**STORE SALES FORECASTING**

Team 7:

Nishita Matlani & Dhir Thacker

**Introduction**

**Background**

The Store Sales Forecasting project is driven by the necessity to navigate the challenges of retail management in an economically volatile environment, which heavily relies on oil revenues. This dependency on oil prices introduces significant fluctuations in the economy, directly impacting consumer purchasing power and, consequently, retail sales. Retailers face the ongoing challenge of matching supply with variable demand, necessitating accurate sales forecasts to minimize costs associated with overstocking or understocking and optimize staffing levels. The project addresses these needs by developing predictive models that can forecast sales with high accuracy, using a dataset that includes sales history, promotional activities, and external economic factors like oil prices and holidays.

**Motivation**

In the heart of Ecuador's fluctuating economy, retailers face the challenge of staying ahead in a rapidly changing consumer landscape. This project is born from the necessity to empower businesses with the foresight to navigate these shifts. By integrating data science and machine learning, our aim is to transform raw sales data into actionable insights. This not only enables informed decision-making on inventory and promotions but also paves the way for a retail environment that adapts in real-time to the ebb and flow of market demands.

**Goal**

Our mission is to harness cutting-edge technology to forge a predictive model that doesn’t just respond to market changes but anticipates them. We envision a tool that offers crystal-clear daily sales predictions across diverse store fronts and product lines, optimizing operations and boosting profitability. Beyond numbers, we aim to decode the complex dance between sales trends and influencing factors like promotions and holidays, offering a deeper understanding of consumer behavior. This is not just about predicting the future; it's about creating it.

**Methodology**

**Data Cleaning and Preprocessing**

The initial phase of the project involved an extensive data cleaning and preprocessing stage, pivotal for ensuring the integrity and reliability of the analysis. This stage included systematic processing of various datasets such as sales data, store information, transaction records, oil prices, and holiday/event data.

**Loading and Initial Cleanup**

Datasets were first loaded into Pandas DataFrames to leverage the library's powerful data manipulation capabilities. The initial cleanup process involved removing duplicate rows and rectifying data inconsistencies. This step was critical to prevent any analytical errors caused by redundant or incorrect data entries.

**Handling Missing Values**

Particular attention was given to handling missing data, as missing values can significantly distort the outcome of data analysis if not addressed appropriately. The oil price dataset, a crucial feature due to the economic significance of oil prices, underwent missing value interpolation to maintain continuity and quality. This method ensured that no gaps would affect the predictive modeling accuracy.

**Validation and Further Processing**

Further data validation and processing included a thorough inspection for any misaligned data types or incorrectly formatted data entries. The process involved adjusting data types where necessary and additional cleaning to meet the analytical requirements of subsequent project stages.

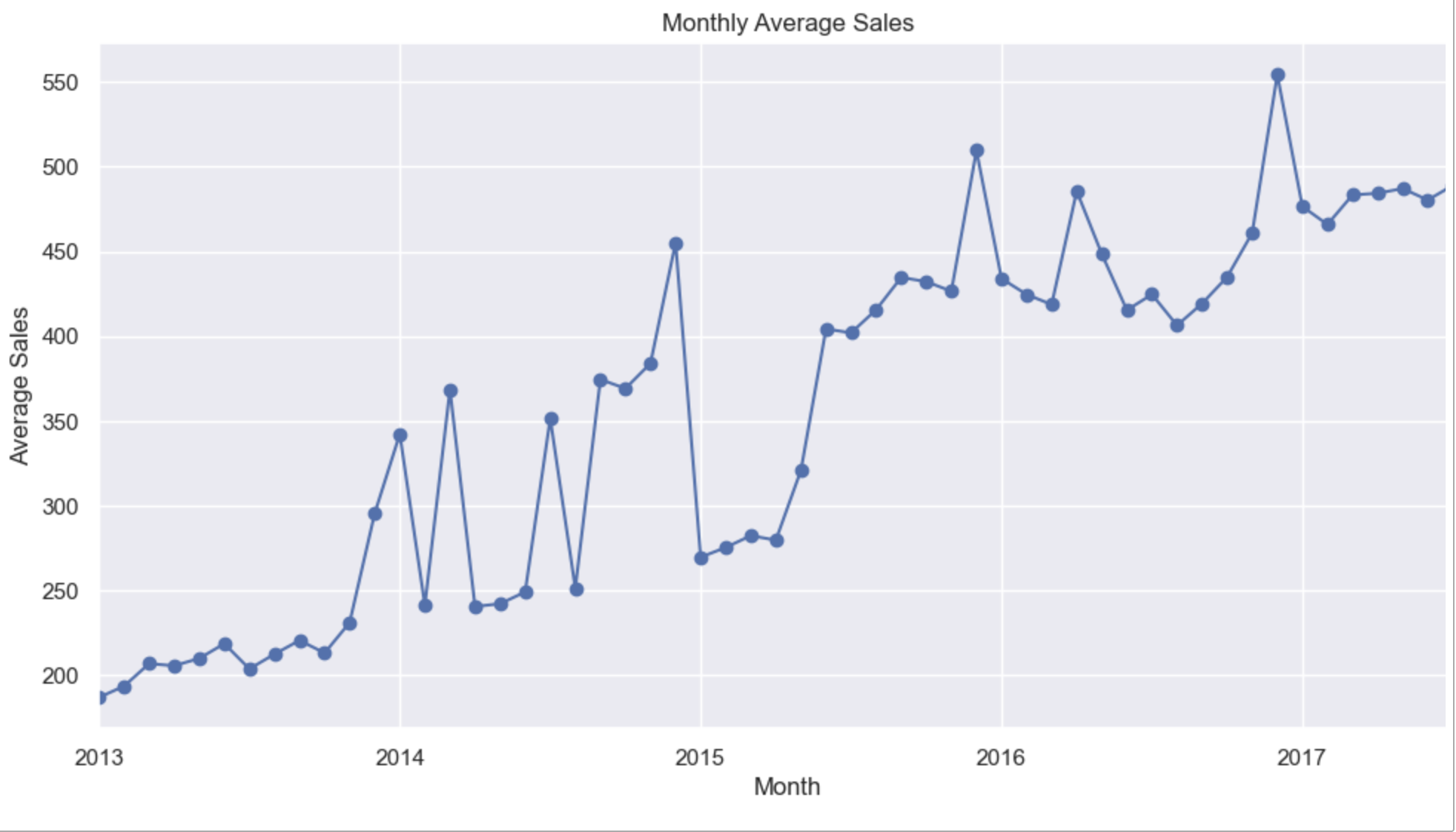
This meticulous approach in the data cleaning and preprocessing phase laid a robust foundation for exploratory data analysis and feature engineering

**Exploratory Data Analysis (EDA)**

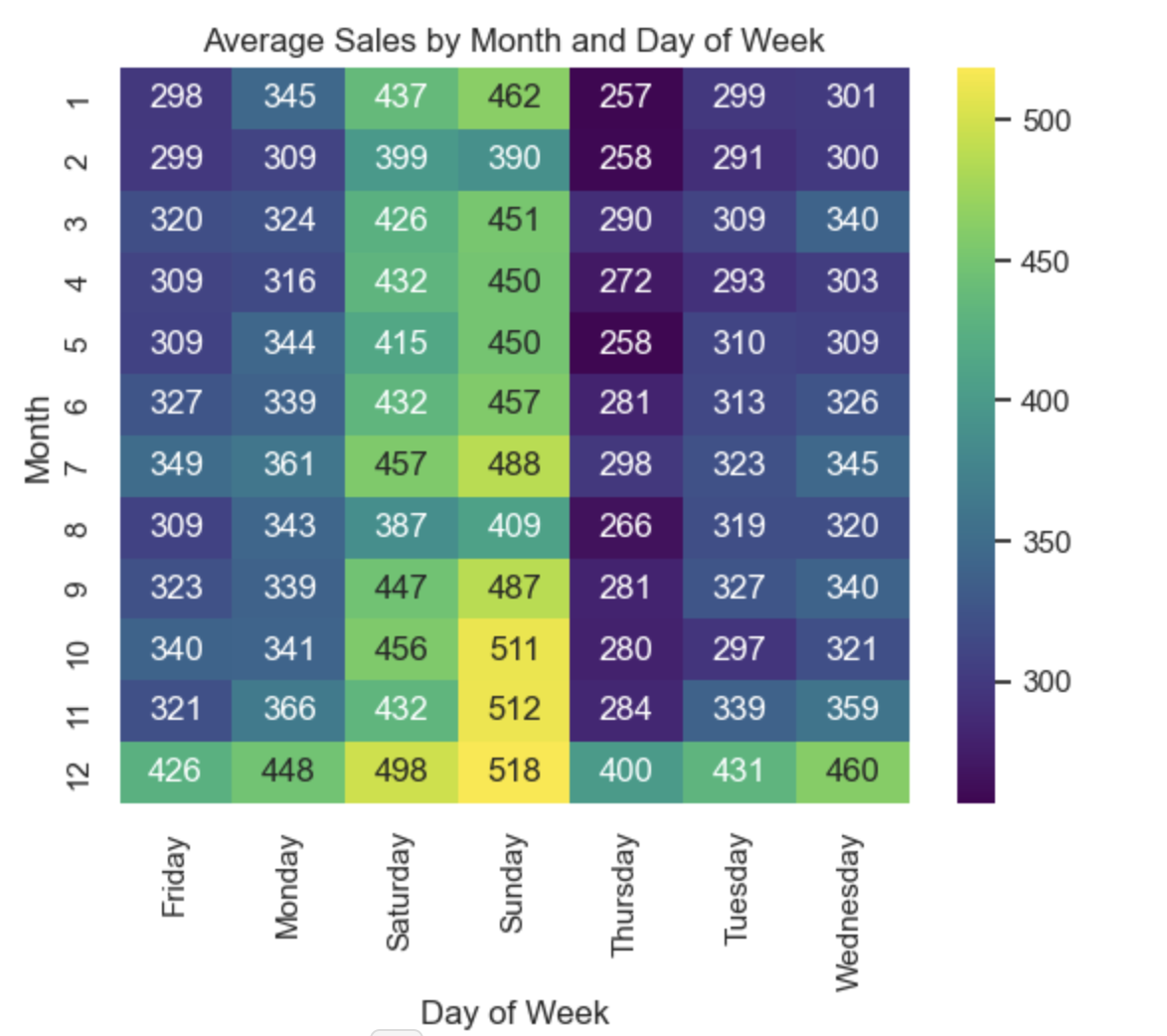
The exploratory data analysis (EDA) phase was carried out to uncover critical insights and understand the underlying patterns and distributions within the dataset. This comprehensive analysis was vital for interpreting complex behaviors in the data and identifying key variables that could influence our predictive models.

**Total Sales by Product Family:** This bar chart provides a clear visualization of total sales distributed across various product families. The disproportionate bars reflect the dominant sales of grocery items and beverages, highlighting the importance of these categories in the overall sales strategy and inventory management.

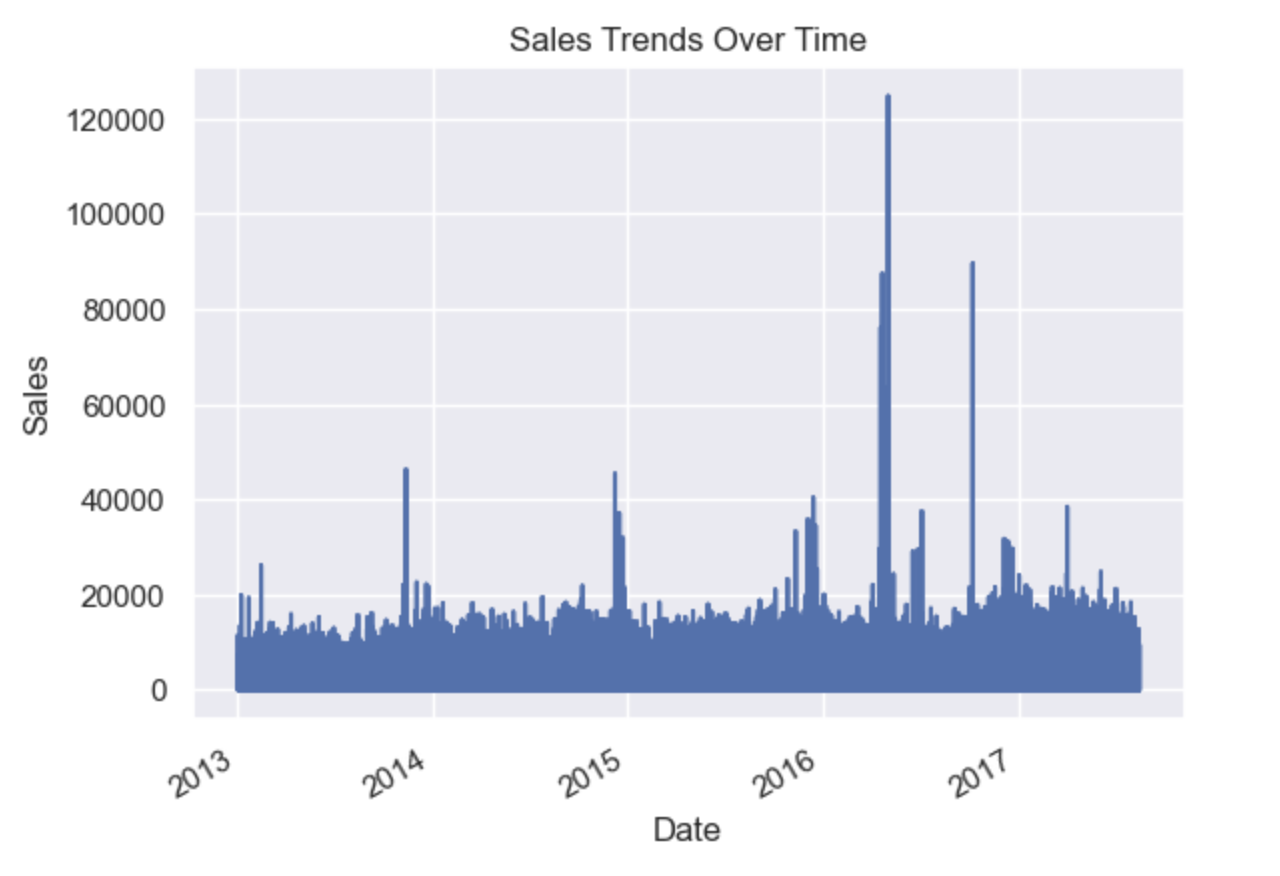


**Monthly Average Sales**: The line chart reveals a cyclical pattern in monthly average sales, indicating seasonality in consumer behavior. Peaks observed during certain months may correspond to holiday seasons or marketing campaigns, suggesting a need to adjust stock and staffing levels accordingly.

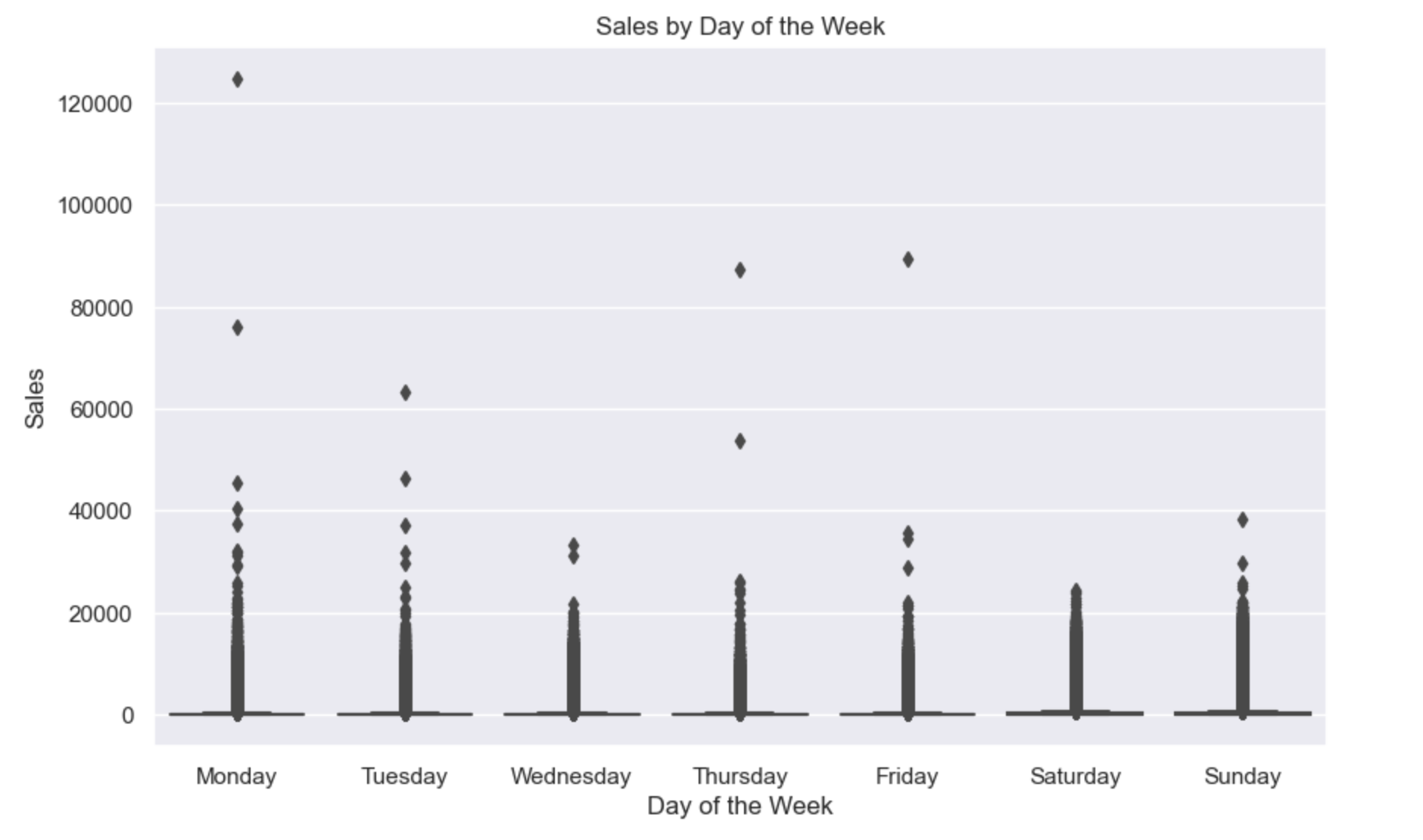
**Average Sales by Month and Day of the Week**: This heatmap shows the average sales figures across different months and weekdays. Higher sales on specific days, like weekends, and during certain months can guide promotional and stocking decisions. The pattern suggests a potential consumer purchasing rhythm tied to weekly and monthly cycles.



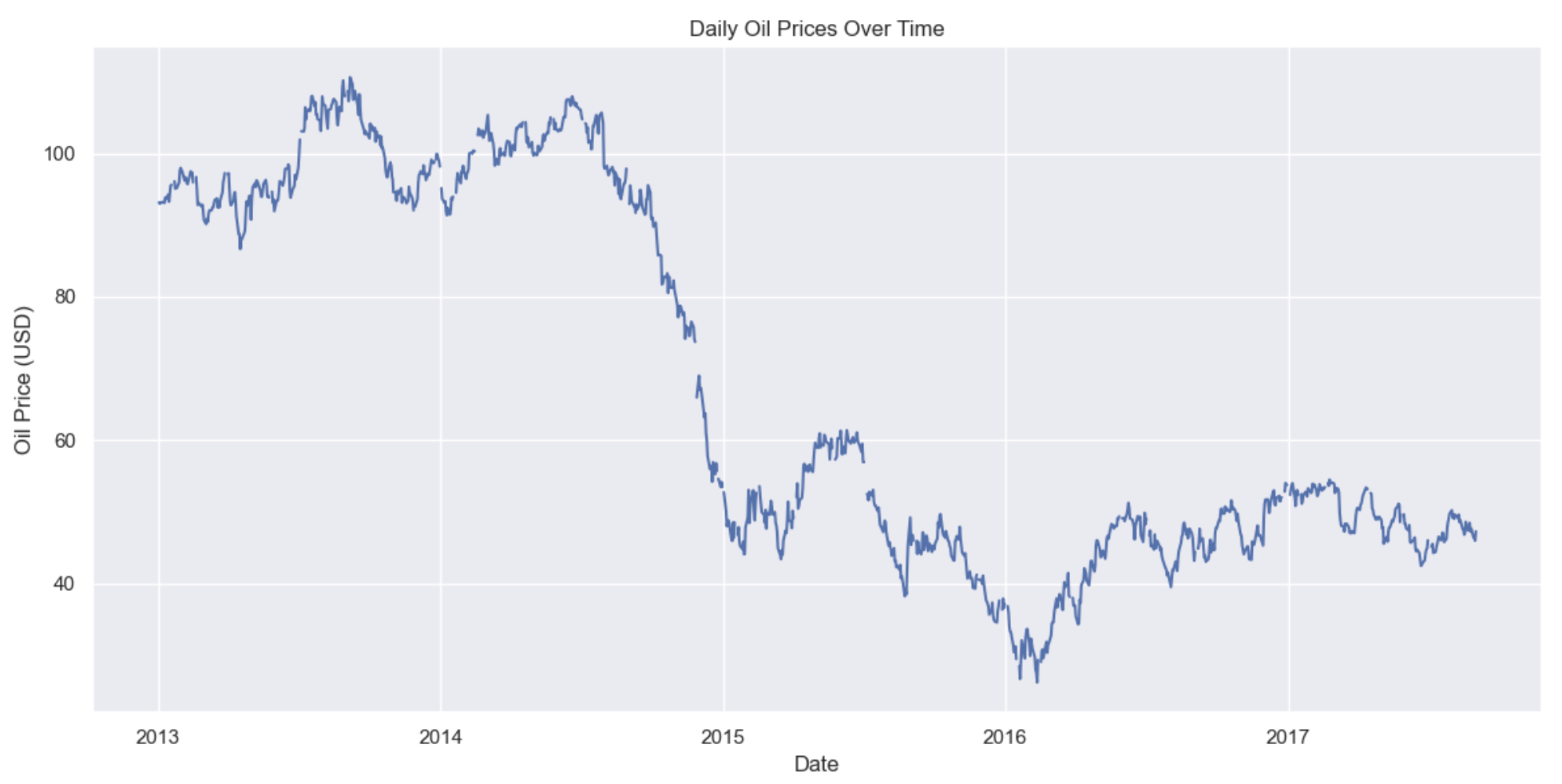
**Sales Trends Over Time:** The bar chart depicts overall sales trends over time, with notable spikes that may correspond to significant events or promotional activities. This visualization is crucial for understanding long-term trends and planning for future demand by recognizing patterns in sales volume.



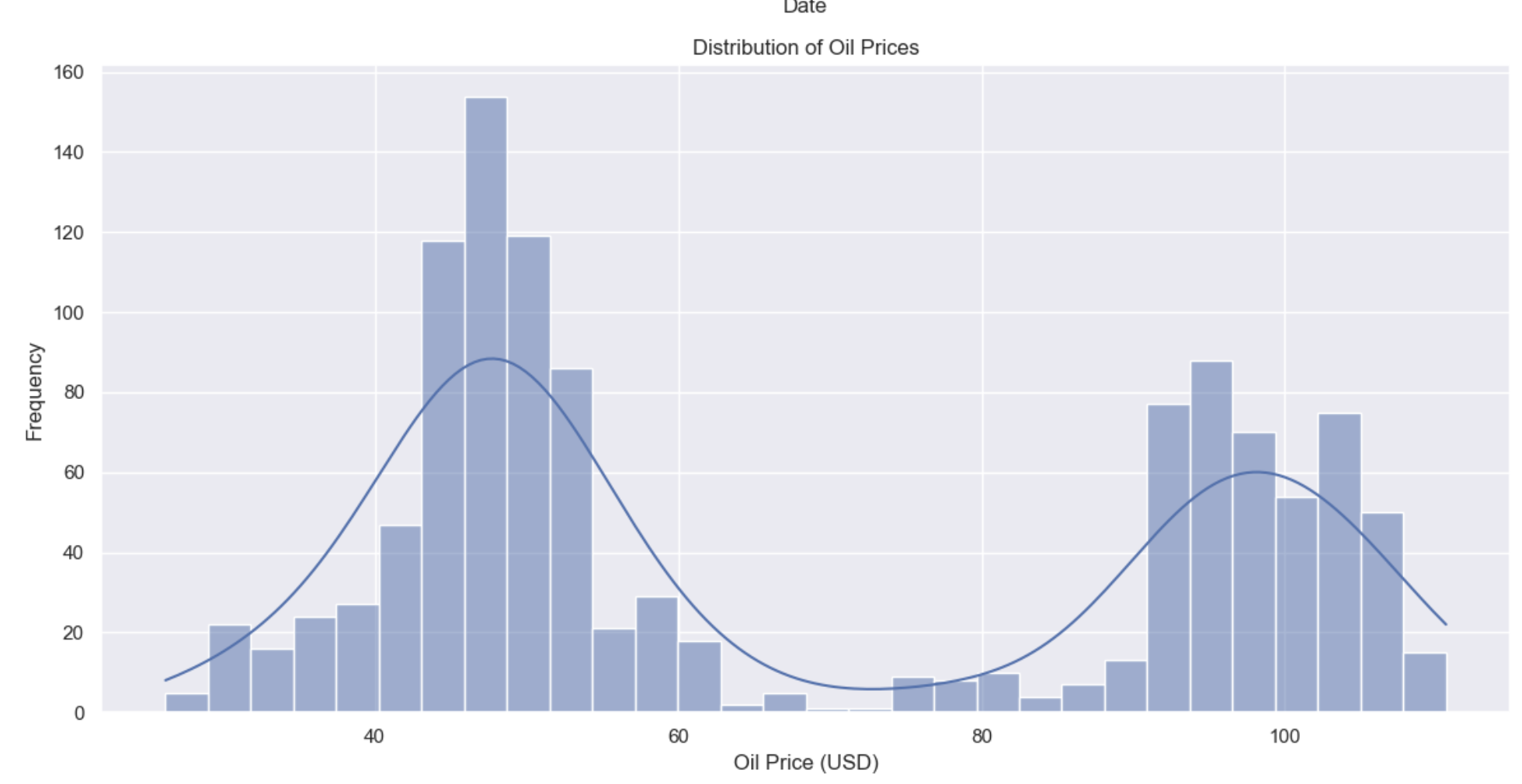
**Sales by Day of the Week:** This scatter plot portrays sales distribution across the days of the week, providing insights into which days have higher sales volumes. Such information could be pivotal for scheduling promotions and optimizing operations on days with historically higher traffic.

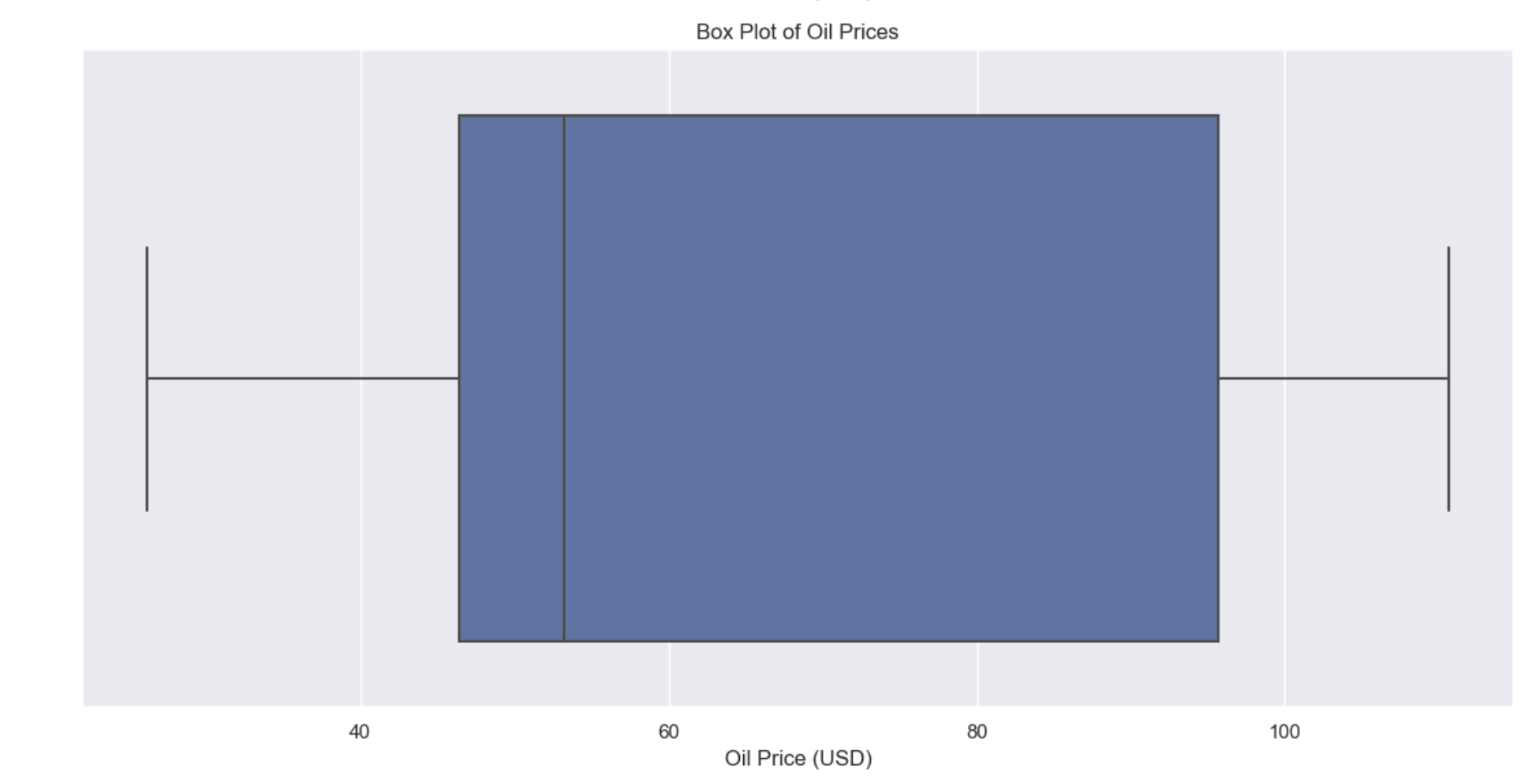


**Daily Oil Prices Over Time:** This line chart of daily oil prices over time could be used to examine the impact of external economic factors on sales. The fluctuation in oil prices might have implications for cost structures and consequently could influence pricing strategies.



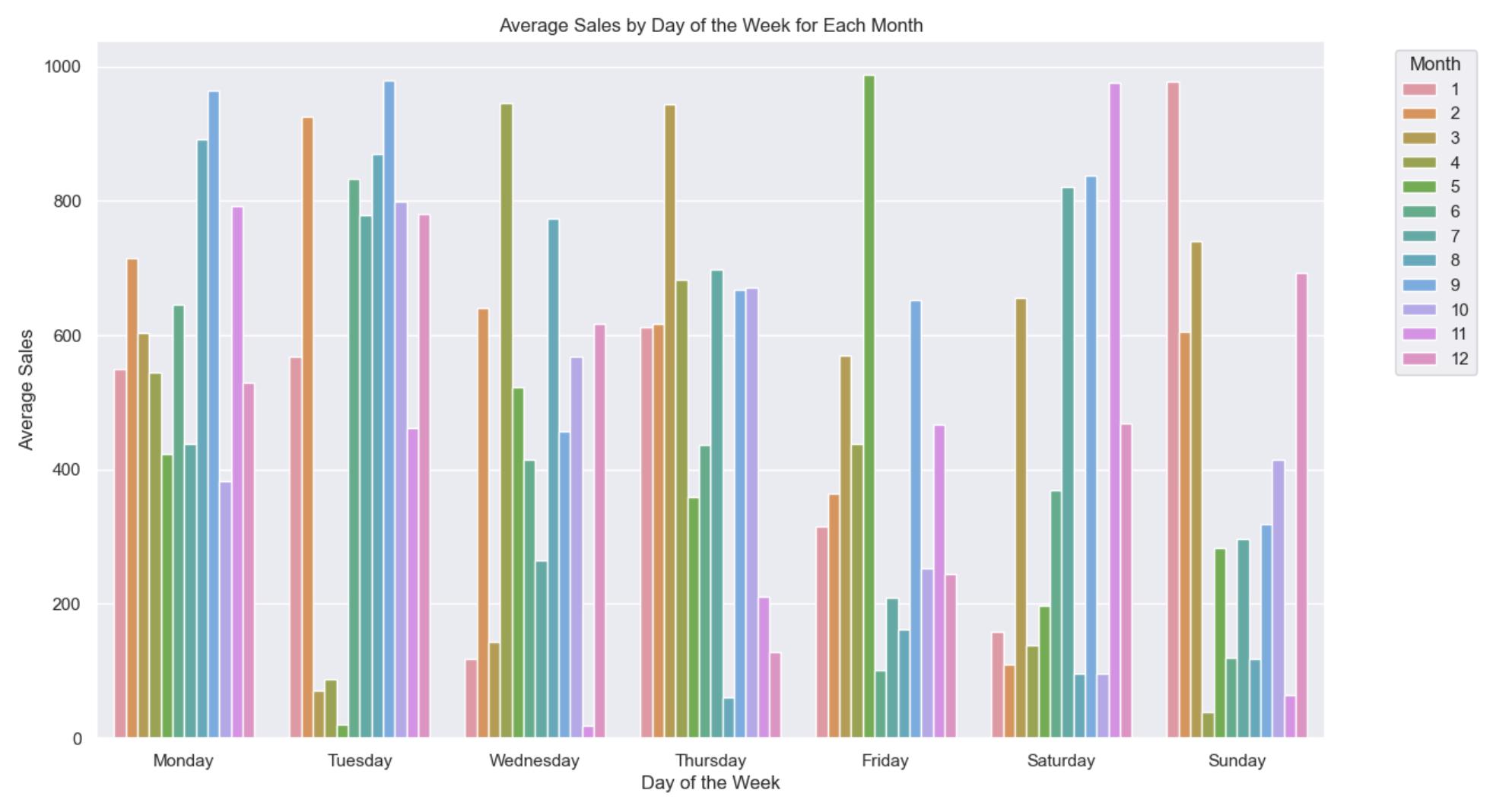
**Distribution of Oil Prices:** The histogram of oil prices, along with the overlaid density plot, illustrates the distribution of oil prices and helps in understanding the most common price ranges. Such information can inform risk management strategies, especially for businesses with oil-dependent cost structures.



**Box Plot of Oil Prices:** This box plot provides a summary of the distribution of oil prices, showcasing the median, quartiles, and outliers. This visualization helps in quickly grasping the central tendency and dispersion of oil prices, which can be crucial for economic forecasting and budgeting.

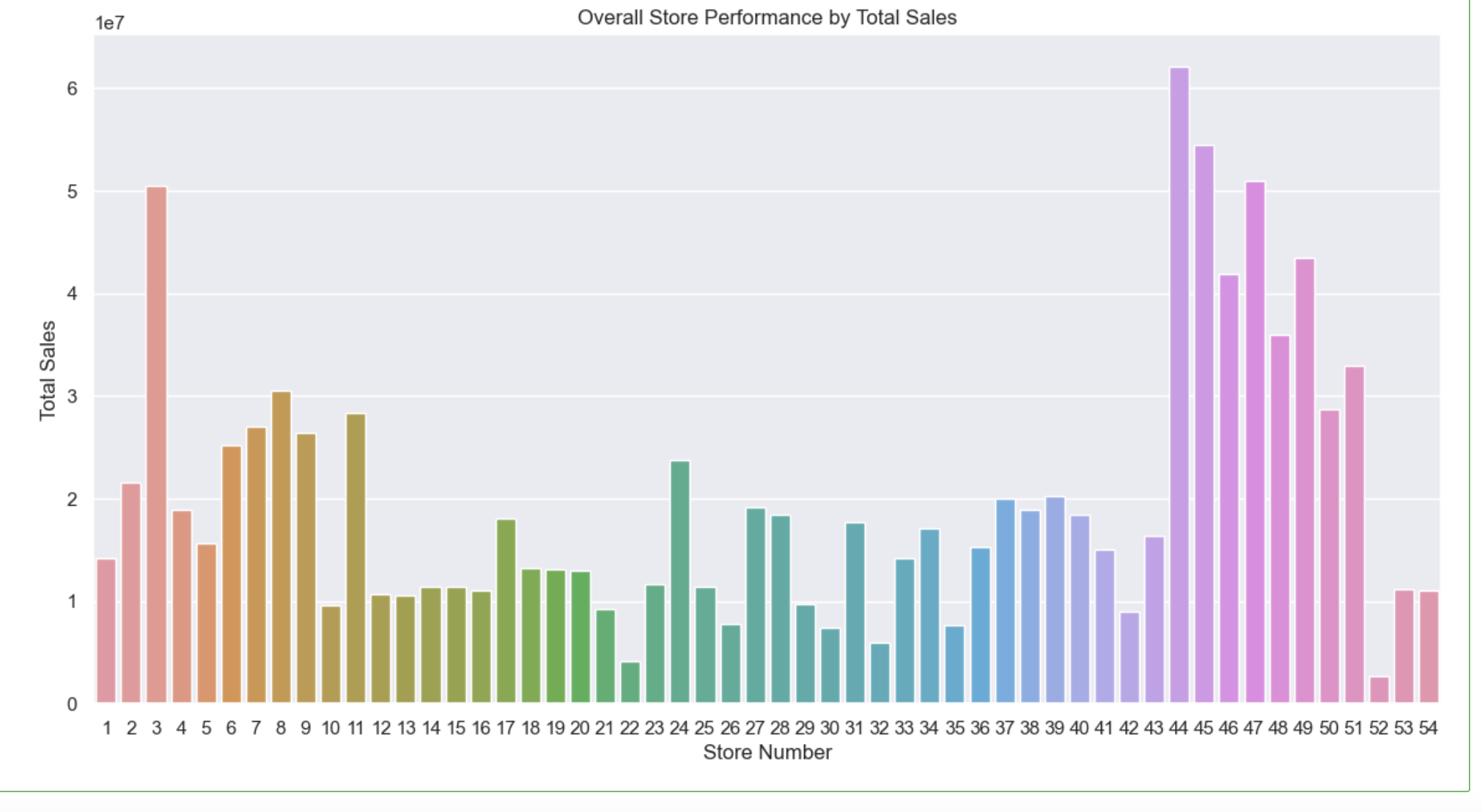
The bar plot "**Average Sales by Day of the Week for Each Month**" visualizes mock data on sales. Each bar represents the average sales for a particular day of the week across 12 different months. The days of the week are laid out on the x-axis from Monday through Sunday, while the y-axis represents the average sales value.

The bars are color-coded to represent each month, with a separate color for each month from January (1) to December (12), and a legend to the right of the plot correlates the colors to the months.



The bar plot "**Overall Store Performance by Total Sales**" presents the total sales for different stores, identified by their store numbers on the x-axis. Each bar represents the sum of sales for one store, with the y-axis showing the total sales in scientific notation (1e7 indicates a multiplier of 10 million). The bars are color-coded, seemingly randomly, providing a visual comparison of each store's total sales.

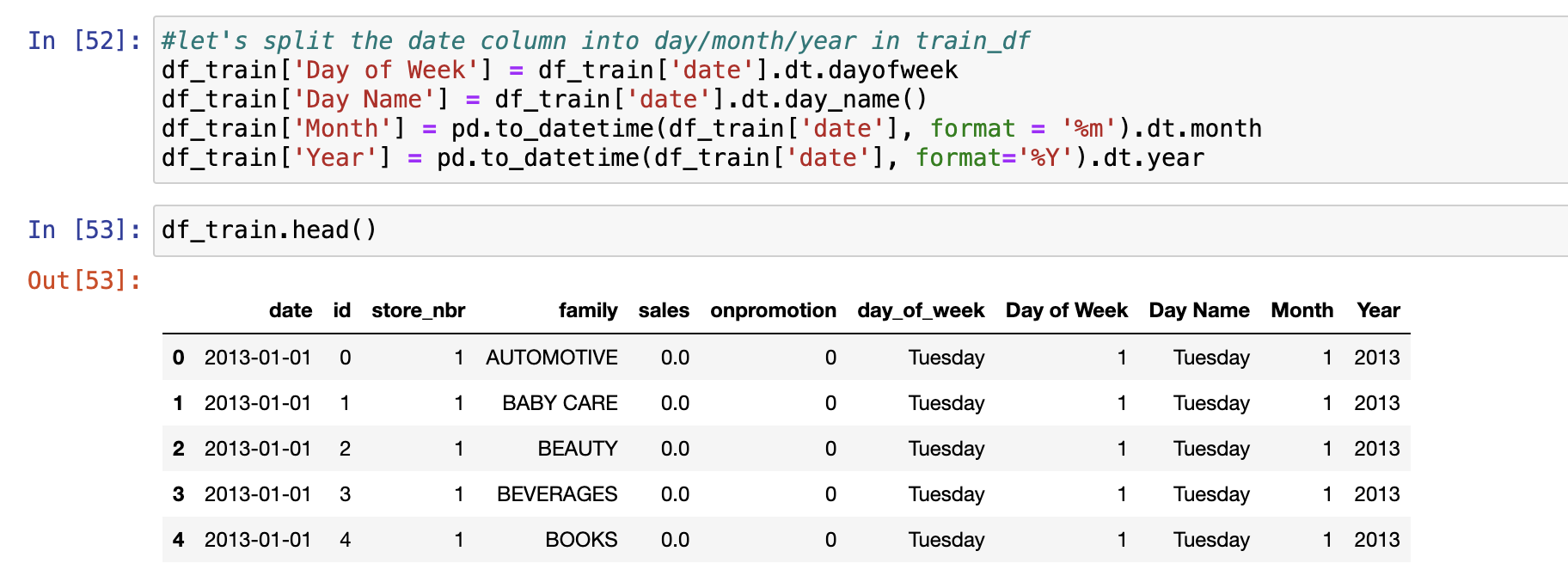
The heights of the bars reflect the sales volume, allowing for quick identification of which stores are outperforming or underperforming in terms of sales. For instance, the tallest bars signal the highest selling stores, while the shortest bars indicate the stores with the least sales. This visualization is valuable for analyzing the performance distribution across different stores and making strategic decisions based on sales data.

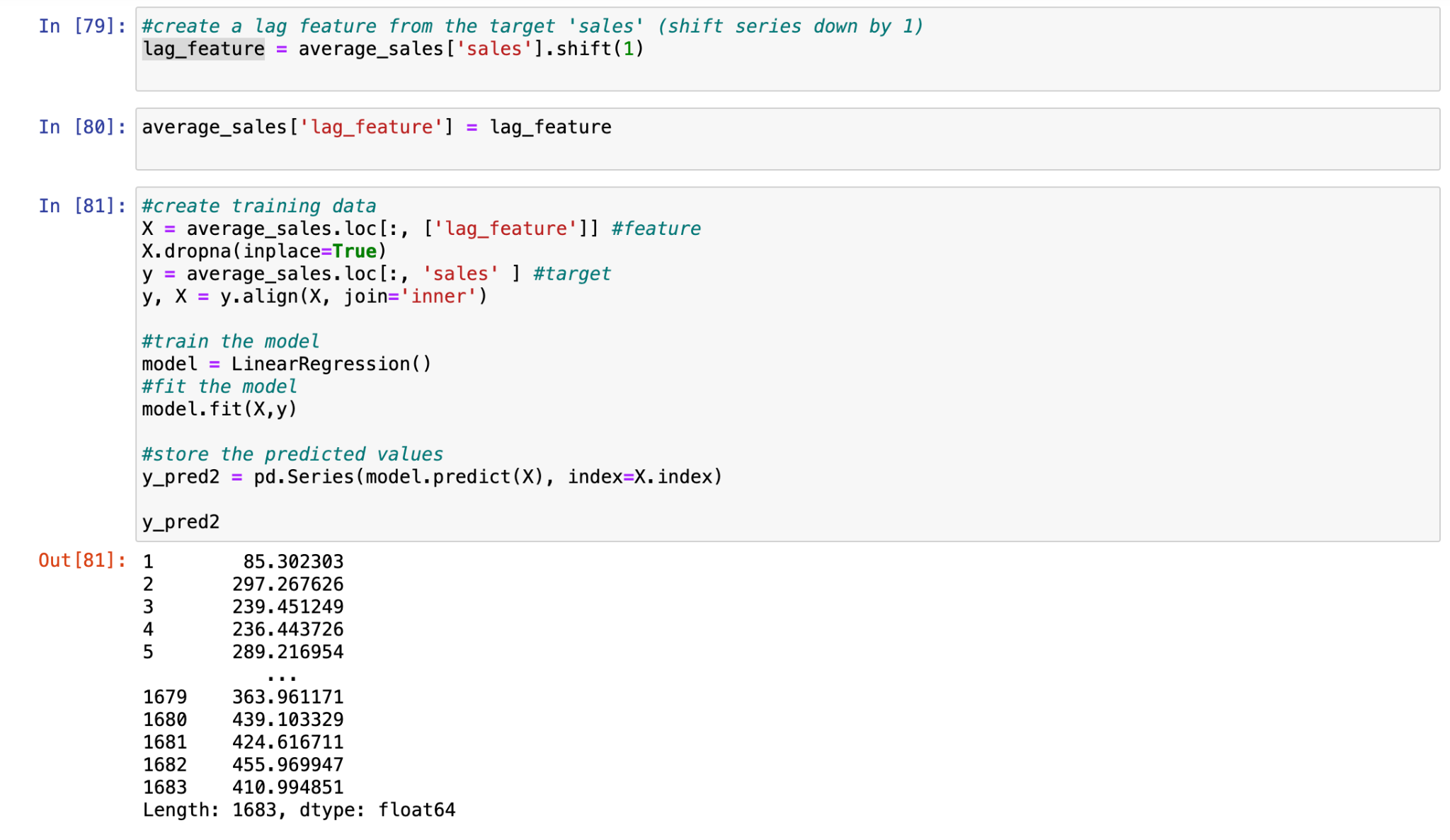


These visuals, taken together, offer a comprehensive overview of sales dynamics, the influence of external factors like oil prices, and consumer purchasing patterns, which are all vital for making informed business decisions.

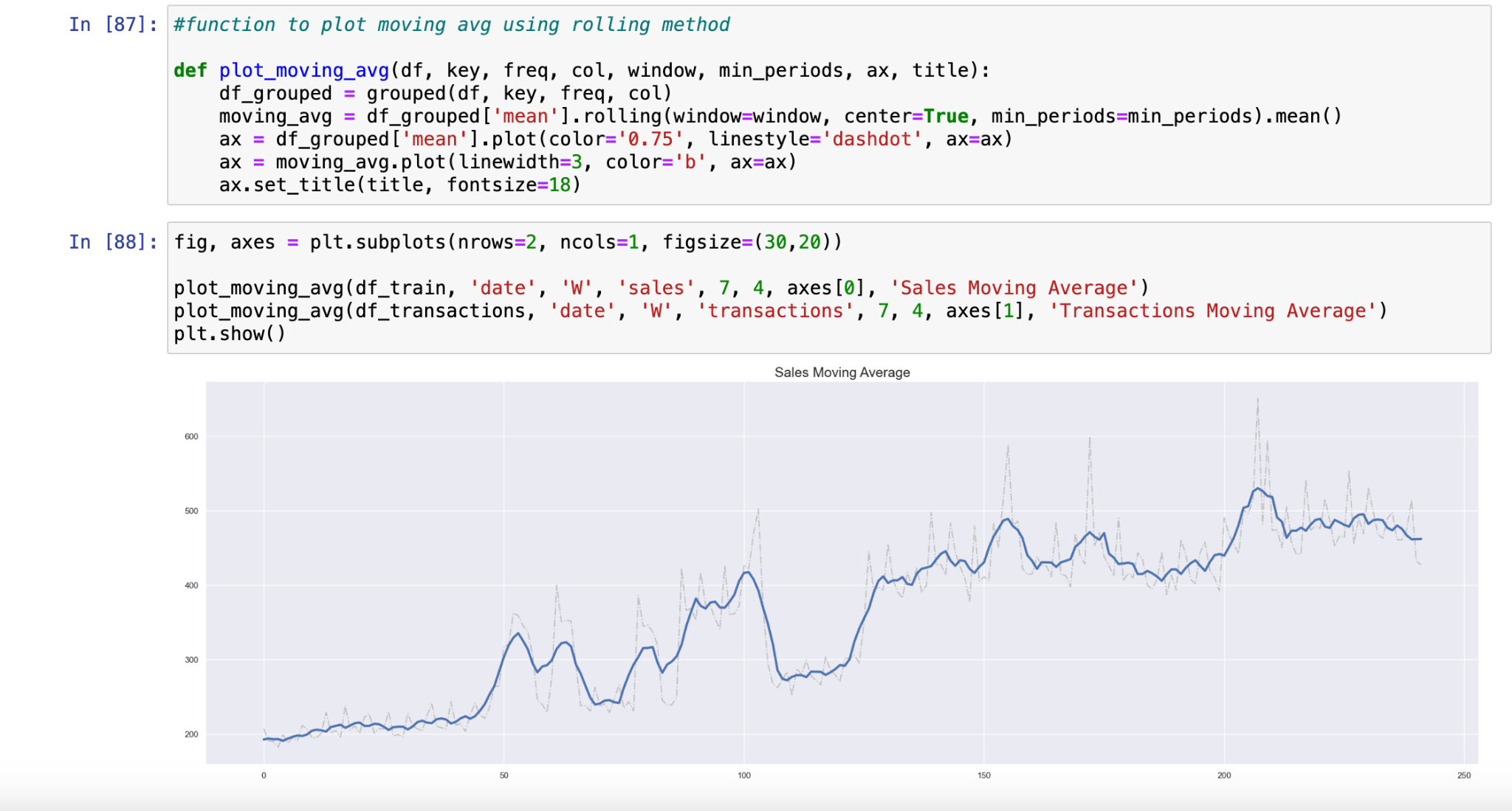
**Feature Engineering:**

Feature engineering was a critical step to improve the model's predictive accuracy by creating new features that captured the essence of the dataset more effectively.

Temporal features like 'Year', 'Month', and 'Weekday Name' were extracted to incorporate the time aspect into the model, acknowledging that sales can be significantly influenced by these factors due to trends, seasonality, and weekly cycles.

Lag features were introduced to provide the model with information on past sales values, which are often indicative of future trends. By shifting the sales data by one period, the model could utilize the previous period's sales as a predictor for the current period, under the premise that recent past performance can be a strong indicator of the near future.

Rolling averages were calculated to smooth out the sales data over time, helping to identify long-term trends by averaging out short-term fluctuations. This is particularly helpful in highlighting underlying patterns in the presence of noisy data or outliers.



The rationale behind these steps is grounded in the domain understanding that sales are not random but influenced by various temporal factors and past performance, which can be leveraged to make accurate future predictions. These features, therefore, play a crucial role in capturing the chronological dependencies within the data, which are expected to be repeated in future observations.

**Modeling:** Apply statistical and machine learning models to forecast sales. Selecting appropriate models like Linear regression, random forest, decision trees and XGBoost to capture the nature of sales data.

**Model Evaluation:** Assess the model's accuracy and reliability in forecasting sales using metrics like RMSE (Root Mean Square Error) or MAE (Mean Absolute Error). Compare different models to select the best performer. Analyze where the model's predictions deviate from actual sales to understand potential areas of improvement in feature engineering or model selection.

**Result and Analysis**

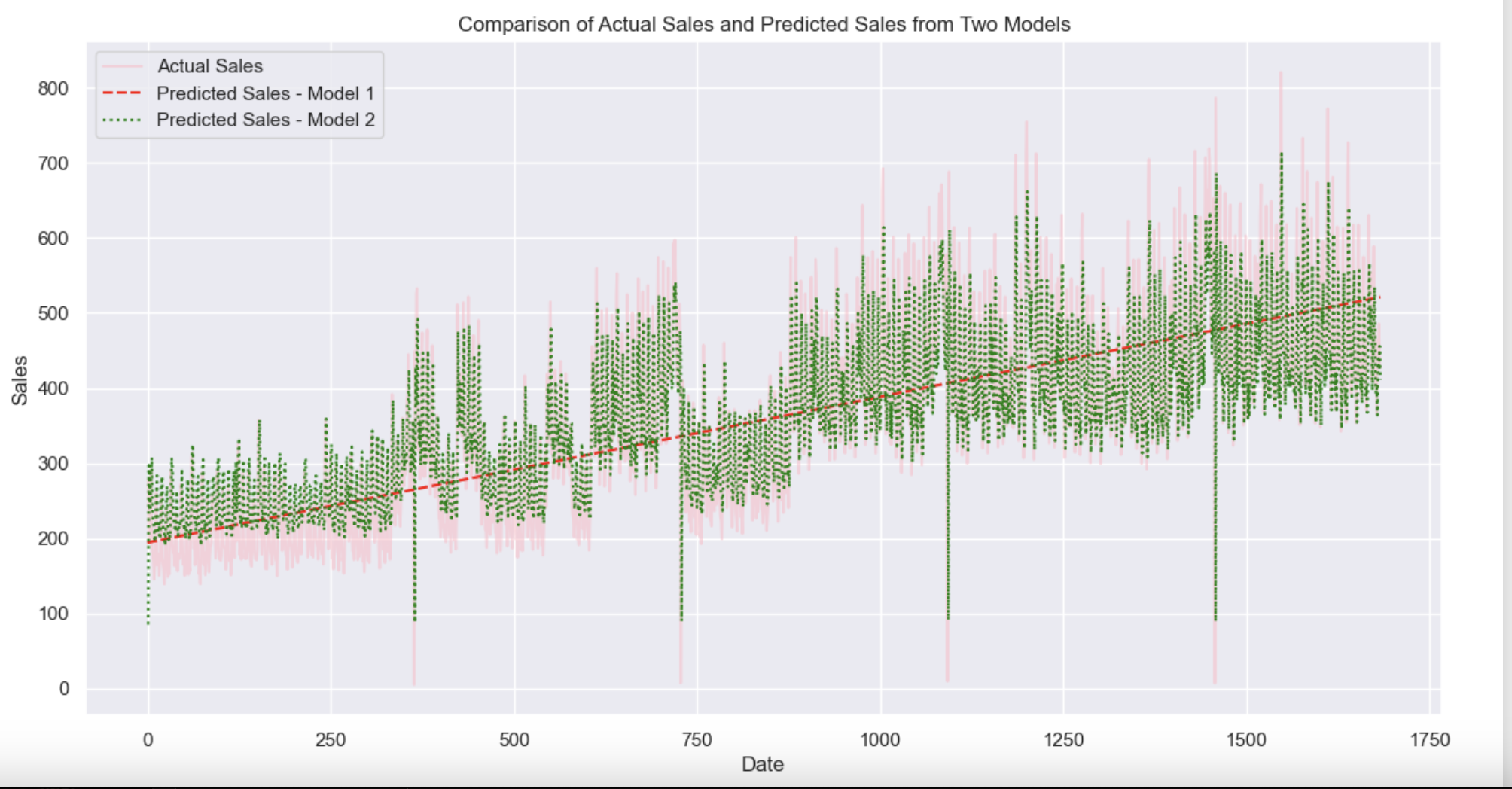
**Linear Regression with Time Series**

Linear Regression was applied to capture linear trends over time, treating the 'time' variable as a direct predictor for sales. This approach is based on the assumption that sales exhibit a linear progression as time advances. Additional features like lag variables were incorporated to utilize historical sales data as a basis for predicting future values, an essential aspect of time series forecasting due to the chronological nature of sales patterns.



The graph titled "**Time Plot of Total Store Sales**" displays two sets of time-series data: the actual sales (in grey) and the predicted sales (in blue), over a span from January to April 2021. Each point on the time series represents a daily sales figure, with the actual sales depicted as a lighter grey line with a dot marker for each day, and the predicted sales shown as a solid, darker blue line.

The fluctuations in the grey line show the variability in actual daily sales figures, ranging from as low as 100 to as high as 500. The blue line, representing the predicted sales, also varies each day but is plotted with a thicker line, making it visually distinct from the actual sales.



The above plot displays a comparison between actual sales and the sales predicted by two different models over time.

Actual sales are represented by a semi-transparent pink line, while the predictions from Model 1 and Model 2 are shown as a dashed red line and a dotted green line, respectively.

This visual comparison allows for the assessment of each model's accuracy, with the proximity of the predicted lines to the actual sales indicating the level of precision. Deviations between the predictions and actual sales provide insights into the models' performance, highlighting areas for potential improvement. The plot is a visual tool for evaluating and contrasting the forecasting capabilities of the two predictive models.

**Importance of Lag Features and Rolling Averages**

Lag features and rolling averages were critically engineered to capture temporal trends and smooth out volatile sales data. Lag features provide the model with access to previous sales points, vital for recognizing sales continuity or change, while rolling averages dampen the noise and reveal more stable trends within the fluctuating sales figures.

**Ridge regression** was employed to address multicollinearity among features and prevent overfitting, due to its regularization capabilities. The model's alpha parameter, which governs the regularization strength, was optimized using GridSearchCV to ensure the most effective shrinkage of the coefficients, enhancing the model's prediction accuracy on unseen data. This approach was particularly beneficial given the large number of features generated from the extensive feature engineering process. The entire process is encapsulated within a pipeline for standardized scaling and efficient hyperparameter tuning.

The purpose of this code is to optimize the Ridge regression model to achieve the lowest mean squared error (MSE: 729768.5559850216), thereby enhancing the accuracy of the model's predictions. The use of Pipeline and GridSearchCV helps streamline the process of scaling, parameter tuning, and cross-validation to efficiently find the best model configuration.

**XGBoost**

XGBoost was employed for its exceptional ability to model complex, nonlinear relationships inherent in the data. The optimized parameters for XGBoost, determined through a rigorous hyperparameter tuning process, included a colsample\_bytree of 0.9, a learning\_rate of 0.2, a max\_depth of 5, and n\_estimators of 300, with a subsample rate of 0.9. These settings were instrumental in enhancing the model's accuracy and its ability to generalize beyond the training dataset.

Before Hyperparameter tuning:

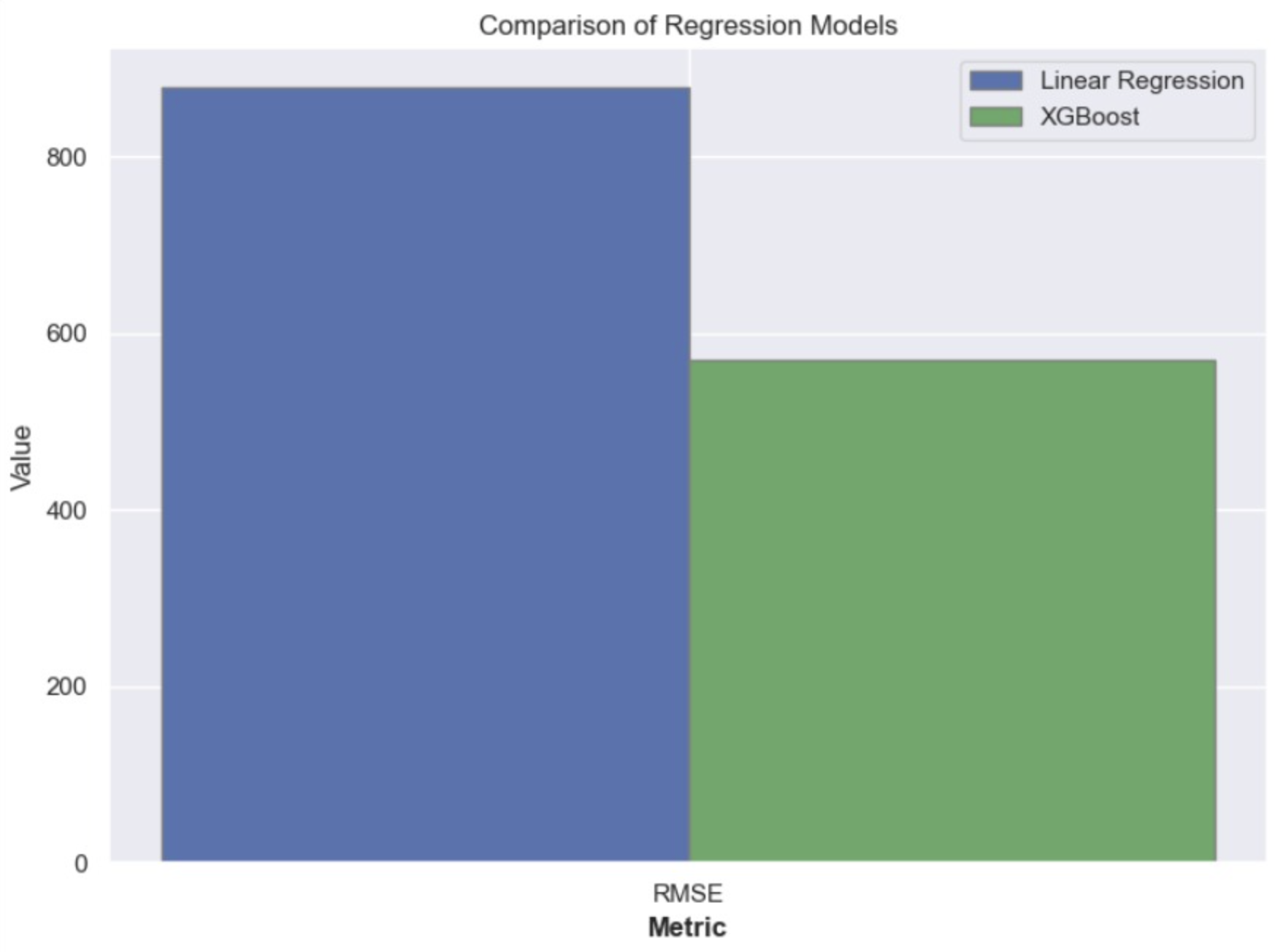
RMSE: 569.158529113529

R2: 0.6669726817185002

After Hyperparameter Tuning:

RMSE: 546.5577987964145

Comparing XGBOOST with LINEAR REGRESSION after Hyperparameter Tuning



**Random Forest**

The Random Forest model, an ensemble of decision trees, was utilized for its robust performance in high-dimensional spaces. Its structure is ideal for capturing complex interactions between variables. The model was fine-tuned using cross-validation techniques to identify optimal hyperparameters, ensuring a balance between the model's bias and variance, thereby enhancing its predictive performance on unseen data.

RMSE: 596.6240420054577

For hyperparameter tuning, we used 100k rows due to computational limitations, as the system was unable to handle the large dataset.

RMSE: 555.1503312622295

**Decision Trees**

Decision Trees served as a straightforward, interpretable model for sales forecasting. The best parameters for the Decision Trees were established through GridSearchCV, which conducted exhaustive searches across a predefined parameter grid to optimize model performance. This process ensured that the model was neither overfitted nor underfitted, providing reliable and understandable predictions.

The selection and tuning of these models reflect a comprehensive analytical strategy tailored to forecast sales with high precision. The models' performance was assessed using metrics like RMSE (Root Mean Square Error), which quantifies the difference between predicted and actual values, offering a clear benchmark for model comparison and selection. The use of these diverse models enabled a robust analysis, catering to the intricate dynamics observed in the sales data, thus paving the way for accurate and actionable sales forecasting.

RMSE: 690.7139261139154

For hyperparameter tuning, we used 100k rows due to computational limitations, as the system was unable to handle the large dataset.

RMSE: 586.6535011376407

**Cross Validation**

The cross-validation of machine learning models was conducted, and the following RMSE scores were obtained:

**Linear Regression Model**

RMSE values were computed for the Linear Regression model using 10-fold cross-validation. The performance of the model was assessed by the root mean squared error metric.

(RMSE mean: 854.13)

**XGBoost Model**

Similar to Linear Regression, the XGBoost model underwent 10-fold cross-validation, and RMSE values were generated to evaluate its predictive accuracy. (RMSE mean: 538.97)

**Random Forest Model**

Due to memory errors when using the full dataset, only 10% of the data was utilized for cross-validation of the Random Forest model. This approach was chosen to avoid computational errors and still obtain an estimate of the model's performance.(RMSE mean: 609.73)

**Decision Tree Model**

The Decision Tree model was also validated using a subset of the data to prevent errors. Cross-validation provided insight into the model's predictive power with a focus on its RMSE score. (RMSE mean: 761.29)

**A graph of different models

Description automatically generated**

**Conclusion**

Based on the RMSE scores obtained from the cross-validation process, we can conclude that the XGBoost model outperforms the other three models with the lowest RMSE mean of **538.97**.

This indicates that XGBoost is the most accurate in terms of prediction among the tested models for this dataset. The Random Forest model also shows strong performance, especially considering that only 10% of the data was used for cross-validation due to memory constraints, achieving an RMSE mean of **609.73.**

The Decision Tree model, with an RMSE mean of 761.29, does not perform as well as XGBoost or Random Forest but is still preferable over the Linear Regression model, which has the highest RMSE mean of 854.13.

The results suggest that ensemble methods like XGBoost and Random Forest, which combine multiple models to make decisions, are more effective for this dataset compared to a single decision tree or a linear approach.

**References**

[1] Store Sales - Time Series Forecasting. Available at : [https://www.kaggle.com/competitions/store-sales-time-series-forecasting/overview]

[2] Interpolation in Python [https://www.geeksforgeeks.org/interpolation-in-python]

[3] Scikit-learn Documentation. Available at: [https://scikit-learn.org/stable/documentation.html]

[4] "XGBoost Documentation". Available at: [https://xgboost.readthedocs.io/en/latest/]

[5] Decision Tree Algorithm [https://scikit-learn.org/stable/modules/tree.html]

[6] Hyperparameter tuning for ML models. Available at: [https://towardsdatascience.com/hyperparameter-tuning-for-machine-learning-models-1b80d783b946]