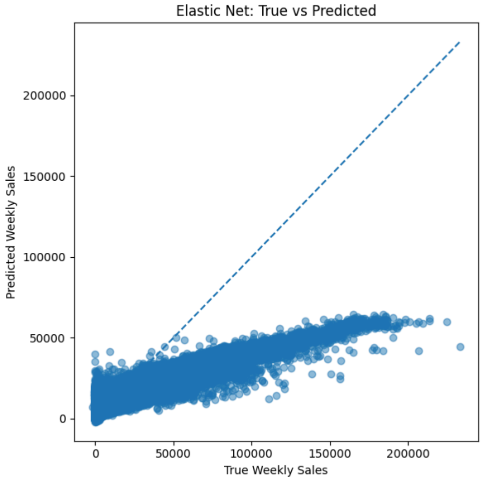
```
=== LINEAR REGRESSION ===
RMSE: 4172.132564866562
MSE : 17406690.13882003
AMSE: 17406690.13882003
R2  : 0.9644912045791509
```



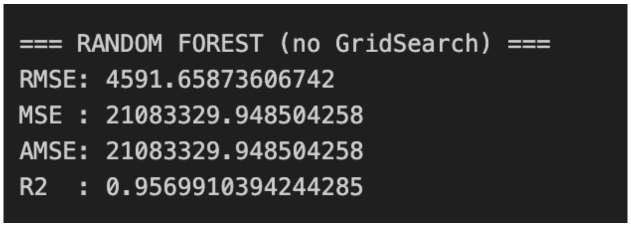
Elastic Net



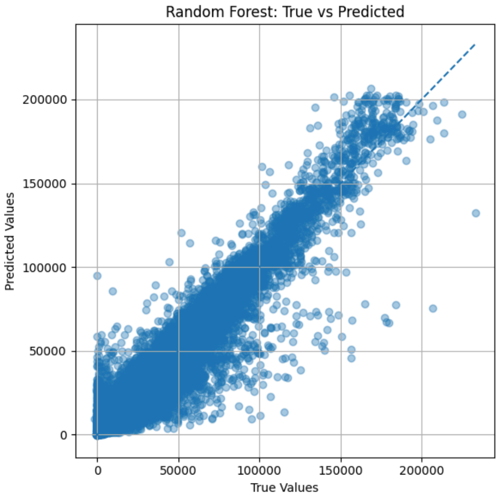
```
=== ELASTIC NET BASELINE ===
RMSE: 15510.929130672439
MSE : 240588922.49674287
AMSE: 240588922.49674287
R2  : 0.5092103805302484
```



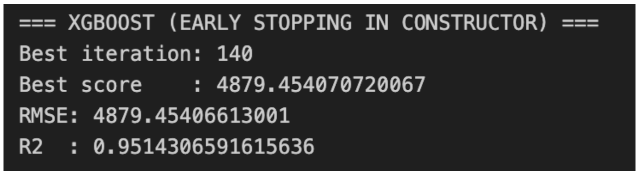
Random Forest



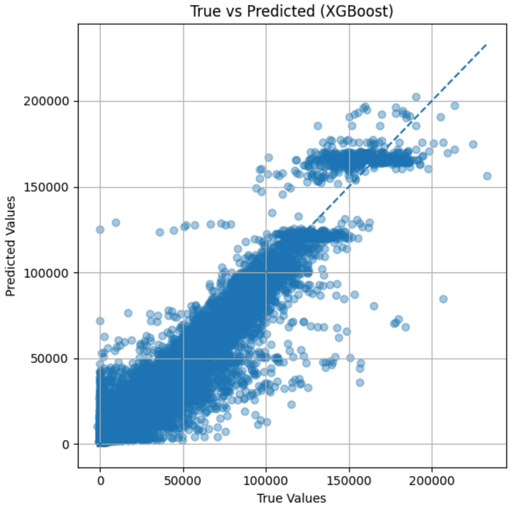
```
=== RANDOM FOREST (no GridSearch) ===
RMSE: 4591.65873606742
MSE : 21083329.948504258
AMSE: 21083329.948504258
R2  : 0.9569910394244285
```



XGBoost



```
=== XGBOOST (EARLY STOPPING IN CONSTRUCTOR) ===
Best iteration: 140
Best score    : 4879.454070720067
RMSE: 4879.45406613001
R2  : 0.9514306591615636
```



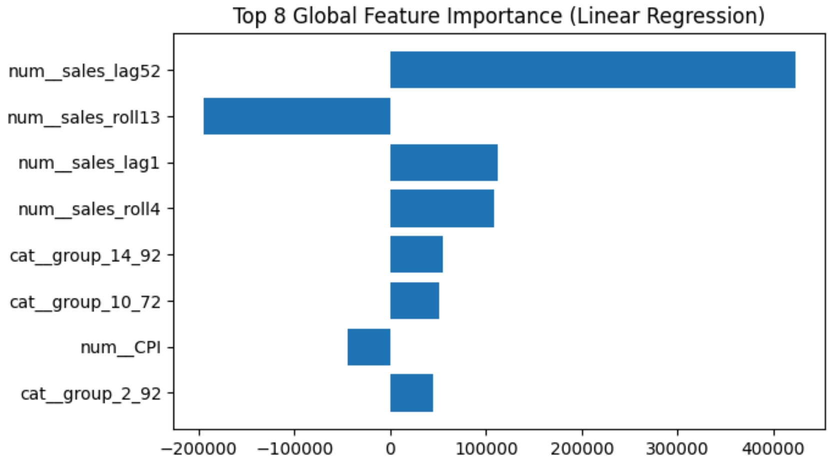
Interpretation

Tables above summarize the predictive performance of all models evaluated on the validation set. The DummyRegressor baseline, which predicts the historical mean, achieves an RMSE of 22,143 and an R² close to zero, providing a reference for model comparison. All machine learning models substantially outperform this baseline.

The Linear Regression model achieves the best overall performance, with an RMSE of 4,172 and an R² of 0.964, outperforming more complex non-linear models. As shown in the true versus predicted scatter plot, the linear model produces predictions that closely align with the diagonal across both low and high sales ranges, indicating minimal systematic bias. The spread of residuals remains relatively constant, and the model does not exhibit noticeable underestimation or overestimation in high-sales periods. In contrast, more complex models tend to introduce slight dispersion at extreme values. This visual evidence suggests that the relationship between the engineered features and weekly sales is well approximated by a linear mapping, explaining why the linear regression model generalizes effectively.

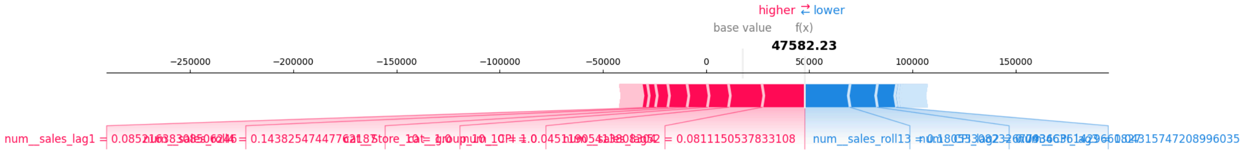
Random Forest and XGBoost also perform well (RMSE = 4,592 and 4,879; R² = 0.957 and 0.951, respectively), but do not surpass the linear model. The scatter plot shows that these two models capture the overall sales trend well across most of the distribution, particularly in the mid-range of weekly sales, reflecting its ability to model non-linear relationships and interactions. However, predictions become more dispersed in high-sales regions, with increased variance around the diagonal, suggesting greater uncertainty when extrapolating to extreme values.

Elastic net performs worst across all Machine Learning Models.As shown in the true versus predicted scatter plot, Elastic Net produces relatively concentrated predictions in the low-to-moderate sales range but exhibits clear systematic underestimation at higher sales levels. Repeated runs of Random Forest with different random seeds yield highly consistent evaluation metrics, indicating that performance differences between models are stable and not driven by stochastic variation. Compared to the baseline, the best-performing model improves RMSE by a large margin, corresponding to multiple standard deviations above the baseline performance.



Global Feature importance(Linear Regression coefficient)

Across models, feature importance analyses consistently identify lag-based sales features as the dominant predictors. Using three different global importance methods—absolute linear coefficients (Linear Regression / Elastic Net), impurity-based importance (Random Forest), and gain-based importance (XGBoost)—the same set of features repeatedly emerges as most important, including last-week sales, rolling averages, and last-year sales. This strong agreement across methods highlights the highly autoregressive nature of weekly sales at the Store–Department level. In contrast, temperature and macroeconomic variables such as CPI and unemployment consistently exhibit low importance, suggesting limited direct predictive power once historical sales information is included.



SHAP value of index[50](linear Regression)

Local interpretability using SHAP values further confirms these findings. For individual predictions, recent sales history (lag1, lag2), medium-term trends (rolling windows), and annual seasonality (lag52) contribute most to the predicted values, while categorical variables provide smaller adjustments.

An interesting and somewhat unexpected result is that the linear model outperforms non-linear models. This outcome is interpretable: extensive lag and rolling features effectively linearize the forecasting problem, leaving limited additional signal for more complex models to exploit. Overall, these results demonstrate that careful feature engineering plays a more critical role than model complexity for this task.

Summary

This project addresses the practical problem of forecasting weekly sales for each Store–Department combination at Walmart, a task that directly impacts inventory planning, staffing, and supply chain decisions. Using historical sales data, the project builds a forecasting pipeline that predicts future demand based on recent sales patterns and seasonal behavior. By incorporating lagged sales features and rolling averages, the model captures short-term momentum as well as yearly seasonality, allowing it to produce accurate forecasts without relying heavily on external assumptions. Multiple models were evaluated to balance accuracy, stability, and interpretability, and the final results show that a well-specified linear model can reliably predict weekly sales at a granular level. The analysis further demonstrates that historical sales behavior is the primary driver of demand, while economic variables such as CPI and temperature add limited additional value once sales history is accounted for. Overall, this project provides a data-driven and interpretable approach to weekly sales forecasting that can be directly applied to operational decision-making in retail settings.

OUTLOOK

With more time and computational resources, several directions could further improve both model performance and interpretability. Additional models such as LightGBM and other ensemble methods could be explored to better capture complex non-linear patterns while remaining computationally efficient. Further feature engineering, particularly incorporating additional lag features and longer rolling windows, may help the model better capture seasonality and long-term temporal trends. A key limitation of the current approach is that hyperparameter tuning and cross-validation were performed on a data subset due to computational constraints, which may not fully capture year-long seasonal patterns. Running cross-validation on the full dataset would likely lead to more robust model selection. In terms of interpretability, advanced SHAP analyses across different time periods could provide deeper insights into how feature importance evolves over time. Finally, collecting additional data such as promotional intensity, pricing information, or regional economic indicators could further enhance predictive performance.

REFERENCES

Dataset: https://www.kaggle.com/datasets/aslanahmedov/walmart-sales-forecast

Previous work(from Kaggle code):
https://www.kaggle.com/datasets/aslanahmedov/walmart-sales-forecast/code