رواد مصر الرقمية

وزارة الاتصـــــالات
وتكنولوجيا المعلومات

SALES FORECASTING PROPOSAL

Initiative: Digital Egypt Pioneers

Round 3

Prepared by

Mohamed Mahrous Mohamed Mahmoud

Sahar Mohamed Abdelmoneim Mahmoud Oransa

Ahlam Reda Almetwally Abdelkarim

Hams Mohamed Hassan Ali

Mohamed Ali Hassan Rehan

Afnan Mohamed Ibrahim Elsenosy

Prepared for:

Eng. Sherif Mohamed

Table of Contents

Abstract

This project develops a machine-learning time-series forecasting system using the Kaggle Store Sales dataset, which closely mirrors key retail factors relevant to the Egyptian market, including promotions, holidays, fuel prices, and regional purchasing patterns. Multiple forecasting models were evaluated—such as XGBoost, CatBoost, and SGDRegressor—with LightGBM achieving the best RMSLE score and selected as the final model. The pipeline includes full data cleaning, multi-source merging, and advanced feature engineering. The system can later be retrained on Egyptian data and deployed through a Streamlit interface to support real-time sales prediction and data-driven inventory planning.

1.1 Introduction

This project aims to build a predictive analytics system capable of forecasting product sales and supporting better inventory decisions using historical time-series data. Accurate demand forecasting is essential for retail businesses to manage stock levels, plan promotions, and make informed operational decisions. Many retailers—especially in developing markets such as Egypt—struggle with fluctuating demand, limited analytical tools, and reliance on manual decision-making.

To address these challenges, this project develops a data-driven machine-learning forecasting pipeline using the Kaggle Store Sales Time Series dataset. Although the dataset originates from Ecuador, it captures key retail drivers that closely resemble the Egyptian market, including promotions, holiday effects, fuel price fluctuations, regional purchasing behavior, and daily transaction patterns. This makes it a suitable foundation for building a forecasting system that can later be adapted and retrained using real Egyptian data when available.

The project follows a complete end-to-end workflow: data cleaning, merging multiple time-based sources, extensive feature engineering, and model experimentation. Several machine-learning models were tested—such as XGBoost, CatBoost, and

SGDRegressor—with **LightGBM achieving the best RMSLE score** and therefore selected as the final model. The resulting forecasting system provides reliable sales predictions that can enhance inventory planning, reduce uncertainty, and improve business decision-making.

Additionally, the model is designed to be deployable through a simple Streamlit interface, enabling users to visualize predictions and upload new data for real-time forecasting. This expands the project's practicality for business teams and supports future deployment within the Egyptian retail environment.

1.2 PROJECT OVERVIEW

The Sales Forecasting Project focuses on building a machine-learning time-series model capable of predicting daily product sales based on historical data and multiple external factors. The project integrates several data sources—including sales records, store metadata, oil prices, holiday schedules, and transactions—into a unified forecasting pipeline.

The final system is designed to support data-driven planning, enhance decision-making for retailers, and serve as a scalable forecasting framework that can later be adapted to the Egyptian retail market once local data becomes available.

1.3 PROBLEM STATEMENT

Retail businesses, particularly in developing markets such as Egypt, often struggle with unpredictable demand, inefficient inventory management, and a lack of advanced analytical systems. These challenges lead to overstocking, stockouts, weak promotion planning, and financial losses.

To address these issues, a reliable and automated forecasting model is needed to predict sales trends accurately, incorporate environmental and economic factors, and support business teams with actionable insights rather than intuition-based decisions.

1.4 Business Benefits:

Implementing a machine-learning forecasting system provides several key business advantages:

- **Reduced stockouts and overstocking**, leading to more efficient inventory management.
- **Improved promotion and marketing planning** based on predicted demand cycles.
- **Better financial forecasting**, supporting budgeting and procurement decisions.
- **Enhanced operational efficiency**, replacing manual estimation with data-driven predictions.
- **Scalability** allows retailers to apply the same model to multiple stores, regions, or product categories.
- **Future adaptability**, enabling the system to be retrained with Egyptian data to reflect local consumer behavior, holidays, and economic indicators

1.5 PROJECT OBJECTIVES

The main objectives of the project are:

1. **Develop a robust time-series forecasting model** for predicting daily product sales using historical multi-source data.
2. **Build a complete data pipeline**, including cleaning, merging, feature engineering, and encoding.
3. **Experiment with multiple machine-learning models** and select the best performer using RMSLE evaluation.
4. **Deploy the final model** through a simple Streamlit interface for interactive forecasting.
5. **Provide a scalable forecasting framework** that can later be adapted to real Egyptian retail data.
6. **Support data-driven decision-making** in inventory planning, promotions, and operational management.

1.6 Dataset Description:

The dataset used in this project consists of multiple files that provide complementary information about daily retail activity. Together, these files capture sales behavior, store characteristics, promotions, holidays, economic indicators, and customer transactions. Although the data originates from Ecuador, the overall structure closely resembles the retail environment in Egypt, making it a suitable foundation for building and later adapting a forecasting system to the Egyptian market.

1.6.1 Raw Data Files Overview:

- **train.csv** contains daily historical sales for every store and product family, along with the number of items on promotion.
- **test.csv includes the same features as the training data but without the** target sales values, representing future dates for prediction.
- **stores.csv** provides store-level metadata such as the city, state, store type, and cluster classification.
- **oil.csv** records daily oil prices (WTI), used as an external economic indicator that may influence consumer purchasing behavior.
- **holidays_events.csv** lists national, regional, and local holidays and events, including transferred and additional holiday days.
- **transactions.csv** contains the daily number of customer transactions for each store, representing customer traffic levels

1.6.2 Dataset Integration Process

Since the dataset was originally divided into several separate tables, it was necessary to combine all sources into a single unified time-series table before modelling.
The merging process was performed using two primary keys:

- **date** – aligns time-based information such as sales, oil prices, holidays, and transactions.

- **store_nbr** – connects store metadata and transaction records to the correct store in the sales data.

These two fields were essential because every dataset in the project was organized around **time** and **store identity**, making them the most reliable keys for creating a consistent and complete view of daily retail behavior.

1.6.3 Additional Feature Processing (Before Time-Series Features)

Before merging all data sources, several feature transformations were applied to enrich the time-series structure. Holiday information was split into national, regional, and local indicators, and multiple date-based features—such as year, month, day, day of week, and week of year—were extracted to capture seasonal and temporal patterns. A payday feature was also added, marking the 15th of each month as a salary payment date based on the dataset description to reflect potential increases in purchasing activity. After completing all merges, the final dataset became a comprehensive consolidated table containing over 3 million rows and 20 features—excluding the ID column. The full list of features included in the merged dataset is presented in Table No (1).

Table No (1)

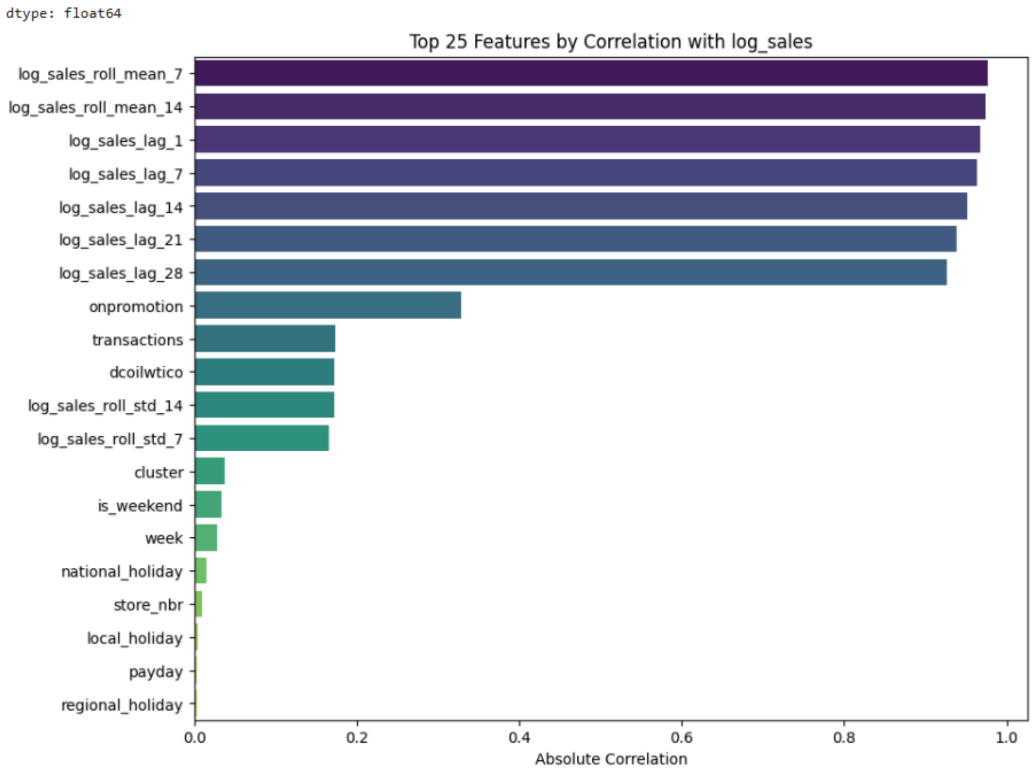
Train & Test Tables (Merged Structure Before Time-Series Features)

Column Name	Description
Id	Unique identifier for each row
date	Daily timestamp of the record.
store_nbr	The store number where the sales occurred(Unique identifier)
family	The product family/category (e.g., dairy, bakery, beverages).
Sales(train only)	Target variable: total sales for the family at that store on that date
on promotion	Number of items on promotion for that specific product family and store.
PayDay	An indicator showing whether the date is a salary payment day.
dcoilwtico	The daily WTI oil price is used as an external economic variable.🎬
city	City where the store is located.
state	State/province in Ecuador.
Type	Store type classification (A, B, C…)
cluster	**cluster** – Grouping of stores with similar commercial behavior.

Column Name	Description
national_holiday	Indicator for national-level holidays on that date.
regional_holiday	Indicator for regional holidays affecting specific areas.
local_holiday	Indicator for city-level or local holidays.
transactions (train only)	Number of customer transactions at the store on that date.
year	Extracted feature representing the year of the record.
month	– Extracted month.
day	Extracted day of month.
day_of_week	Day of week (0–6).
Week of the year	Week number of the year

1.6.4 Introduce Rolling & Lags in Feature Engineering:

Further enhancement of the time-series structure was achieved through additional feature engineering techniques, including **lag features** and **rolling window statistics**. These transformations produced new columns that reflect historical sales patterns and capture both short- and long-term trends. Although this process noticeably increased the number of features in the dataset, only the most relevant time-series features were retained for the final model. The selection was guided by their correlation with the target variable and their importance, as shown in *Figure 1*



```
dtype: float64
```

Figure 1

Top 25 Features by Correlation with Sales (Target variable)

1.7 Data Cleaning & Preprocessing

Before conducting exploratory analysis or building forecasting models, the raw dataset underwent several preprocessing steps to ensure data quality, consistency, and readiness for feature engineering. The cleaning procedures focused on handling missing values, detecting outliers, validating data types, identifying inconsistencies, and preparing the dataset for time-series modelling.

1.7.1 Handling Missing Values:

- The oil dataset contained several missing values in the dcoilwtico column, which were imputed using forward fill, an appropriate method for continuous daily time-series data.
- Missing values in holiday-related fields (e.g., national, regional, local holidays) were filled with zero, representing “no holiday.”
- All other datasets contained no critical missing values requiring intervention.

1.7.2 Checking Data Types and Conversions

- Date fields were converted to a datetime format.
- Categorical variables such as family, city, state, and type were correctly cast as category types.
- Numerical fields (sales, transactions, dcoilwtico, onpromotion) were validated to ensure they were recognized as numeric types.

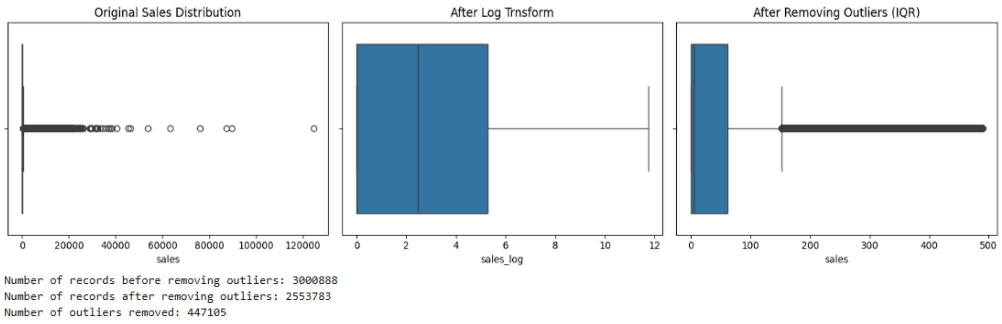
1.7.3 Duplicate Records:

No dataset contained problematic or repeated rows requiring removal.

1.7.4 Outlier Detection

A full outlier analysis was performed across all numerical columns in all datasets. Outliers were detected in three features, in addition to the *sales* target variable**.**

- **Sales**: The sales column contained 447,105 high-value observations, which appear as statistical outliers. However, these values are expected due to natural variation in store size, demand patterns, and product family volume across different regions.
- Because the sales variable does not follow a normal distribution, it requires transformation before modelling to properly study its relationship with other predictors. Multiple transformations were tested—including log scaling and quarter-based segmentation—but the log transformation produced the most suitable normalized distribution, as shown in **Figure 2**.



```
Number of records before removing outliers: 3000888
Number of records after removing outliers: 2553783
Number of outliers removed: 447105
```

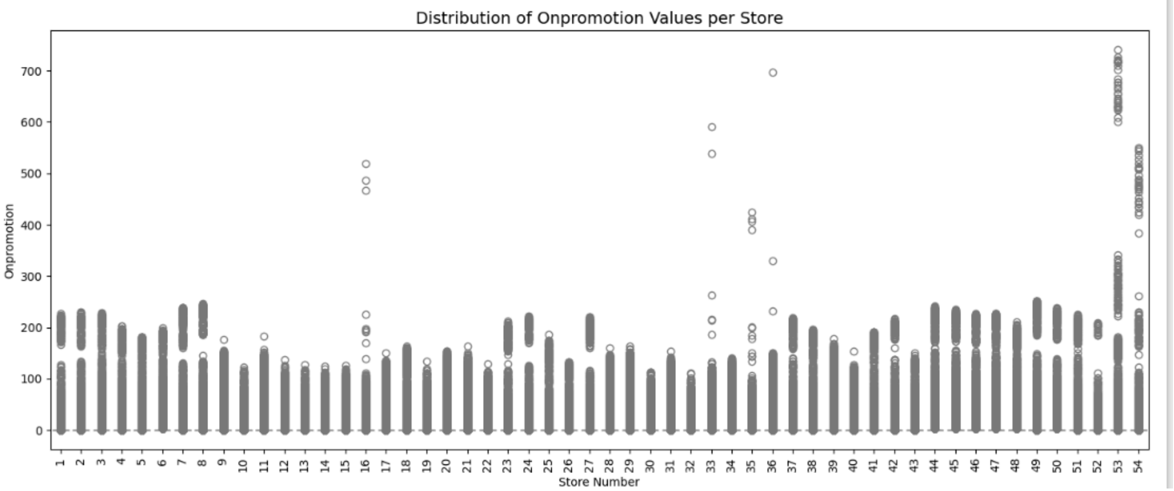
Figure 2

Distribution of Sales Before and After Applying Log and Quarter Transformations

- ### Outliers in *promotions and* transactions

In the training dataset, the onpromotion column contained 611,329 statistical outliers, while the test dataset included 3,325 outliers, as illustrated in Figure 3. After closer examination, these extreme values were interpreted as natural variations resulting from differences in store size, product family volume, and promotion intensity across locations; therefore, they were not removed.

Similarly, in the transactions dataset, the transactions column showed 4,583 outliers, as shown in Figure 4. However, applying a log transformation significantly distorted the actual values and altered the original distribution. For this reason, both onpromotion and transactions were retained in their original scale without applying a log transformation. It is also important to note that the test dataset does not contain a transactions column, so outlier evaluation for this feature was performed only on the training data**.**



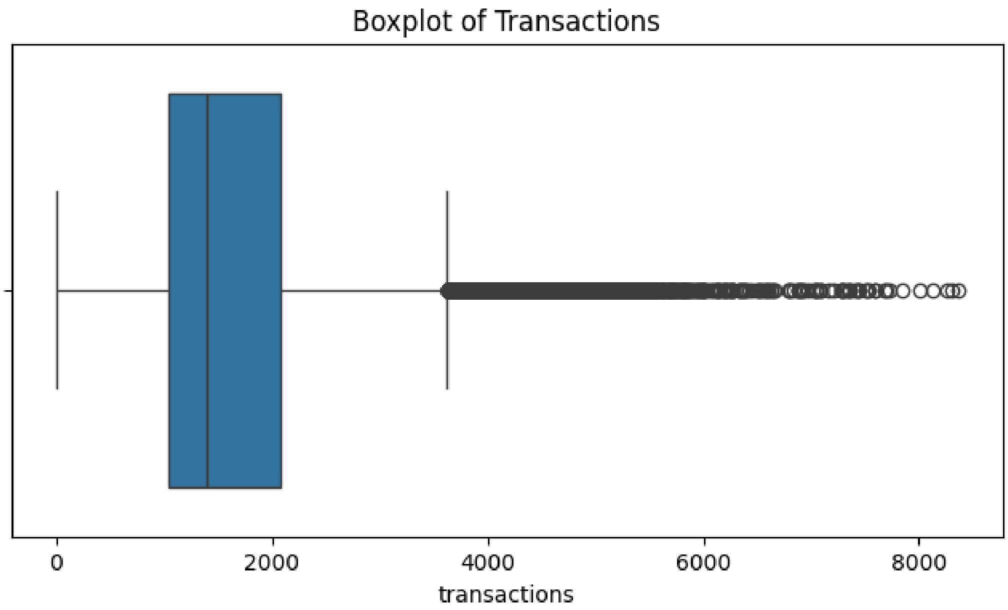


Figure 4: Transactions Boxplot Showing Natural High Outliers

1.7.5 Final Pre-processed Dataset:

After completing all preprocessing steps:

- No rows were removed from any dataset.
- All outliers were retained because they represent real business behaviour rather than errors.
- A log-transformed version of the sales variable (**sales_log**) was created to correct skewness and support more stable modeling.
- All date, categorical, and numerical fields were cleaned and converted to their correct formats.

This resulted in a consistent and reliable dataset, ready for EDA and feature engineering.

1.8 Exploratory Data Analysis (EDA)

The Exploratory Data Analysis stage aims to understand the behavior of the retail time series, identify patterns, explore seasonality, assess feature relationships, and quantify the effect of promotions, holidays, and external factors. These insights guide feature engineering and help shape an accurate forecasting model.

1.8.1 Sales Trend and Seasonal Patterns

A detailed analysis of the sales trends revealed several important characteristics:

- **Long-Term Trend**

The overall sales time series shows large fluctuations rather than a clear long-term increasing or decreasing trend. These variations are driven by differences in store size, product families, promotions, and external economic and seasonal events.

- **Weekly Seasonality**

The weekly trend (Figure 5 – Weekly Sales Trend) shows a clear cyclical pattern where sales increase toward weekends and decline at the start of the week.

This pattern was later captured through features such as day_of_week and is_weekend.

- **Monthly Seasonality**

Monthly sales (Figure 5 – Monthly Sales Trend) display noticeable fluctuations influenced by:

- salary payment cycles (payday feature),
- holiday seasons,
- special local or national events

- **Annual Patterns**

  Monthly sales compared across years (Figure 5 – Monthly Sales Trends per Year) show consistent seasonality from 2013 to 2017, confirming stable yearly demand cycles.

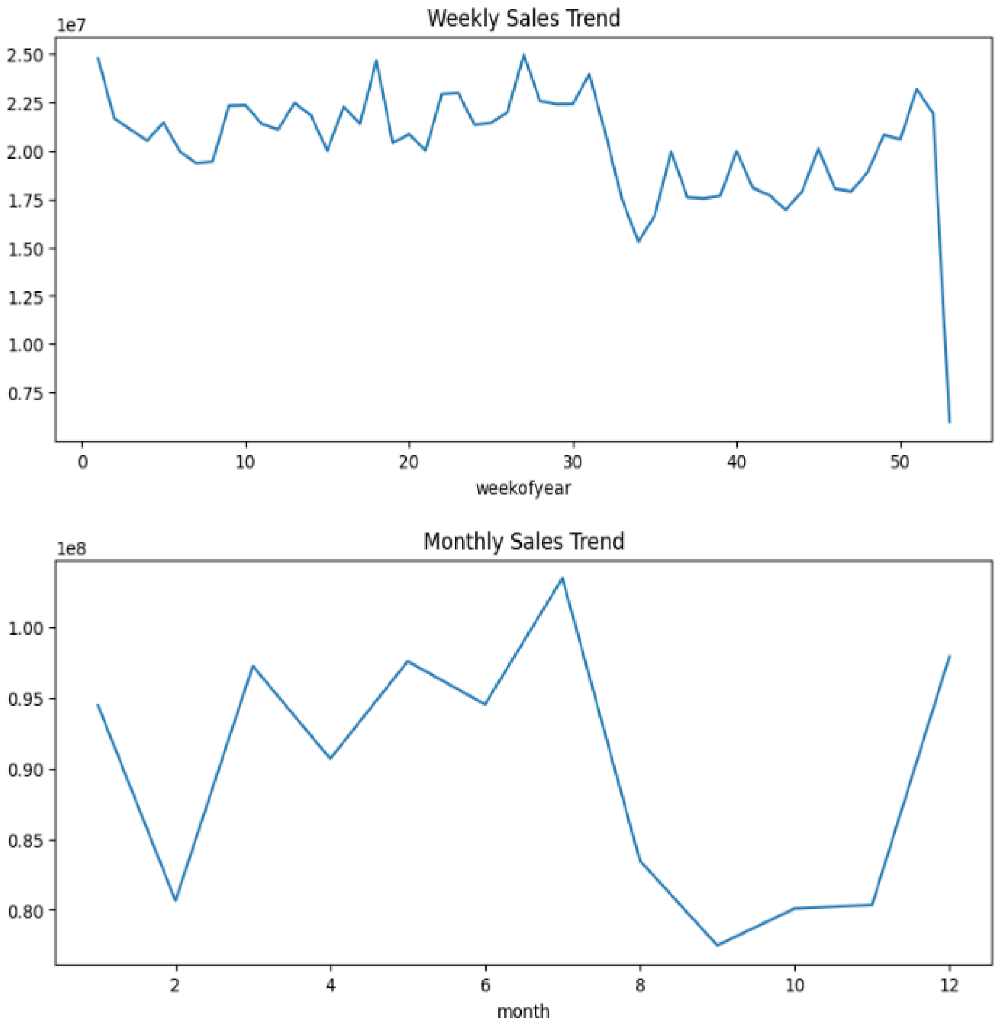
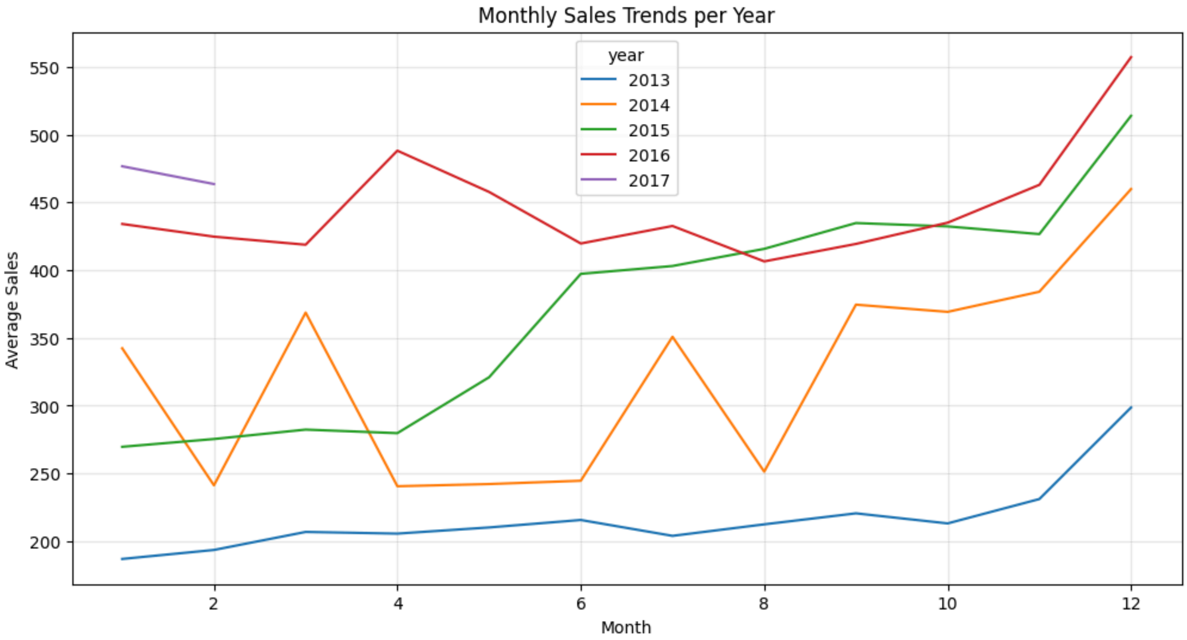


Figure 5 Sales Trend and Seasonal Patterns



Continuation of Figure 5: Sales Trend and Seasonal Patterns

1.8.2 Impact of Promotions and Transactions

- **Impact of Promotions**

Periods with higher promotion counts show a moderately positive relationship with sales (correlation ≈ 0.40), as illustrated in the correlation heatmap (Figure6).

- **Impact of Transactions**

Higher customer traffic (transactions) also shows a positive moderate correlation with sales (correlation ≈ 0.23), confirming that busy days drive higher sales volume.

The heatmap (Figure 6) summarizes the correlations among all numerical features after preprocessing and confirms the importance of both onpromotion and transactions for forecasting**.**

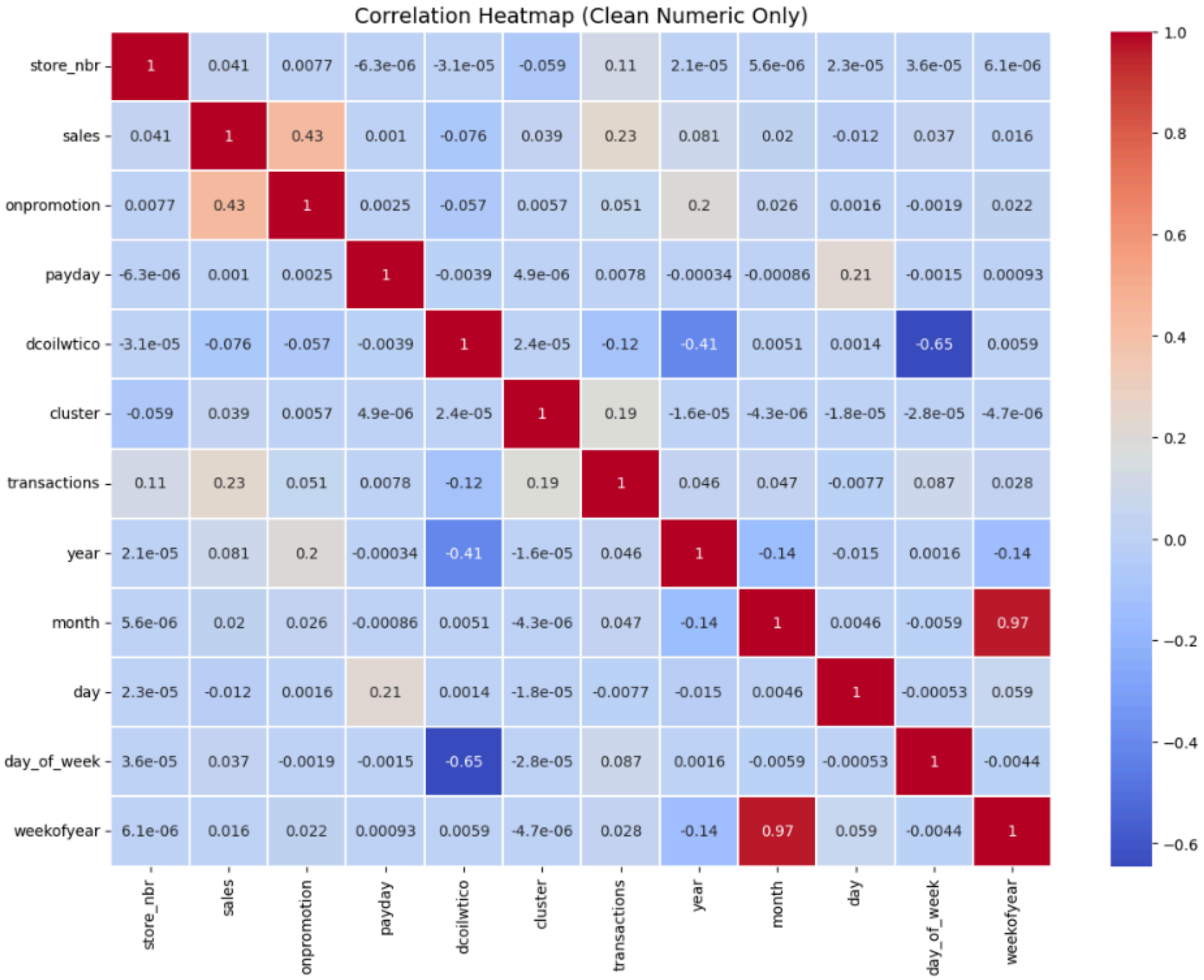


Figure 6. Correlation Heatmap of Numerical Features After Preprocessing

1.8.4 Conclusion from Exploratory Data Analysis

Based on the patterns observed in Figure 5, we noticed clear changes across months, years, and weekly cycles, indicating that the original time-based features alone could not fully capture the temporal dynamics of the sales data. Therefore, we decided to create lag features and rolling window statistics to represent historical sales behaviour and better model short-term and long-term trends.

Additionally, the correlations shown in Figure 6 provided clear guidance on selecting the most relevant variables for the forecasting model. The heatmap revealed that promotions and transactions had the strongest positive relationships with sales, confirming their importance as key drivers of demand. In contrast, several other variables showed weak or negligible correlations and were therefore excluded from the final feature set. These insights from Figure 6 directly informed our feature selection decisions and helped define the structure of the final forecasting model.

1.9 Feature Engineering

Feature Engineering is a critical step in preparing the dataset for machine-learning forecasting. It aims to extract meaningful patterns, capture temporal behaviors, and generate new variables that help the model learn seasonal, cyclical, and promotional effects more effectively. Based on the insights obtained from the EDA and the correlation results, several transformations and new features were introduced to enhance the predictive capability of the model.

1.9.1Calendar-Based Features

To capture the temporal structure of the sales data, a series of date-related variables was extracted from the **date** column, including **year**, **month**, **day, day_of_week**, **weekofyear**, and **is_weekend** (derived feature)

These features allow the model to learn monthly cycles, weekly repeating patterns, and yearly seasonal effects, which were clearly observed in Figure 5.

1.9.2 Holiday & PayDay Indicators

The holiday indicators (national, regional, local) and the payday feature were already created during Milestone 1 – Data Collection, Exploration & Preprocessing, and included directly in the modeling dataset.

1.9.3 Lag Features

Based on EDA results, it became clear that the model must capture historical patterns directly. Therefore, several lag features were created to represent past sales values:

- **lag_1**
- **lag_7**
- **lag_14**

- **lag_21**
- **lag_28**

These lags help the model learn short-term weekly repetitions and longer seasonal cycles.

1.9.4 Rolling Window Statistics

Rolling windows were applied to capture smoothed historical trends and volatility:

- **rolling_mean_7**, **rolling_mean_14**
- **rolling_std_7**, **rolling_std_14**

These features capture momentum, moving averages, and variability in sales over time, which significantly improved correlation with the target (as shown in your correlation results).

1.9.5 Log Transformation of Sales

As shown in Figure 2 (before/after transformation), the sales distribution did not follow a normal pattern. Therefore, **sales_log = log1p(sales)** was generated to:

- reduce skewness
- stabilize variance
- improve the model's ability to learn proportional changes

This transformed target was used during model training.

1.10 Feature Selection Before Modeling

Before starting the modeling phase, a feature-selection step was carried out to keep only the most relevant predictors. A correlation matrix was generated twice—before and after creating all lag and rolling features (Figure 7). Based on these results, we removed variables with weak or redundant relationships to the target, including id, all holiday indicators, weekofyear, day, payday, and several lag/rolling features.

Many lag and rolling features also showed very high correlation with one another, so only the most informative ones were retained to reduce dimensionality, avoid multicollinearity, and prevent the curse of dimensionality. This streamlined feature set improved model efficiency and supported faster, more stable

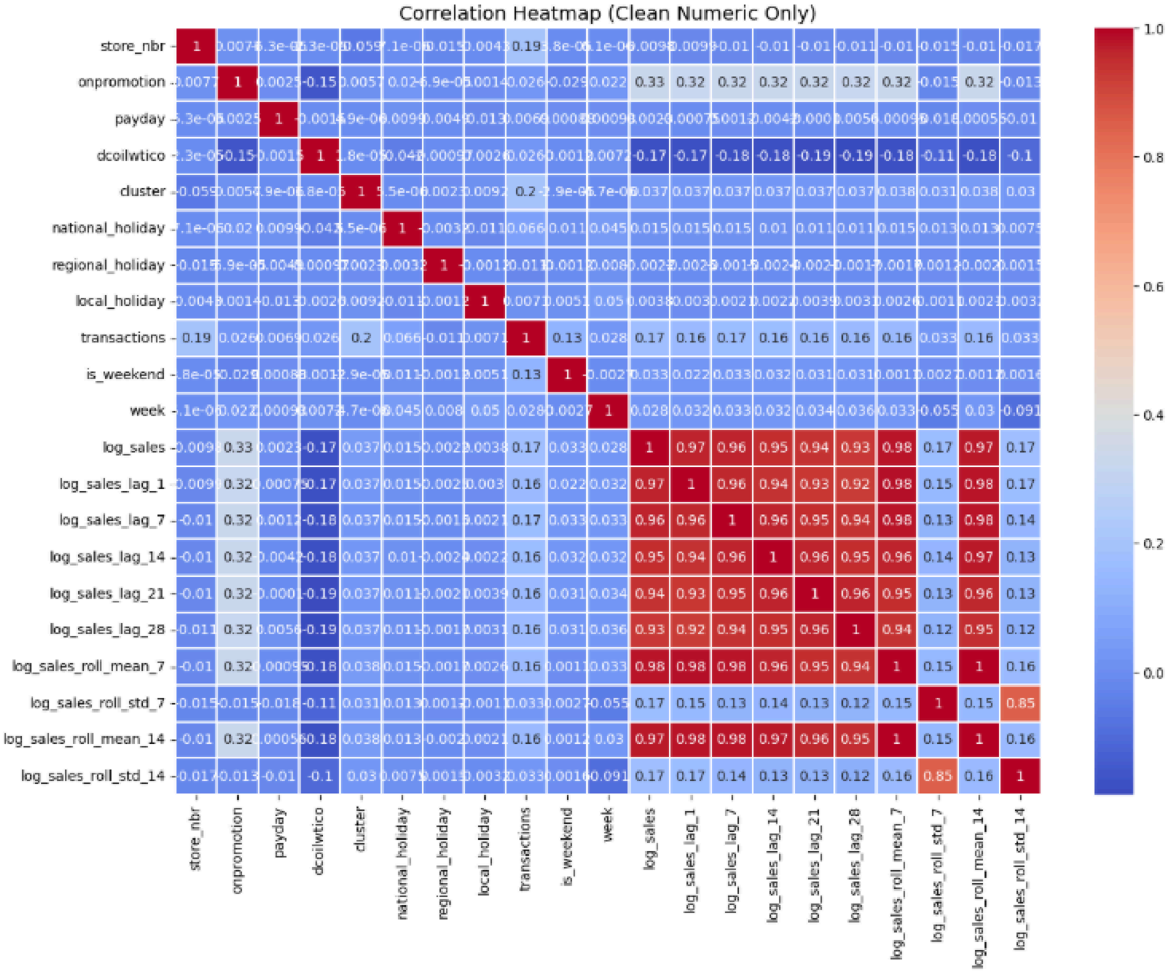


Figure 7. Correlation Heatmap Used for Feature Selection

1.11 Forecasting Model Development & Optimization

The model development phase aimed to compare multiple machine-learning algorithms and identify the most accurate forecasting method for the sales time-series dataset. After completing data preprocessing and feature engineering, several regression models were trained, evaluated, and optimized using the transformed target variable sales_log.

The following subsections summarize the full modeling workflow based on the code and results in the uploaded notebook.

1.11.1 Models Implemented

According to the notebook, four main models were developed and tested:

1. SGDRegressor

A linear model trained using stochastic gradient descent. It was used mainly as a baseline model, but its performance was significantly lower than tree-based models.

2. XGBoost Regressor

A gradient-boosted tree model. We trained it on the engineered dataset (after dropping id, date, and target and other features like holidays and payable day), predicted on the validation set, applied np.expm1() to reverse the log-transform, and computed RMSLE.

3. CatBoost Regressor:

Trained with categorical features detected automatically.

Provided stable results but still below the best model**.**

4. LightGBM Regressor:

This was the best-performing model in your notebook.

You tuned the parameters, used early stopping, predicted using the best iteration, then reversed the log transform using np.expm1().

The resulting RMSLE score was the lowest among all models**.**

✔ **LightGBM:** achieved the best accuracy and was selected as the final forecasting model.

1.11.2 Train–Validation Split

We used:

- X_train, X_valid extracted after preprocessing and feature selection
- The target variable transformed as:
  y_train_log = np.log1p(train['sales'])

This ensures stable variance and more robust training. Validation performance was consistently monitored using RMSLE.

1.11.3 Model Evaluation Metric

We evaluated all models using Root Mean Squared Logarithmic Error (RMSLE)

Because:

- sales values contain extreme spikes (promotion periods, holidays)

log-transform helps stabilize high-magnitude differences

✔ This makes RMSLE the most appropriate metric for retail demand forecasting.

1.11.4 Model Comparison

Table 2: Model Comparison

Model	Performance Notes
SGDRegressor	Lowest accuracy; used only as baseline
XGBoost	Good performance; required careful handling of log transform
CatBoost	Stable with categorical features; slightly underperformed
LightGBM	Best RMSLE score and fastest training

✔ Based on these results, LightGBM was selected as the final model for deployment.

1.12 Final Forecasting Pipeline

The final forecasting pipeline integrates all data processing, feature engineering, and modeling steps into a single coherent workflow. This pipeline ensures repeatability, scalability, and the ability to generate accurate forecasts when new data becomes available.

The complete pipeline includes the following stages:

1. Data Ingestion

Loading raw datasets (train, test, oil, stores, holidays, transactions) and converting them into structured DataFrames.

2. Preprocessing & Cleaning

- Handling missing values (forward-fill for oil prices, zero-fill for holidays).
- Converting data types (datetime, categorical, numeric).
- Outlier analysis without removal.
- Creating the sales_log transformed target.

3. Feature Engineering

- Temporal features (year, month, day, weekday, is_weekend).
- Payday and holiday indicators (created earlier in Milestone 1).
- Lag features: 1, 7, 14, 21, 28 days.
- Rolling windows: 7- and 14-day means and standard deviations.
- Merging all tables into one unified time-series dataset.

4. Feature Selection

Using correlation matrices (Figure 7) to exclude weak, redundant, or highly correlated features, for example: id, holiday indicators, weekofyear, payday, several lag/rolling features.

5. Model Training (LightGBM)

- Training using sales_log
- Hyperparameter tuning
- Early stopping for optimal iteration selection
- Predicting on the validation set

6. Post-processing

- Reversing log transform using expm1()
- Computing RMSLE
- Preparing final submission file

7. Deployment Preparation

Integrating the final model into a Streamlit app for interactive forecasting.

1.13 Model Deployment (Streamlit Interface)

To make the forecasting system accessible to non-technical users, a simple and interactive Streamlit dashboard was designed.

● Business benefits

The deployment transforms the model into a practical tool that supports everyday operations such as:

- inventory planning
- promotion optimization
- demand forecasting
- supply-chain adjustments

This implementation makes the project suitable for real-world use in both Ecuadorian-style datasets and future Egyptian datasets.

1.14 Overall Conclusion

This project developed a complete end-to-end machine-learning forecasting system for retail sales prediction using the Kaggle Store Sales dataset. Although the data originates from Ecuador, it closely resembles the core retail dynamics of the Egyptian market, making it an effective prototype for future local deployment.

Key achievements include:

✔ **A fully cleaned and integrated multi-source dataset**

From raw CSV files into one unified time-series structure with 3M+ rows**.**

✔ **Advanced feature engineering**

Including lag features, rolling statistics, oil-price impact, transactions, promotions, and temporal features.

✔ **Rigorous feature selection**

Using correlation matrices to eliminate redundancy, avoid multicollinearity, and reduce dimensionality**.**

✔ **Evaluation of multiple ML models**

SGDRegressor, XGBoost, CatBoost, and LightGBM.

✔ **LightGBM selected as the final model**

It achieved the best RMSLE score and demonstrated strong performance with engineered features.

✔ **Deployment-ready forecasting system**

A Streamlit interface enabling real-time predictions and interactive visualization.

✔ **Adaptability for the Egyptian market**

The model can be retrained easily when real Egyptian retail data becomes available, and its structure already supports:

- local holidays
- promotions
- fuel price changes
- seasonal patterns
- consumer behavior cycles

1.15 Stakeholder Analysis

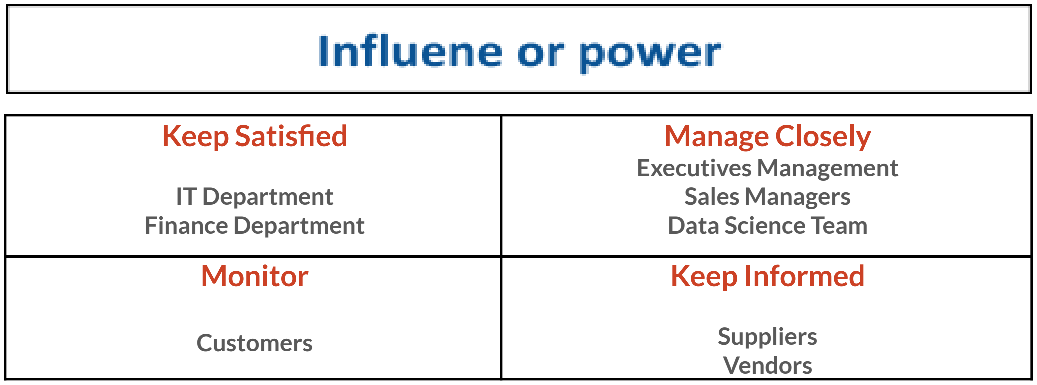
Table 3

Stakeholder Analysis

Stakeholder	Role/Position	Interest in Project	Influence Power	Impact of Project	Engagement Strategy
Executive Management CEO, CFO	Project sponsors Decision-makers	**High** want accurate forecasts for budgeting and strategy	**High**	Improved profitability Better strategic planning	● Regular updates ● Executive summaries ● Dashboards
Sales Director / Sales Managers	End users of forecasts	**High** Rely on forecasts for targets and planning	**High**	Better sales plan Enhance stock management	Include in requirement gathering, periodic feedback sessions

Data Analysts / Data Scientists Team	Development Analysis	**High** Responsible for building and maintaining the model	**High technical**	Skill enhancement, increased workload	Agile collaboration, clear project scope, technical workshops
Finance Department	Budgeting Financial planning	**Medium** Forecasts inform cash flow and inventory costs	**High**	Improved budget accuracy	Monthly reports, integrated forecast dashboards
IT Department	Infrastructure System support	**Medium** Ensure data integration and system stability	**High**	System reliability and security	Coordination on tools, access management, maintenance schedules
Marketing Department	Campaign planners	**Medium** Depend on forecasts to align promotions	**Medium**	More effective campaign targeting	Collaborative meetings, data-sharing sessions
Vendors / Suppliers	Provide products based on demand forecasts	**Medium**	**Low**	Stable orders Reduced variability	Share aggregate forecasts periodically
Customers	Indirect beneficiaries	**Low** Benefit from product availability and stable pricing	**Low**	Better service Fewer stock outs	Indirect engagement via improved service metrics

Stakeholder Matrix



Influene or power

Keep Satisfied IT Department Finance Department	Manage Closely Executives Management Sales Managers Data Science Team
Monitor Customers	Keep Informed Suppliers Vendors

Figure 8: Stakeholder Interest in Project

1.16 Project Milestones Overview

The project was organized into a series of structured milestones to ensure a clear, systematic workflow—from data preparation and exploration to feature engineering, modeling, and evaluation. Each milestone was built on the previous one, allowing the work to progress efficiently and consistently. The following table summarizes the key milestones completed throughout the project.

Table 4

🔵 Project Milestones Summary 🔵

Milestone	Duration	Objectives	Key Tasks	Deliverables
1. Data Collection, Exploration & Preprocessing	Week 1	Collect and understand the dataset	• Load all raw datasets (train, test, stores , oil, holidays, transactions) • Explore trends, seasonality, missing values • Clean data (missing, duplicates, inconsistencies) • Detect and analyze outliers • Apply log transformation to sales • Create date-based features (year, month, week, day, etc.) • Add payday feature • Merge all datasets using date & store_nbr	• EDA Report • EDA Notebook (plots, heatmaps, trends) • Cleaned & Processed Dataset
2. Data Analysis, Visualization & Feature Engineering	Week 1	Analyze relationships & engineer powerful features	• Correlation analysis • Seasonal analysis (weekly, monthly, yearly) • Create rolling features (mean/std) • Create lag features (1,7,14,21,28) • Create holiday indicators • Select important features using correlation results	• Feature Engineering Summary • Advanced Visualizations • Updated cleaned dataset

Table 5

🔵 Project Milestones Summary (Continuation of Table 4) 🔵

Milestone	Duration	Objectives	Key Tasks	Deliverables
3. Forecasting Model Development & Optimization	Week 1	Build and optimize forecasting models	• Train models (LightGBM, XGBoost, CatBoost, SGD) • Time-based train/validation split • Hyperparameter tuning • Compare performance usingRSMLE • Select best model (LightGBM)	• Model Evaluation Report • Model Training Code • Final Selected Model
4. Deployment & Monitoring	Week 1	Deploy model & set up monitoring	• Build prediction app using Streamlit	• Deployed Streamlit App
5. Final Documentation & Presentation	Week 1	Prepare final report & present results	• Write full project documentation • Create presentation slides • Demonstrate forecasting model • Highlight business impact & future improvements	• Final Project Report • Presentation Slides

References

Kaggle. (2023). ***Store Sales – Time Series Forecasting.***
Retrieved from https://www.kaggle.com/competitions/store-sales-time-series-forecasting

Microsoft. (2024). *LightGBM documentation.*
Retrieved from https://lightgbm.readthedocs.io/

Chen, T., & Guestrin, C. (2016). *XGBoost: A scalable tree boosting system.*
Retrieved from https://xgboost.readthedocs.io/

Yandex. (2024). *CatBoost documentation.*
Retrieved from https://catboost.ai/

Pandas Development Team. (2024). *Pandas documentation.*
Retrieved from https://pandas.pydata.org/docs/

Scikit-learn Developers. (2024). *Scikit-learn: Machine learning in Python.*
Retrieved from https://scikit-learn.org/stable/

Hunter, J. D. et al. (2024). *Matplotlib documentation.*
Retrieved from https://matplotlib.org/

Waskom, M. et al. (2024). *Seaborn documentation.*
Retrieved from https://seaborn.pydata.org/