# EVALUATING TIME-SERIES AND MACHINE LEARNING MODELS FOR RETAIL SALES FORECASTING

**Alexander Coles** **acoles6@gatech.edu**
**Tanmayee Kolli** **tkolli3@gatech.edu**
**Simran Mallik** **smallik9@gatech.edu**
**Chris Park** **cpark453@gatech.edu**
**Vaishnavi Vuyyuru** **vvuyyuru3@gatech.edu**

GEORGIA INSTITUTE OF TECHNOLOGY

## ABSTRACT

Sales forecasting is a critical component of inventory management, budgeting, resource allocation, and operational planning for retailers, yet it remains challenging due to high volatility, seasonality, and data scarcity. This study benchmarks classical time-series models (ARIMA, SARIMA) against additive models (Prophet), ensemble machine learning methods (Random Forest, XGBoost), and deep learning architectures (LSTM) on weekly technology sales data. We address the challenge of applying and comparing these models on a smaller-scale, more volatile retail dataset and highlight which approaches are most effective and why. Our results demonstrate that while classical models capture general trends, they fail to predict the magnitude of demand spikes. Feature-engineered ensemble methods—specifically a seasonal & holiday-aware XGBoost model—achieved the strongest performance. We conclude that for volatile, limited-historical retail datasets, explicitly engineered features combined with tree-based boosting outperform both classical statistical methods and raw sequence-based deep learning.

## 1 INTRODUCTION

Accurate sales forecasting is the backbone of retail operations. Retailers rely on weekly revenue predictions to optimize inventory levels, manage supply chains, allocate resources, and budget for promotions. The cost of forecasting errors is high: under-forecasting leads to stockouts and lost revenue, while over-forecasting results in excess inventory and markdown losses. Retail sales data is further complicated by irregular spikes driven by promotions or holidays and by underlying seasonal patterns.

In this study, we operate as an analytics consulting team tasked with identifying the optimal forecasting strategy for a superstore's Technology product category. Our primary objective is to evaluate modern machine learning (ML), deep learning (DL), and classical time-series forecasting models for predicting weekly revenue for Technology products.

Specifically, we investigate:

- **Comparative performance:** How do ARIMA/SARIMA compare to Random Forest, XGBoost, and LSTMs in predicting weekly Technology revenue?
- **Feature importance:** To what extent does feature engineering (including lagged features, seasonal encodings, promotional variables, and rolling statistics) impact model accuracy?

We hypothesize that gradient-boosted tree models (XGBoost) with explicit seasonality will outperform both classical statistical models and deep learning approaches due to their ability to ingest non-linear interaction terms (e.g., Quantity $\times$ Discount) and incorporate multiple seasonal and promotional signals.

## 2 Related Work

Sales forecasting has traditionally relied on statistical methods. ARIMA (AutoRegressive Integrated Moving Average) and its seasonal variant SARIMA are industry standards for linear, stationary time-series data. While effective for stable trends, these models often lack the flexibility to incorporate exogenous regressors (such as holidays or promotions) easily.

More recently, Facebook Prophet introduced a decomposable additive model that handles seasonality and holidays intuitively, becoming a popular baseline in industry.

In the machine learning domain, ensemble methods like Random Forest and XGBoost have gained traction. Unlike time-series models, these treat forecasting as a supervised regression problem, requiring the engineering of lag features. They are robust to noise and capable of capturing non-linear relationships.

Deep Learning, particularly Long Short-Term Memory (LSTM) networks, offers the promise of learning temporal dependencies directly from raw sequences without manual feature engineering. However, LSTMs typically require large datasets to avoid overfitting, posing a challenge for weekly aggregated retail data which is often limited in sample size.

## 3 Data: Source, Challenges, and Preparation

### 3.1 Dataset Overview

We utilized the *Superstore Sales* dataset sourced from Kaggle, which contains 9,994 transaction records spanning January 2014 through December 2017. Each record includes features such as *Order Date*, *Sales*, *Quantity*, *Discount*, *Profit*, *Category*, and *Region*. These attributes support both time-series forecasting and feature-based machine learning approaches.

### 3.2 Preprocessing and Challenges

**Category selection.** Exploratory analysis revealed data imbalance in the number of orders among the three product categories: *Office Supplies*, *Furniture*, and *Technology*. Although the Technology category had the fewest orders, it exhibited the highest profit and sales. Additionally, the revenue patterns for Technology did not follow the clear seasonal trends observed in the aggregate dataset. These findings motivated us to focus specifically on forecasting Technology sales, as it represented the most challenging and analytically interesting subset.

**Aggregation.** To establish a consistent time series suitable for classical and machine learning forecasting models, the transactional data was aggregated to a weekly level. This resulted in a final dataset consisting of $N = 208$ weekly observations.

**Data scarcity.** With only 208 weekly data points, deep learning models face a high risk of overfitting, as they typically require large datasets. This constraint places greater emphasis on model selection, regularization strategies, and the value of feature engineering.

## 4 Methodology

### 4.1 Exploratory Data Analysis

Our exploratory data analysis (EDA) examined structural patterns, temporal behavior, distributions, and variability in the Superstore dataset to guide model selection, with particular emphasis on the Technology category.

**Category-level patterns.** The Technology category contained the fewest total orders but produced the highest revenue and profit, along with the greatest variability. Its weak seasonal structure and volatile behavior made it the most challenging and informative category to model, motivating our focus on forecasting Technology revenue.

**Temporal behavior.** Monthly visualizations (used for clarity, even though all modeling uses weekly data) revealed substantial volatility, including isolated revenue spikes and only mild, inconsistent Q4 seasonality. These patterns suggested that strict seasonal models may struggle, whereas flexible models could still benefit from the inclusion of seasonal indicator features.

**Distribution and outliers.** Sales and profit were heavily right-skewed. Approximately 12% of weekly sales observations were flagged as outliers under an IQR rule; however, these spikes corresponded to genuine high-revenue periods and were retained. Their presence implies that classical smoothing-based models may underfit, while machine learning models incorporating lagged, rolling, and volatility-aware features may better capture abrupt changes.

**Segment and geographic insights.** Customer segment, region, and subcategory analyses revealed structural differences in purchasing behavior, but no strong recurring or time-dependent patterns that would enhance forecasting performance for the Technology category. This reinforced that Technology revenue is driven more by volatility than by stable trend or seasonal cycles.

**Implications for modeling.** Technology revenue displays high variance and irregular spikes, making flexible modeling approaches essential. Seasonality exists but is weak, limiting the effectiveness of SARIMA-type models while still allowing machine learning methods to benefit from seasonal indicators. Because the observed outliers represent genuine demand surges, models must account for non-linear, abrupt changes rather than smoothing them out. These insights informed our modeling pipeline: beginning with classical baselines, incorporating feature-engineered tree-based methods, and ultimately evaluating deep learning approaches.

### 4.2 Modeling Approach

Our modeling approach followed a progressive complexity curve, beginning with simple statistical baselines, advancing through classical time-series models, and ultimately evaluating machine learning and deep learning methods for forecasting weekly Technology sales.

#### 4.2.1 Classical Time-Series Models

**Baseline.** We used a naive baseline model that extended the last observed weekly value forward.

**ARIMA.** ARIMA served as our initial statistical benchmark to evaluate how much of the weekly sales pattern could be explained using only trend and short-term autoregressive structure. After confirming stationarity with the Augmented Dickey–Fuller (ADF) test, a grid search identified ARIMA(0,0,0) as the best-fitting specification, indicating no meaningful AR or MA structure without seasonality. This model consistently underpredicted major revenue spikes and exhibited high error, confirming that ARIMA alone cannot capture the cyclical, volatile retail patterns revealed in our EDA.

**SARIMA.** To incorporate repeating patterns, we extended ARIMA to SARIMA and used the autocorrelation function (ACF) to evaluate candidate seasonal periods. Although yearly seasonality (lag 52) appeared in the ACF, quarterly seasonality (13 weeks) yielded better performance and aligned with common retail promotional cycles. Our strongest configuration, SARIMA$(1,0,2)(1,1,1)_{13}$, captured the timing of seasonal fluctuations but underestimated spike magnitudes, highlighting the need for models that incorporate promotions or additional external drivers.

**Prophet.** Prophet, a flexible additive time-series model designed to decompose trend and seasonality, was used as an industry-standard baseline. We experimented with variants incorporating quantity, discount, interaction terms (e.g., quantity $\times$ discount), and one-hot categorical features such as ship mode and region. However, these one-hot regressors introduced noise and overfitting, degrading performance. Models with weekly and yearly seasonality did not improve performance, and our simplest Prophet variant, using only Sales, Quantity, and Discount, remained the strongest Prophet model.

#### 4.2.2 Machine Learning Models (Ensemble Methods)

**Random Forest.** Random Forest is an ensemble method that aggregates predictions across many decision trees, reducing the overfitting risk inherent to single-tree models. We developed multiple iterations, beginning with lag-based predictors and later incorporating seasonal encodings using trigonometric functions. To evaluate temporal generalization, we applied walk-forward cross-validation. Random Forest significantly outperformed all previous models and closely followed the test-set pattern but still failed to capture the largest revenue spikes.

**XGBoost.** XGBoost is an extreme gradient boosting algorithm that builds trees sequentially, with each tree correcting errors from prior iterations. It handles noise, non-linearity, and lag-based structure effectively, and is widely used in industry settings such as Amazon, Uber, and Walmart. We created a series of XGBoost models incorporating feature selection, seasonality, cross-validation, and randomly sampled hyperparameter tuning.

Our modeling progression included: (1) a baseline model using only lags, (2) a seasonal, feature-rich model with lags and rolling statistics, (3) models with seasonal one-hot features, (4) models using randomized hyperparameter sampling, (5) models combining lag and rolling features with mutual-information-based feature selection, and (6) holiday-aware models with multiplicative interactions. Our strongest model was a seasonal, holiday-aware model with multiplicative interactions, MI-selected predictors, lag features, and nested cross-validation. Nested CV ensured unbiased hyperparameter tuning: the inner loop optimized parameters, while the outer loop produced a leakage-free performance estimate. Although the outer CV error was pessimistic (due to small sample size and early-fold volatility), the final model remained statistically rigorous and demonstrated strong real-world forecasting reliability (see Figure 1 in Appendix D).

#### 4.2.3 Deep Learning (LSTM)

**Architecture.** We implemented a Long Short-Term Memory (LSTM) network to evaluate the sequence-learning potential of deep learning models. The architecture consisted of a single LSTM layer with 24 units applied over an 8-week lookback window of raw sales history, followed by a small dense branch encoding cyclical time features and a final dense output layer.

**Addressing data scarcity.** Given the limited dataset size ($N = 208$ weeks), overfitting posed a significant challenge. To expand the effective training signal, we applied data augmentation by "jittering" the training values through noise injection and scaling, increasing the training set size by a factor of four. Additional regularization strategies included L2 penalties, dropout, Gaussian noise applied to inputs, Huber loss, Adam optimization, early stopping, and learning rate reduction on plateau.

**Methodology check.** Early iterations of the model exhibited data leakage due to improperly fitting scalers on the full dataset. This issue was corrected by fitting all preprocessing transformations exclusively on the training split, ensuring that the validation and test sets remained untouched and that the LSTM's performance estimates reflected true generalization ability.

## 5 Experimental Setup and Results

### 5.1 Evaluation Strategy

We explored multiple forecasting algorithms (ARIMA, SARIMA, Prophet, Random Forest, XGBoost, and LSTM), developing several model variants for each. For every algorithm, we selected the best-performing variant, and from these candidates we identified the overall strongest forecasting model.

All models used the same initial temporal split: data prior to January 1, 2017 served as the training set (70%), while data on or after January 1, 2017 constituted the test set (30%). This ensured that all models were evaluated on strictly unseen future data.

For several models, we implemented baselines for comparison, including a SARIMA model with a 52-week seasonal lag, a simple XGBoost model using only lagged sales features, and a basic Prophet

configuration using Sales, Quantity, and Discount as inputs. To ensure robust performance estimates, we used cross-validation for both XGBoost and Random Forest models, tuning hyperparameters while maintaining proper temporal ordering to avoid leakage.

### 5.2 METRICS

**Metrics considered.** We evaluated model performance using several complementary metrics, each capturing a different aspect of forecast accuracy and business relevance:

- **RMSE** highlights large deviations in dollar terms; lower values indicate better capture of extreme spikes such as end-of-year peaks.
- **MAE** measures average prediction error in dollars, making it directly interpretable for business impact.
- **MAPE** provides scale-independent percentage error, enabling comparison across weeks of varying revenue.
- **MdAPE** offers a robust measure of central tendency that is less sensitive to outliers, reflecting typical weekly forecast accuracy.
- **Coverage** evaluates how well prediction intervals capture true observations, indicating reliability under uncertainty.

**Metrics used.** For our final comparisons, we focused primarily on RMSE and MdAPE. RMSE measures the magnitude of errors in dollar terms and heavily penalizes large deviations, ensuring that models accurately capture critical revenue spikes. MdAPE complements RMSE by quantifying typical percentage error while being robust to outliers and irregular spikes. Together, these metrics allowed us to assess both overall accuracy and week-to-week reliability of model predictions.

### 5.3 RESULTS

Our seasonal, holiday-aware XGBoost model with multiplicative interactions, MI-selected predictors, lag features, and nested cross-validation outperformed all other forecasting models. It achieved a test RMSE of 1,079.58 and a Median Absolute Percentage Error (MdAPE) of 12.71%. These results represent a substantial improvement over classical statistical models and deep learning approaches, particularly in capturing high-variance revenue spikes. For a full detailed model comparison table, see Table 1 in Appendix C.

## 6 INTERPRETATION & ERROR ANALYSIS

### 6.1 THE SUCCESS OF XGBOOST

XGBoost was the strongest model by a wide margin, reducing the error rate (MdAPE) from approximately 48% (ARIMA, LSTM) to 12.71%. Error analysis indicates that XGBoost succeeded because it leveraged interaction features. By explicitly modeling terms such as *Profit* $\times$ *Quantity* and *Quantity* $\times$ *Discount*, the model captured the underlying causes of revenue spikes (promotional effects) rather than relying solely on temporal recurrence.

### 6.2 FAILURE ANALYSIS OF CLASSICAL MODELS

ARIMA and SARIMA effectively smoothed the time series but failed to capture the amplitude of major spikes. While SARIMA detected the timing of seasonal peaks (e.g., Q4 increases), it consistently underestimated their magnitude. In a retail context, such underestimation leads to significant inventory shortages during high-demand periods. Prophet captured seasonality but lacked the capacity to represent multiplicative interactions, such as discount-driven demand amplification, which proved essential for achieving the highest forecasting accuracy.

### 6.3 Deep Learning vs. Data Scarcity

The LSTM model performed comparably to the basic ARIMA model (MdAPE $\approx 48\%$), substantially underperforming tree-based models. Visual inspection shows that although the LSTM captured the general seasonal rhythm (e.g., dips in early 2017, rises in Q4), it failed to model the extreme volatility of Technology sales. This result highlights that deep learning architectures require large datasets to learn the complex relationships that feature engineering provides explicitly to ensemble models. On small tabular datasets ($N = 208$), the complexity of LSTMs yields diminishing returns compared to well-tuned gradient boosting.

### 6.4 Methodological Learnings

During experimentation, we identified a critical instance of data leakage in our initial LSTM approach, where iterating on the test set artificially inflated performance. Implementing a rigorous Nested Cross-Validation (Inner CV for tuning, Outer CV for evaluation) corrected this, raising the validation error but providing a realistic estimate of production performance.

While developing various XGBoost models, we initially applied cross-validation on the testing set rather than the training set, which produced overly optimistic results. This experience highlighted how easily data leakage can occur when building and evaluating multiple models simultaneously. We corrected this by training all XGBoost models exclusively on the training data, performing cross-validation on that training set for hyperparameter tuning and metric validation, then retraining on the full training set before reporting final results.

For our XGBoost models, we also initially attempted manual hyperparameter tuning for model depth (md) and learning rate (lr), assuming that tuning these first would reduce complexity. However, manual tuning proved error-prone and introduced bias. Switching to a randomized hyperparameter search reduced human-induced contamination and provided more reliable results. In our LSTM workflow, Nested CV further ensured unbiased estimates: the outer CV produced a true generalization score, while the inner CV tuned hyperparameters without leaking information.

Overall, these methodological corrections reinforced a key principle of time-series forecasting: all evaluation must strictly respect temporal ordering to avoid look-ahead bias. Ensuring leakage-free validation was essential for producing performance estimates that would generalize reliably in a real retail forecasting environment.

## 7 Summary and Takeaways

In this study, we benchmarked six forecasting architectures on volatile weekly Technology sales data. Our findings highlight several key insights:

- **XGBoost + seasonality is the winning formula.** Combining gradient boosting with explicit seasonal features and interaction terms produced the strongest accuracy (MdAPE = 12.71%).
- **Rich features $>$ simple lags.** Models relying solely on past sales (ARIMA, simple lag models) failed to capture spike magnitudes. Rolling statistics, holiday indicators, and interaction terms (e.g., Quantity $\times$ Discount) were necessary for avoiding stockouts and over-ordering.
- **Deep learning suffers from data scarcity.** For small tabular datasets, feature-engineered ML approaches outperformed LSTM, which struggled to generalize without a large training corpus.

**Recommendation.** For limited-history, high-volatility retail environments, we recommend deploying a seasonal and holiday-aware XGBoost model validated with nested cross-validation. Its combination of strong accuracy, interpretability, and low deployment complexity makes it well suited for operational forecasting workflows.

## A HYPERPARAMETER TUNING FOR XGBOOST

We performed randomized hyperparameter search within a nested cross-validation framework to ensure unbiased performance estimation.

**Outer CV:** 3-fold time-series split.
**Inner CV:** 3-fold hyperparameter tuning.

**Parameters tuned:**

- `max_depth`: 3–10
- `learning_rate`: 0.01–0.3
- `n_estimators`: 100–1000
- `subsample`: 0.7–1.0

## B FEATURE IMPORTANCE (XGBOOST)

The top five features contributing to predictive performance were:

1. `Profit_x_Quantity` (interaction term)
2. `rolling_std_3` (volatility measure)
3. `lag_1` (immediate past sales)
4. `week_of_year` (seasonality)
5. `is_holiday_season` (event flag)

## C MODEL COMPARISON TABLE

| Model | RMSE | MdAPE | Coverage | Key Limitation |
|---|---|---|---|---|
| Baseline | 5,476.59 | 126.68% | 5.7% | Failed to predict volatility. |
| ARIMA | 4,541.87 | 48.61% | 5.7% | Captured trend only; missed spikes. |
| SARIMA | 4,814.00 | 49.76% | 11.3% | Captured seasonal timing but not magnitude. |
| Prophet | 3,495.00 | 47.41% | 88.68% | Handled seasonality, did not handle multiplicative effects. |
| Random Forest | 2,789.72 | 21.19% | N/A | Learned short-term patterns, struggled with long-term. |
| LSTM | 4,960.00 | 48.10% | N/A | Captured sequential trends but failed to predict peak magnitudes. |
| **XGBoost** | **1,079.58** | **12.71%** | **N/A** | **Best performance; Combined seasonality, lags, promotions well but risk of overfitting.** |

Table 1: Comparative performance across all forecasting models.

## D XGBOOST FORECAST VISUALIZATION

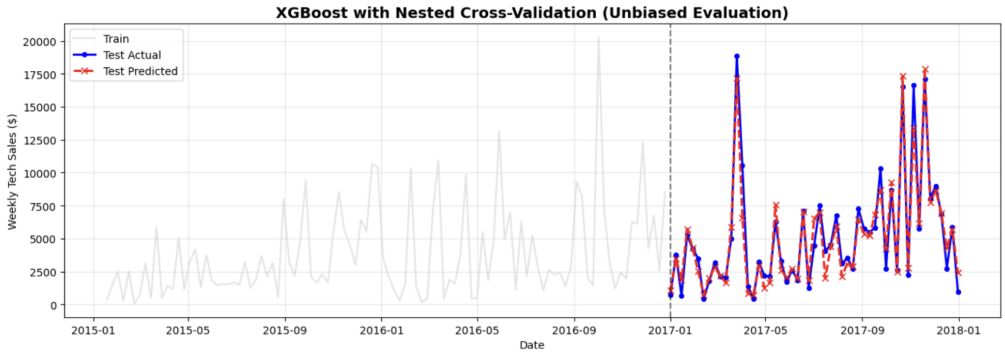


Figure 1: Predicted vs. actual weekly sales using the best XGBoost model with nested cross-validation.