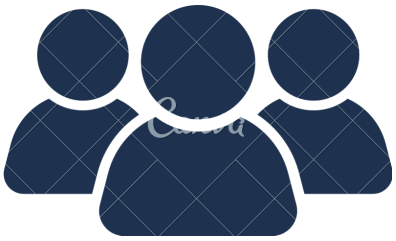
SALES FORECASTING AND OPTIMIZATION

PRESENTED TO:

Digital Egypt Pioneers

OUR TEAM



Hana Ahmed Nabhan

Seif Eldin Omar

Mohammed Rabey Mohammed

Mohamed Walid Salah

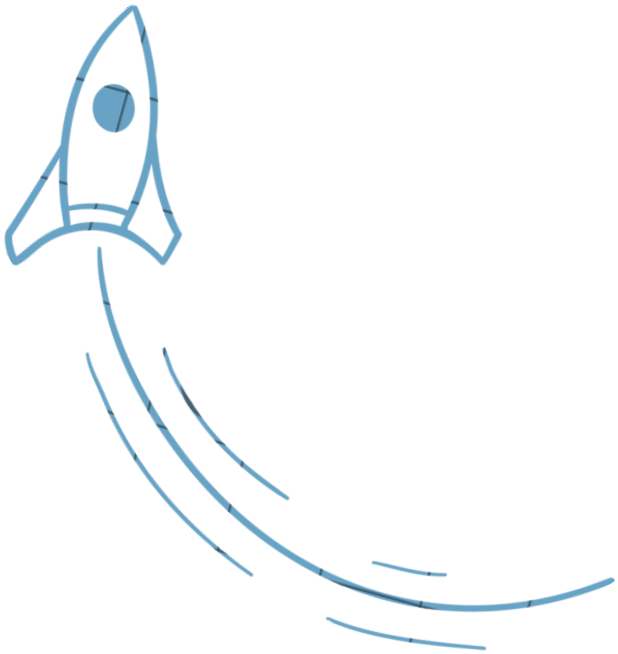
Ali Sheriff Salaheldin

Abdelhamid Eisa ElSharnouby

TABLE OF CONTENT

TABLE OF CONTENT

Introduction



Introduction

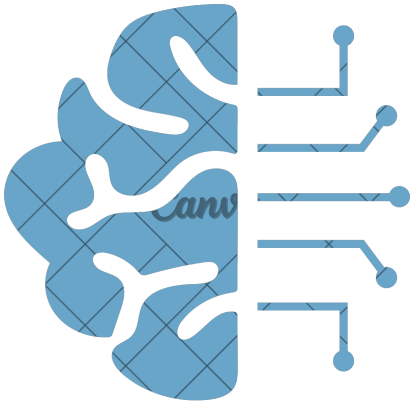
In today’s data-driven retail and e-commerce landscapes, accurate sales forecasting plays a crucial role in strategic planning and operational efficiency. From inventory management to marketing campaigns and resource allocation, forecasting empowers businesses to anticipate demand, reduce waste, and boost profitability. However, traditional forecasting methods often fall short when faced with complex, high-volume data and volatile market conditions.

Your paragraph text

This project, titled **"Sales Forecasting and Optimization"**, addresses this challenge by leveraging historical sales data to build robust forecasting models, optimize decision-making processes, and deploy practical solutions. Through a systematic approach that includes data collection, cleaning, exploratory analysis, model development, and deployment, the project aims to equip businesses with predictive tools that can significantly enhance sales performance and operational planning.

Introduction

The core objective of the project is twofold:



- To accurately forecast future sales using statistical, machine learning, and deep learning models.
- To optimize sales strategies and business decisions by integrating model outputs into real-world applications through dashboards and deployment platforms.

The methodology combines time-series analysis with modern MLOps practices to ensure the developed solutions are not only accurate but also scalable, maintainable, and easy to integrate into existing business systems. By the end of this project, we deliver a deployed forecasting application supported by a clean, version-controlled pipeline, comprehensive visualizations, and actionable business insights.

Your paragraph text

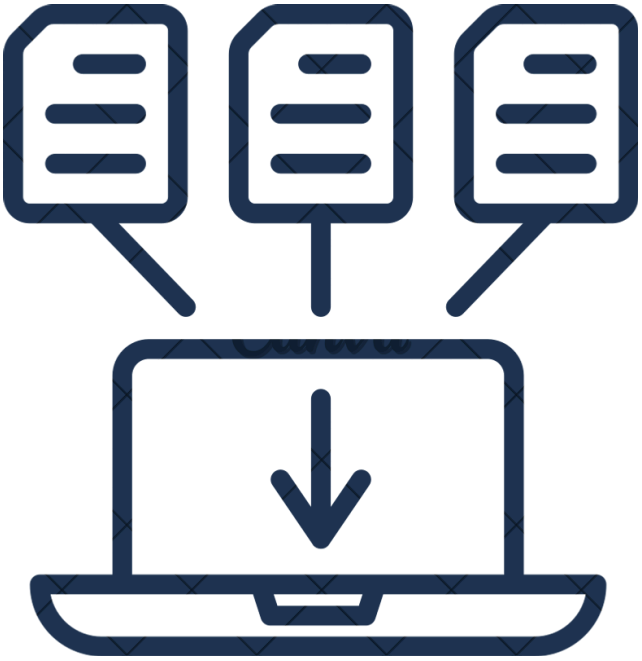
This report documents the entire journey of the project—starting from data acquisition to the final deployment—highlighting the methodologies used, results achieved, and the tangible business value created through the application of forecasting and optimization techniques.

Introduction

Project Structure

- deployment
- mlruns
- models
- notebooks
- plots
- reports
- scripts
- visualizations
- .gitignore
- README.md
- requirements.txt

Data Collection



Data Collection

- **Data Source:**

The dataset is publicly available on Kaggle as part of the Store Sales Time Series Forecasting competition. It includes historical sales data for stores in Ecuador, along with external factors such as oil prices and holidays that may influence sales.

- **Dataset Description:**

Size:

- The dataset contains 3,054,348 rows across multiple files.
- Time Period: Data spans from 2013 to 2017, with both training and test data available for forecasting.

Files:

- train.csv: Contains the sales data for the training set.
- stores.csv: Includes metadata about the stores.
- oil.csv: Provides daily oil prices.
- holidays_events.csv: Contains data on holidays and special events.

Data Cleaning and Preprocessing

1.Missing Values:

- **Holiday-related Columns**: Columns like holiday_type, locale, locale_name, description, and transferred have missing values. We will analyze and decide whether to impute or ignore these based on their relevance.
- **Oil Prices**: The dcoilwtico column has missing values, which will be handled with appropriate imputation methods (e.g., forward-fill or rolling average).
- **Transactions:** Missing values in the transactions column will be assessed and imputed if necessary.

2.Duplicates:

- We have verified that there are no duplicates in the dataset, so no further action is needed in this regard.

3.Outliers:

- **Sales:** There are 2,788,335 outliers in the sales column, indicating extreme sales events. We will review these instances and decide whether to transform or cap the outliers, depending on their relevance.
- **Transactions:** There are 154,308 outliers in the transactions column, suggesting abnormal transaction counts. These will be investigated for potential removal or transformation.

Exploratory Data Analysis

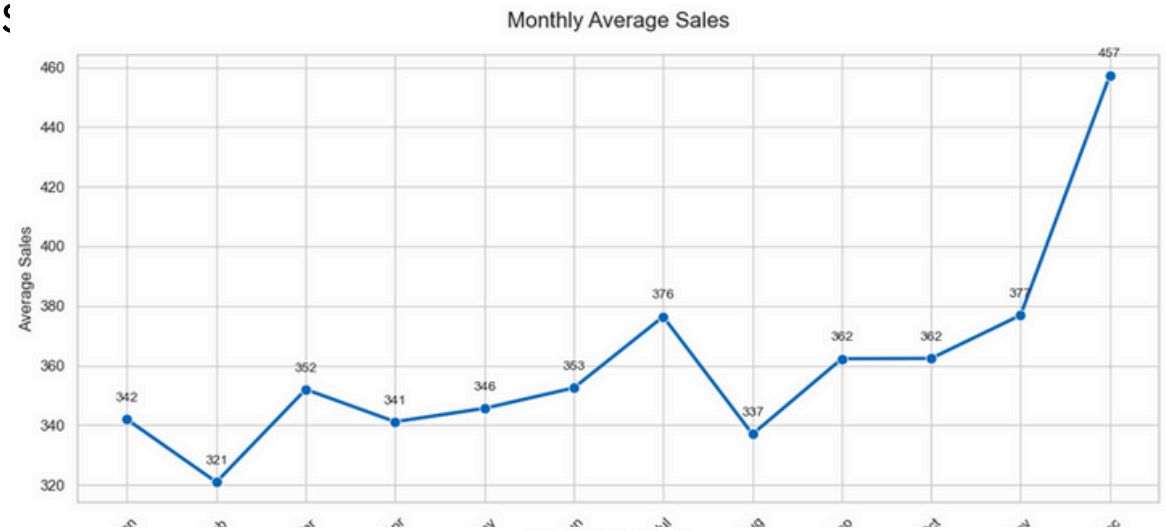
Exploratory Data Analysis

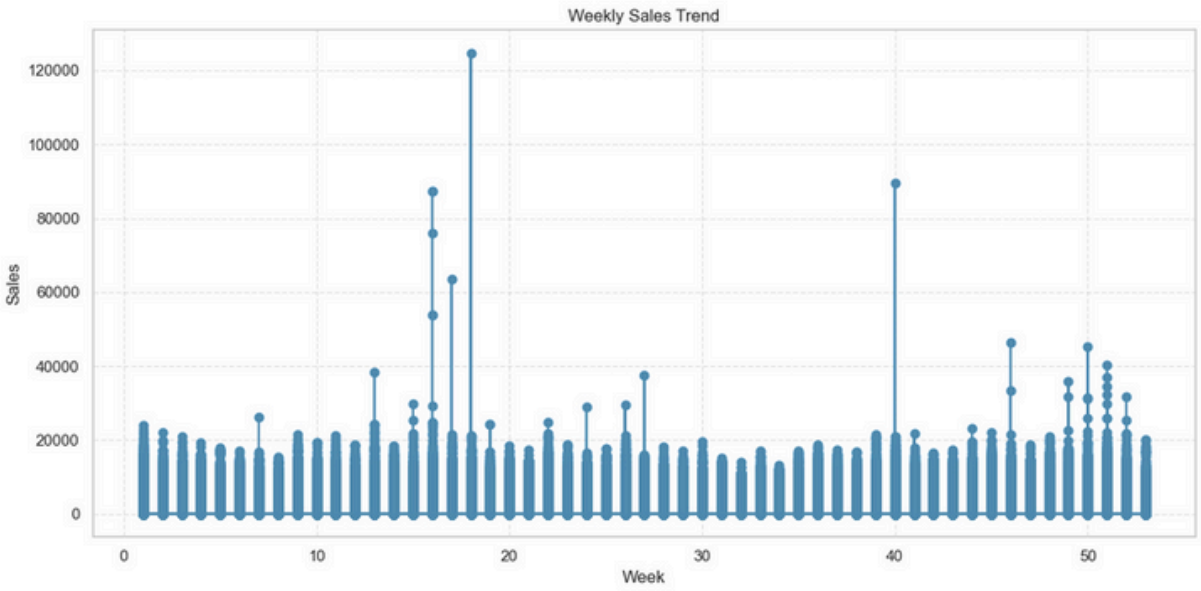
1. Trends and Patterns

The sales data exhibited clear upward and downward trends influenced by time, store location, and product family. Notable observations include:

- **Weekly and Monthly Trends:**

Sales followed a repeating weekly pattern, with peaks typically observed on weekends and promotional periods

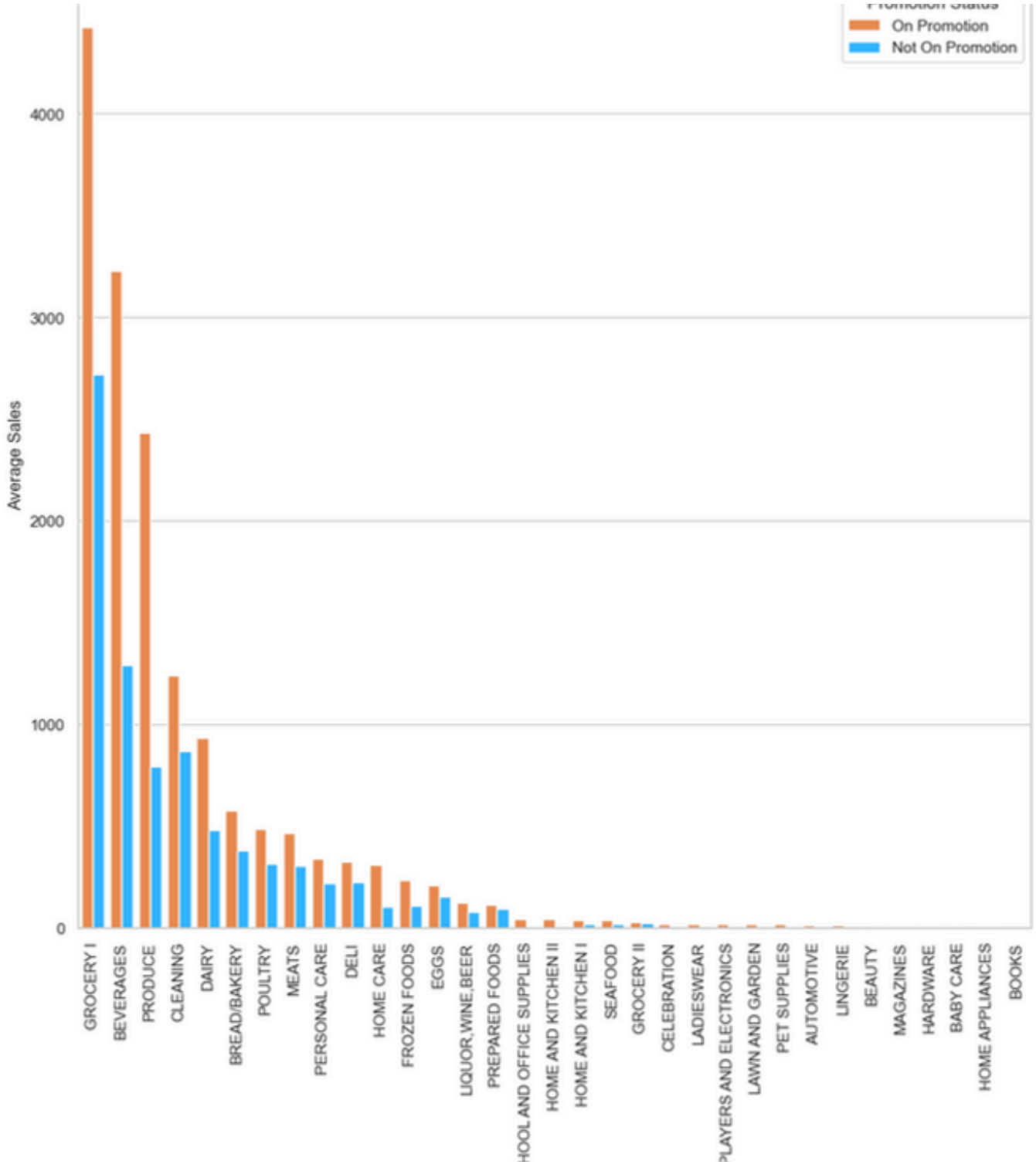




Exploratory Data Analysis

- **Growth Patterns:**

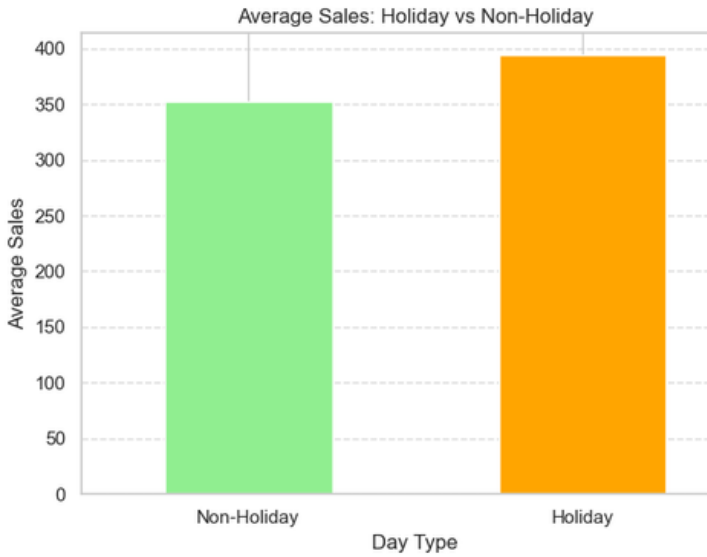
A general upward trend was visible in some product categories, indicating expanding market demand.

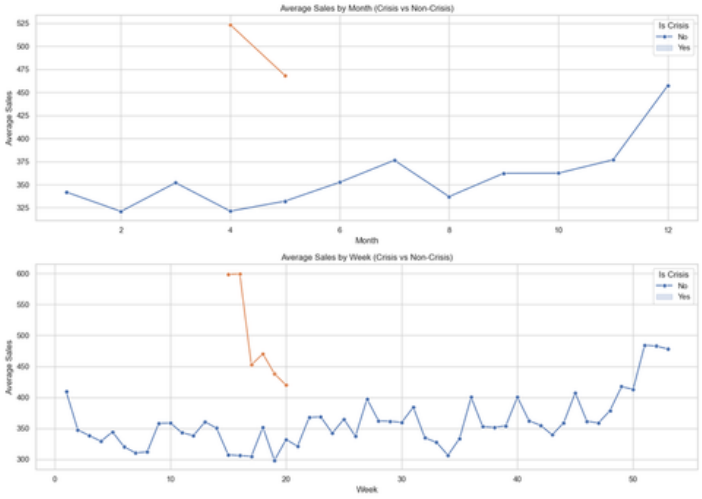


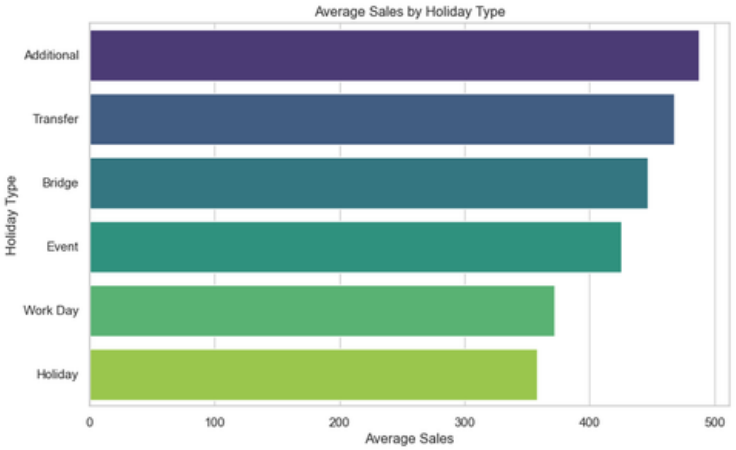
Exploratory Data Analysis

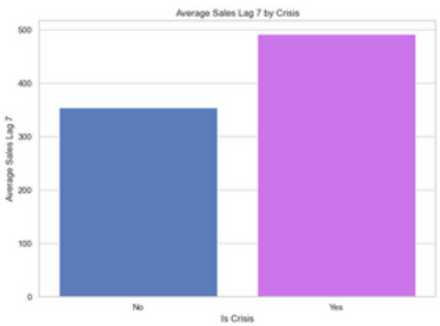
- **Anomalies and Dips**

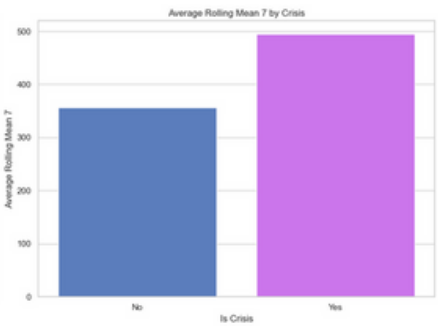
Contrary to typical retail patterns, sharp spikes in sales were observed during national crises and major holidays. These increases likely reflect real-world behavioral responses such as panic buying, supply concerns, or increased travel. Rather than disrupting operations, these events appear to have temporarily boosted demand, creating distinct anomalies in the sales trend.

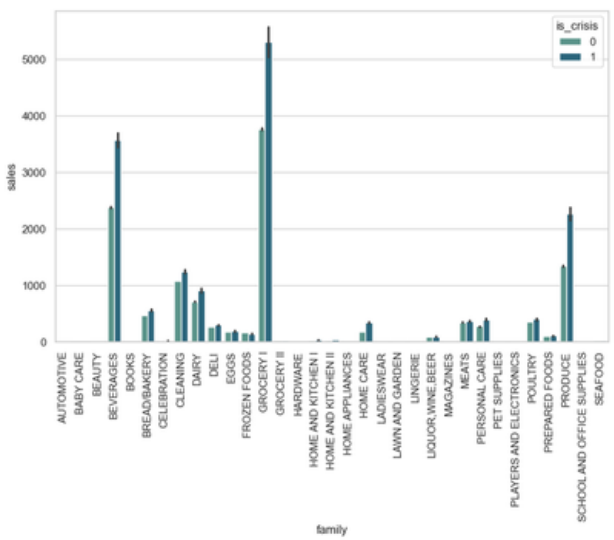












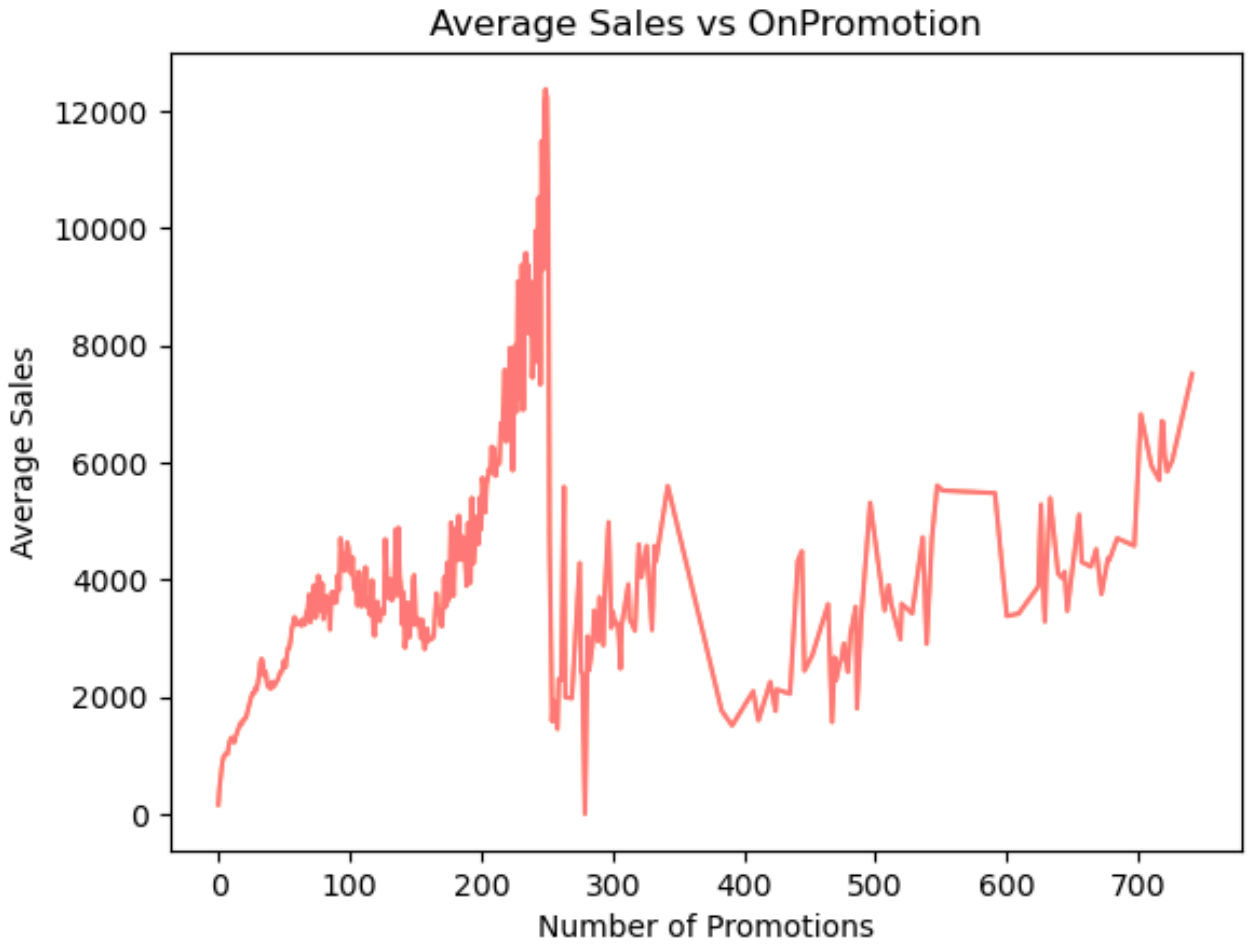
Exploratory Data Analysis

2-Correlation Analysis

Correlation analysis was performed to understand relationships between key numerical features

- **Sales and Promotions:**

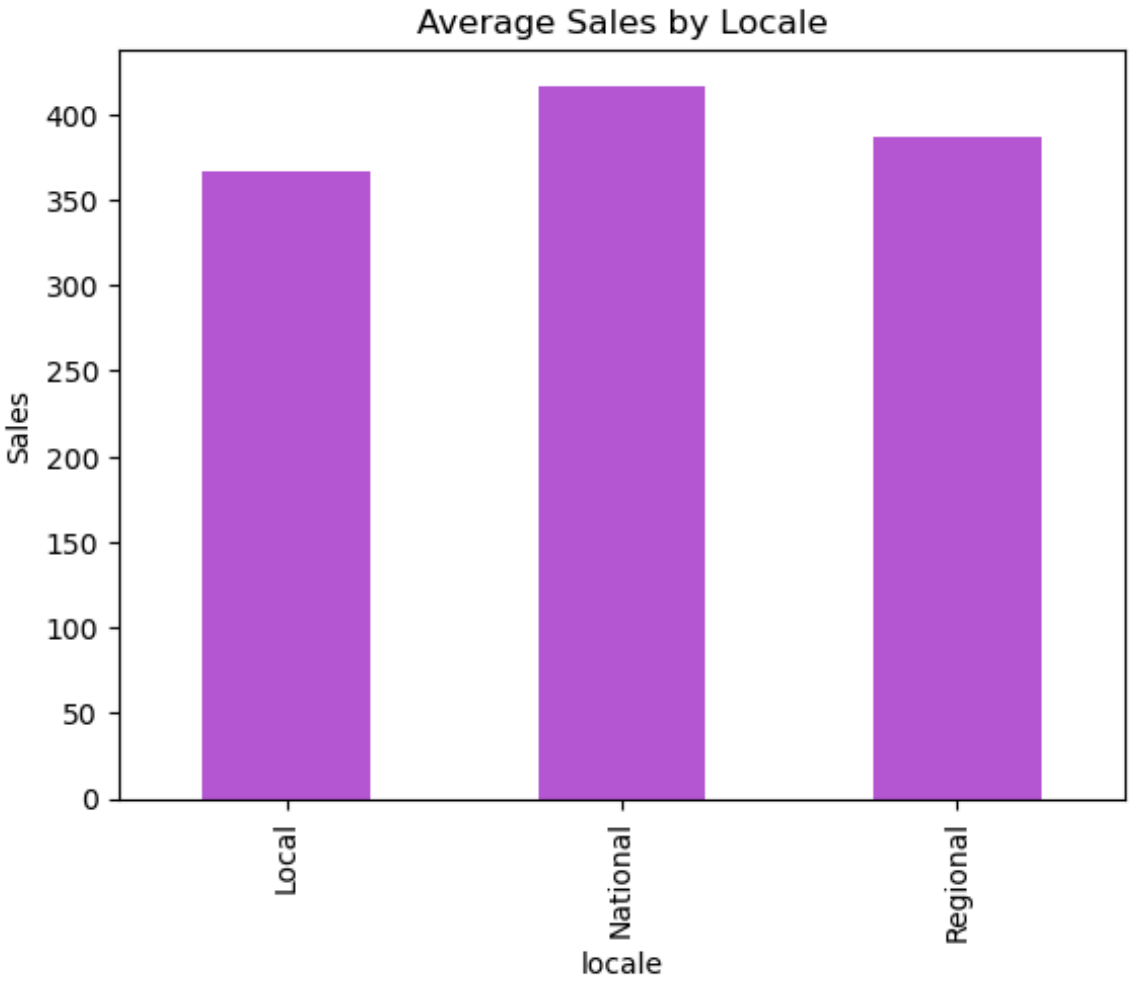
Strong positive correlation observed between promotional campaigns and increased sales volume.



Exploratory Data Analysis

- **Sales Patterns Across Locales**

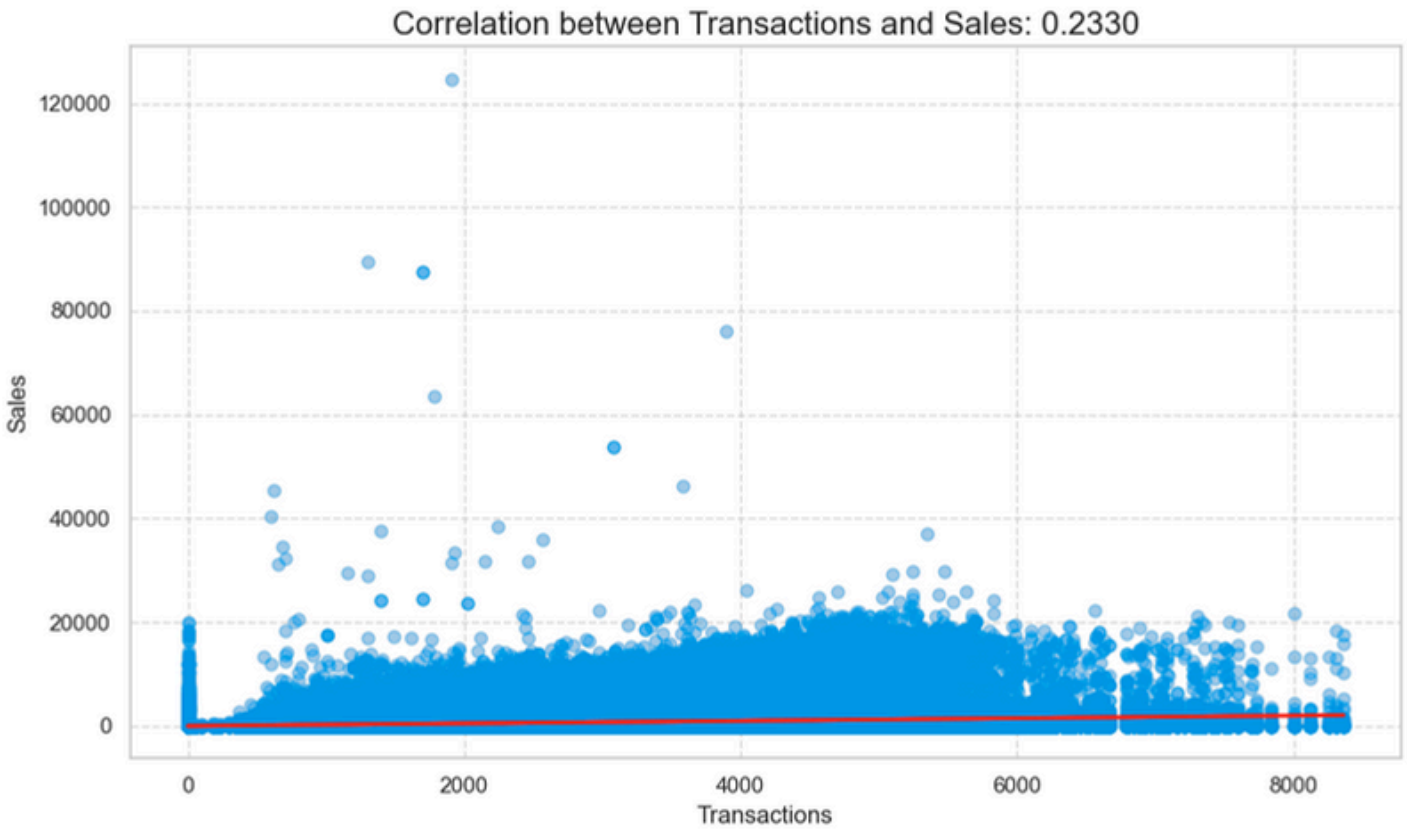
Sales trends varied across national, regional, and local locales. National-level holidays and events had the most significant impact on sales volumes, while local promotions and store-specific events caused more subtle fluctuations. This suggests that the geographic scope of an event or campaign plays a key role in influencing sales behavior.



Exploratory Data Analysis

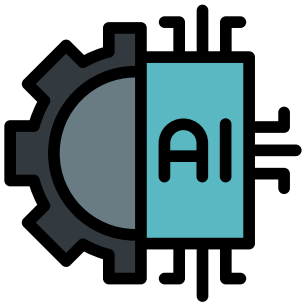
- **Transactions and Sales:**

High positive correlation suggests transaction counts can act as a reliable proxy for sales performance.



Modeling

Model Overview



To forecast daily sales accurately, multiple machine learning models were developed, evaluated, and compared based on their performance. The modeling pipeline included data preprocessing, feature engineering, and hyperparameter tuning. The final models aimed to capture temporal trends, seasonality, promotional effects, and store-family-specific patterns.

Modeling Approach:

We adopted a multi-model strategy for time series sales forecasting, exploring statistical, machine learning, and deep learning models to compare performance and extract insights from different perspectives.

1. Baseline Model

Linear Regression (baseline_model_linear_regression.ipynb):

Served as the benchmark model. It incorporated historical sales and selected features to set a performance baseline for all subsequent models.

Model Overview

2. Statistical Models

- ARIMA (Arima.ipynb):

Used to capture temporal dependencies and trends. Best suited for stationary series after differencing.

- SARIMA (SARIMA.ipynb):

An extension of ARIMA that handled both seasonality and trend. Tuned based on AIC and performance metrics.

both Attempted model fitting, but the dataset was too large, leading to significant computational challenges. The resulting R^2 score was negative, indicating poor model performance.

- Prophet (Prophet.ipynb):

Developed by Meta, this model provided flexibility in capturing trend and seasonality, and handled holidays and missing data gracefully. Useful for explainability and visualization.

Model Overview

3. Machine Learning Models

- XGBoost Regressor (XGboost_Model.ipynb):

Chosen as the primary model for its ability to model complex, non-linear relationships, handle missing values natively, and leverage historical lag features effectively. Provided strong performance and interpretability.

4. Deep Learning Models

- LSTM Networks (LSTM.ipynb, trial_lstm.ipynb):

Recurrent neural networks that modeled sequential dependencies in sales data. Although computationally intensive, they were valuable for long-range pattern recognition.

Results

Results and Insights

To evaluate different forecasting approaches, we compared multiple models based on their predictive accuracy and training efficiency. Metrics used include Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R^2 Score, along with training time to assess computational cost.

The models tested were:

1. A Baseline Linear Regression to provide a simple benchmark,
2. A more advanced XGBoost model for capturing complex patterns,
3. A Long Short-Term Memory (LSTM) network for deep temporal learning,
4. Prophet

Model Evaluation

1. Baseline Model

- MSE: 178504.86
- RMSE: 422.5
- R^2 Score: 0.90
- Training time: 1 sec

2. XGBoost

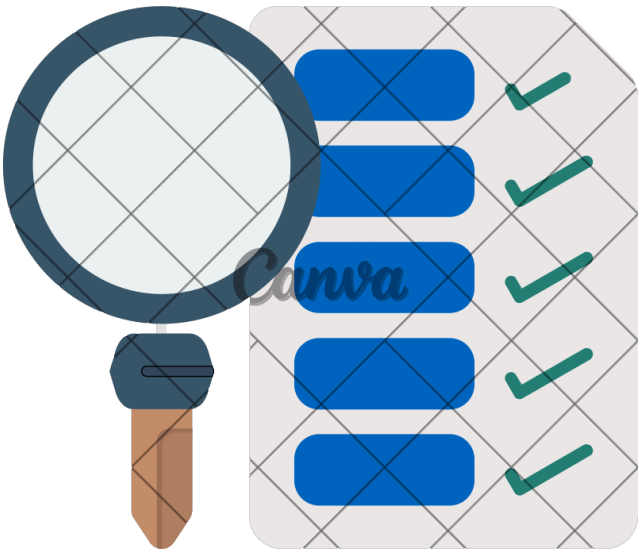
- MSE: 148600.95
- RMSE: 385.49
- R^2 Score: 0.93
- Training time: 25 sec

3. LSTM Model

- MSE: 131,067.80
- RMSE: 362.03
- R^2 Score: 0.93
- Training Time : 41 min

4. Prophet

- MSE: 12700
- RMSE: 1130
- R^2 Score: 0.14
- Training Time : 30 sec

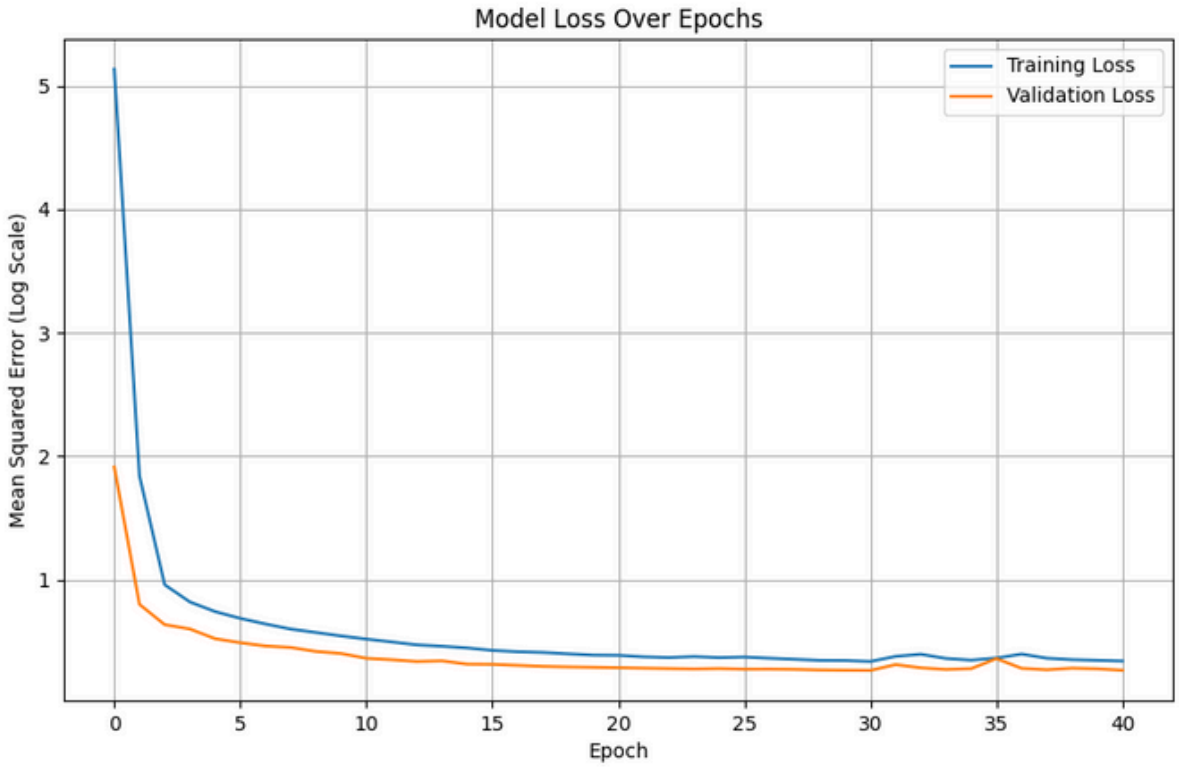


Visualizations

- **LSTM Model**

The plot below shows the LSTM model's training loss across epochs.

The gradual decrease in loss indicates that the model was learning effectively over time, minimizing the error on the training data. A smooth downward trend without major fluctuations suggests stable and consistent training.

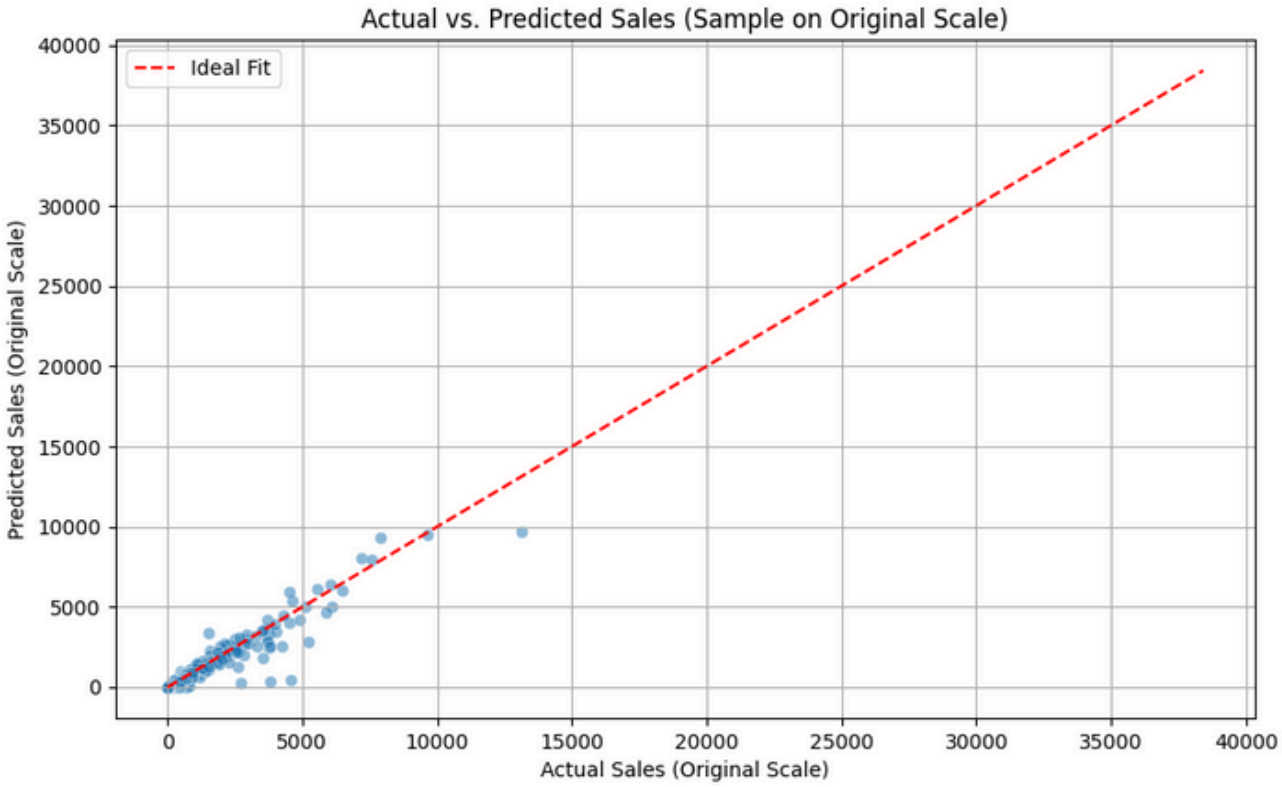


Visualizations

- **LSTM Model**

The plot below illustrates a sample comparison between the LSTM model's predicted sales and the actual sales values.

Within this segment, the LSTM demonstrates a strong ability to follow the trend and magnitude of the actual data, confirming its effectiveness in capturing temporal dependencies.



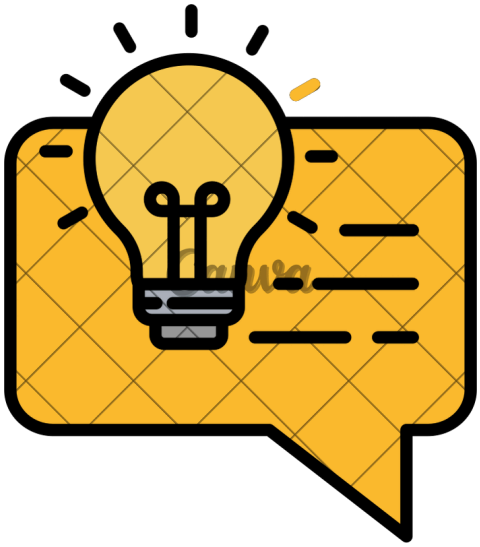
Insights from Model Evaluation

- **Baseline Model (Linear Regression)**

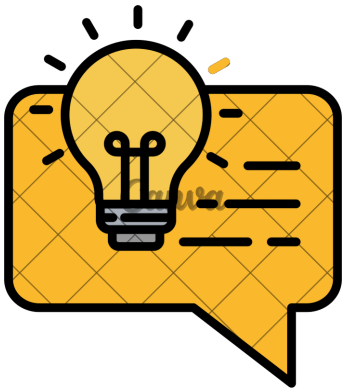
Serves as a simple benchmark with decent performance ($R^2 = 0.90$).
Extremely fast to train (1 sec), but less accurate compared to other models.
Good for quick estimates, but lacks the ability to capture complex patterns.

- **XGBoost**

Offers a solid improvement in both MSE and RMSE over the baseline.
Achieves a strong R^2 score of 0.93, indicating excellent predictive power.
Training time is reasonable (25 sec), making it a good balance between performance and efficiency.



Insights from Model Evaluation



- **LSTM Model**

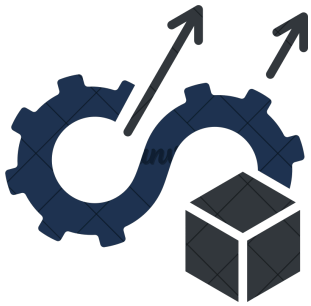
Slightly outperforms XGBoost in MSE and RMSE with the same R^2 score (0.93).
However, it requires significantly more training time (41 minutes), which may not be practical for frequent retraining or real-time applications.
Best suited for applications where accuracy is critical and computational resources are available.

- **Prophet**

Performs poorly compared to other models, with a very high RMSE (1130) and a low R^2 score (0.14), indicating it failed to capture the data's underlying patterns effectively.
Although it trains quickly (30 sec), its performance is insufficient for this dataset.
May be more suitable for datasets with clearer seasonal trends or when interpretability and fast deployment are prioritized over accuracy.

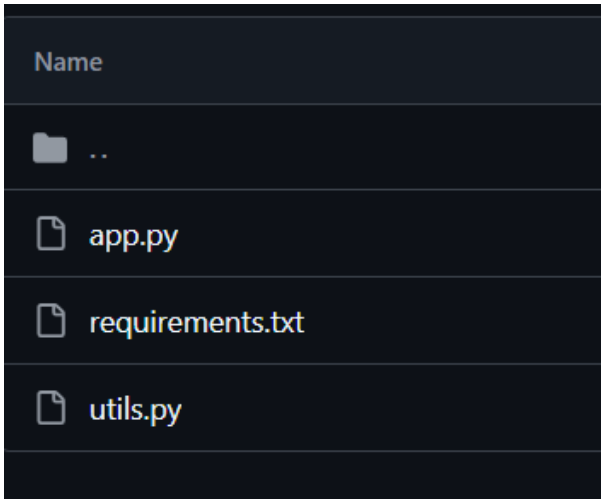
DEPLOYMENT

FINAL APPLICATION OVERVIEW



- To make the forecasting model accessible and interactive for end users, we deployed the solution using Streamlit, a lightweight web framework for building data apps quickly and efficiently.

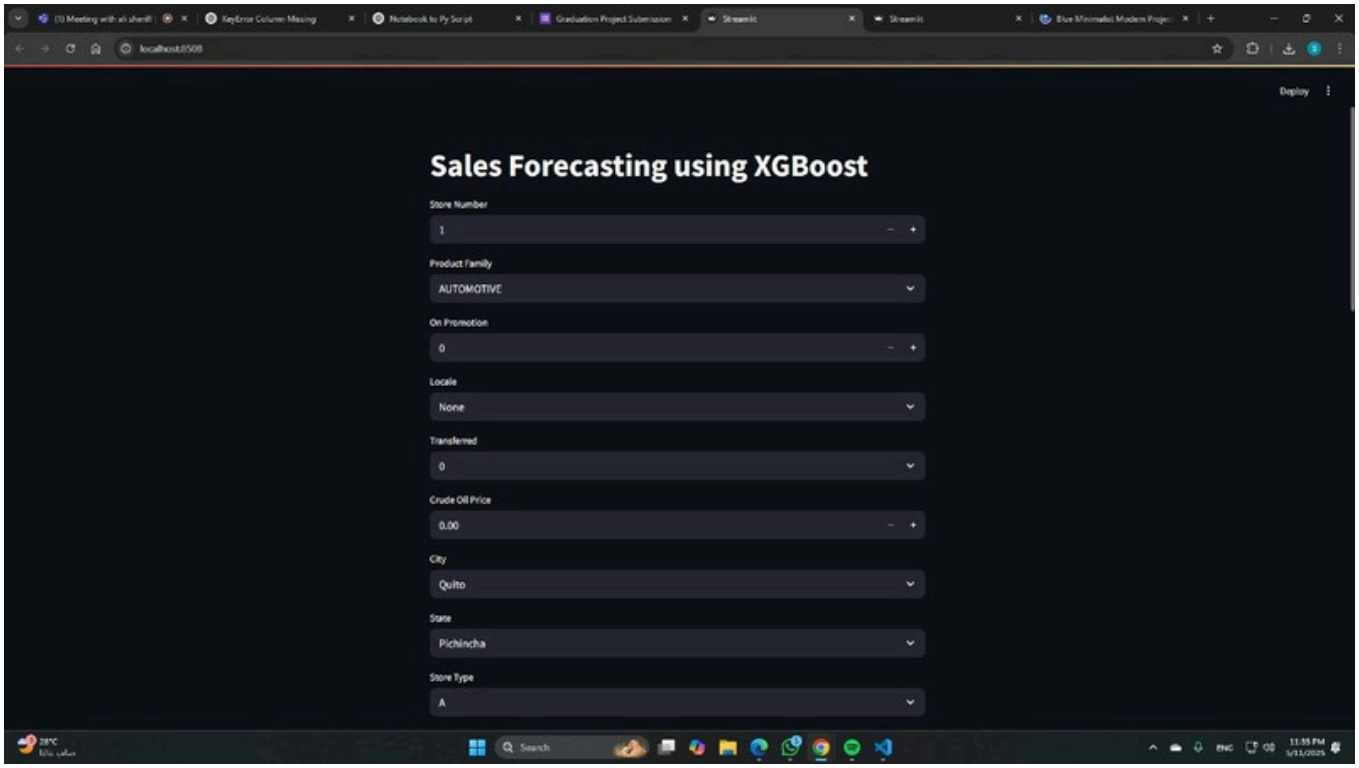
The core files for deployment include:



- app.py: The main Streamlit application that provides a user-friendly interface to upload data, run the XGBoost forecasting model,and visualize predictions.

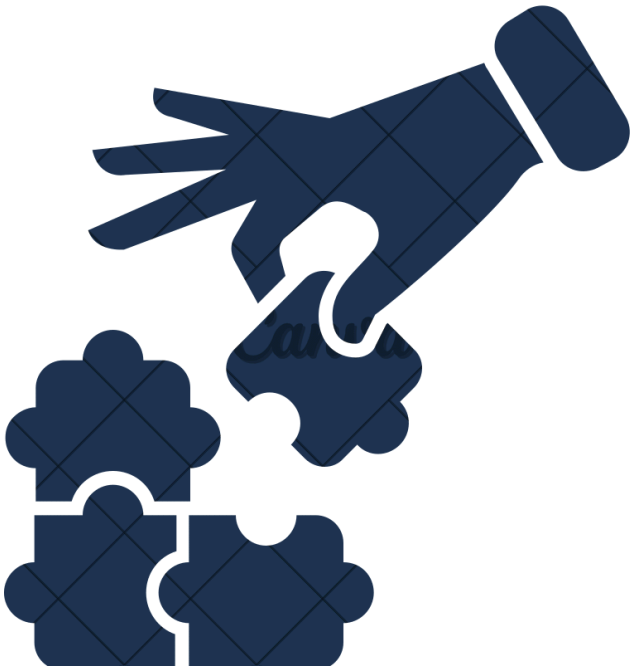
- utils.py: Contains helper functions for data preprocessing, feature engineering, and model inference.
- requirements.txt: Specifies all required Python packages, including xgboost, pandas, scikit-learn, and streamlit.

IMPLEMENTATION STRATEGY



The final deployed sales forecasting application uses a XGBoost regression model. Users input features such as store number, product family, promotion status, crude oil price, and location details to receive a predicted sales value. The interface is designed for ease of use and real-time interaction.

Conclusion and Recommendations



Conclusion and Recommendations

Summary of Outcomes

This project, titled "**Sales Forecasting and Optimization,**" set out to build accurate and scalable sales prediction models that can empower businesses to make data-driven decisions. By leveraging historical sales data, the team implemented and compared various models, ranging from traditional statistical approaches to state-of-the-art machine learning and deep learning methods. The outcomes of each model are summarized below:

Baseline Model (Linear Regression):

Served as a simple benchmark with decent performance ($R^2 = 0.90$).

Extremely fast training time (~1 second), but limited in capturing complex patterns.

Useful for quick estimates but less reliable for nuanced forecasting.

XGBoost:

Provided a significant improvement over the baseline with a strong R^2 score of 0.93.

Conclusion and Recommendations

Summary of Outcomes

Balanced performance and efficiency, with moderate training time (~25 seconds).

A reliable choice for production environments requiring both speed and accuracy.

LSTM Model:

Slightly outperformed XGBoost in terms of MSE and RMSE while achieving the same R^2 (0.93).

Required considerably longer training time (~41 minutes), limiting its practicality for frequent retraining.

Best suited for high-accuracy applications where computational resources are not a constraint.

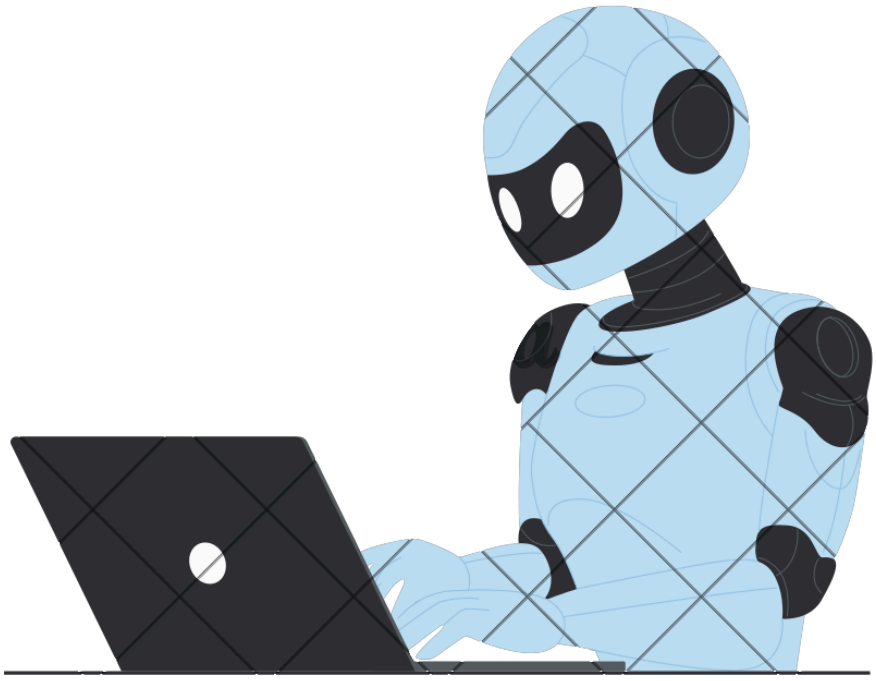
Prophet:

Underperformed significantly with a high RMSE (1130) and low R^2 (0.14).

While training was fast (~30 seconds), its poor accuracy makes it unsuitable for this dataset.

More appropriate for datasets with strong seasonal patterns or where model interpretability is essential.

Future Work



Limitations and Future Work

Training Time for Deep Models:

- LSTM’s long training time hinders real-time adaptability. Future work can explore optimized architectures or model distillation for faster inference.

Feature Limitations:

- Additional external factors such as holidays, weather data, and macroeconomic indicators could further improve model accuracy.

Limitations and Future Work

- **Model Retraining Pipeline:**

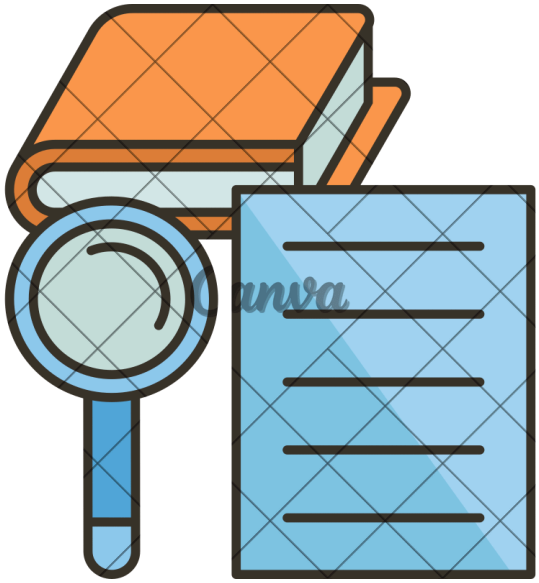
Implementing automated model retraining with CI/CD pipelines would improve scalability and ensure predictions remain accurate over time.

- **Extended Use Cases:**

Expanding the model to forecast at the SKU level or across new regions can widen its application scope.

References and Appendices

Appendices



Programming Language
Python
Data Manipulation & Analysis
Pandas
NumPy
Visualization
Matplotlib & Seaborn
Machine Learning & Forecasting Models
Scikit-learn
XGBoost
Keras (with TensorFlow backend): Used to build and train the **LSTM** deep learning model.
Facebook Prophet
Model Evaluation
Scikit-learn Metrics
Cross-validation tools
Deployment & Dashboard
Streamlit
Version Control & Collaboration
Git & GitHub