# Introduction

In this project, we aimed to build robust machine learning models to forecast sales data. Accurate sales forecasting is crucial for optimizing inventory levels, reducing costs, and improving customer satisfaction. We used historical sales data to train and evaluate several regression models, assessing their performance and identifying the most effective model for our forecasting needs.

Additionally, the project is integrated into a web application where user can input the number of days he wants to forecast. As an initial demonstration, we performed a forecast for one month. This feature allows users to make customized forecasts based on their specific needs and helps in making data-driven decisions for their business operations.

# The Dataset

The dataset used in this project consists of daily sales records for a store over a period of five years. It contains 1,826 entries, with each record including a date and the corresponding sales figure. The dataset was loaded into a Pandas DataFrame for analysis and modeling.

**Dataset Information:**

* **Number of Entries:** 1,826
* **Columns:**
  + date: The date of the sales record.
  + sales: The sales amount on the given date.

**Sample of the Dataset:**

|  |  |
| --- | --- |
| date | sales |
| 2013-01-01 | 1316 |
| 2013-01-02 | 1264 |
| 2013-01-03 | 1305 |
| 2013-01-04 | 1452 |
| 2013-01-05 | 1499 |

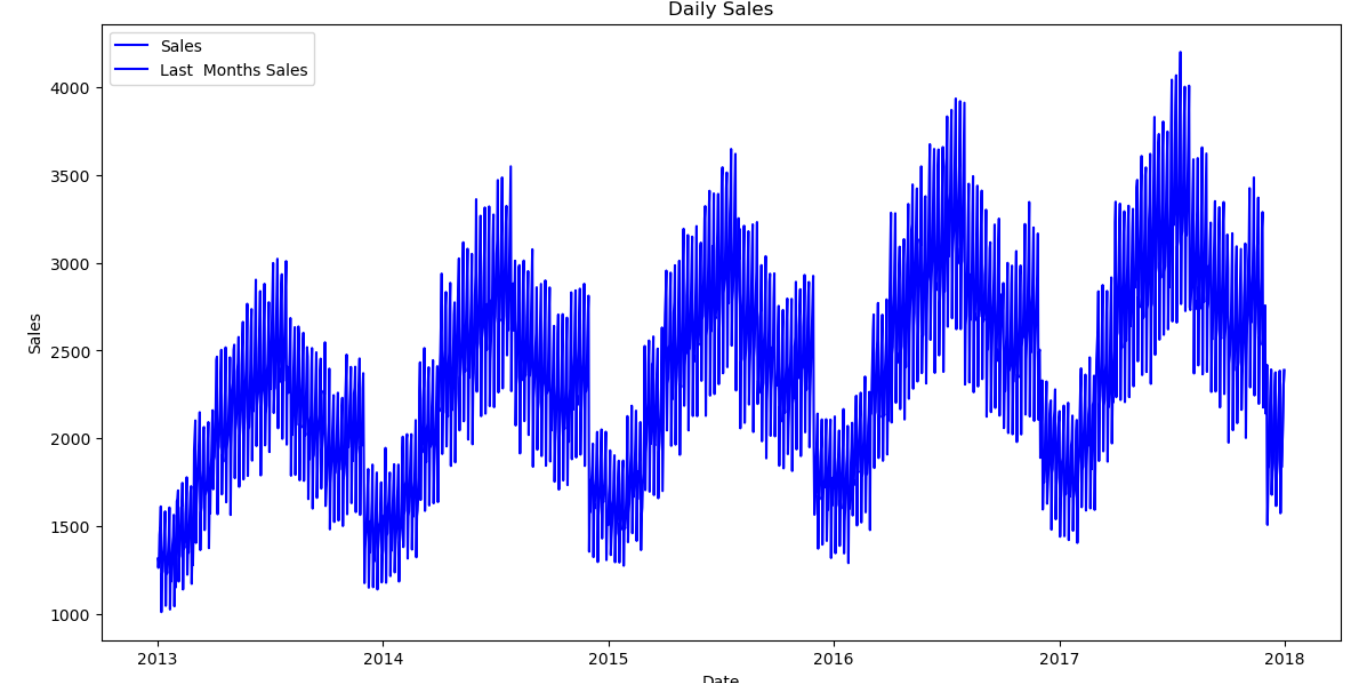
# Methodology

Our methodology involved several key steps: data loading, cleaning, feature engineering, data visualization, model training, hyperparameter tuning, model evaluation, and feature importance analysis. We documented our process and results using Jupyter Notebook.

# Dataset Cleaning

We began by loading the sales data and performing necessary cleaning steps. This included converting the date column to a datetime object and aggregating sales by date. We also identified and handled any missing or inconsistent data.

# Dataset Analysis and Visualization

We created various plots to visualize sales trends and the impact of different time-based features on sales.

# Data Preprocessing and Preparation

Feature engineering was performed to add new time-based features such as the day of the week, month, quarter, and year. These features helped the models capture temporal patterns in the data.

The dataset was then split into training and test sets. The training set was used to train the models, while the test set was used to evaluate their performance. Specifically, the last month of the dataset was reserved for testing, ensuring that the models were evaluated on recent data that was not seen during training.

This approach not only provided a realistic assessment of the models' performance but also allowed us to verify the models' ability to generalize to new, unseen data. Additionally, the duration of the test set can be adjusted based on specific requirements, allowing for flexible evaluation periods.

# Machine Learning Regression Models

We trained multiple regression models, including Linear Regression, Polynomial Regression, and XGBoost, and evaluated their performance using metrics such as RMSE, MAE, MAPE, and R².

## Linear Regression

The Linear Regression model performed as follows:  
- Train RMSE: 414.82  
- Test RMSE: 938.44  
- Test MAE: 936.39  
- Test MAPE: 46.72%  
- Test R²: -12.15

## Polynomial Regression

The Polynomial Regression model performed as follows:  
- Train RMSE: 162.40  
- Test RMSE: 246.52  
- Test MAE: 217.73  
- Test MAPE: 10.27%  
- Test R²: 0.09

## Hyperparameter Tuning

For the XGBoost model, we performed hyperparameter tuning to optimize the model's performance. Hyperparameter tuning is a crucial step in machine learning that involves selecting the best combination of parameters to improve the model's accuracy and efficiency.

We used GridSearchCV with time-series split cross-validation to search for the best combination of hyperparameters. The hyperparameters we tuned included:

* **max\_depth**: The maximum depth of the tree.
* **learning\_rate**: The learning rate (step size shrinkage).
* **n\_estimators**: The number of boosting rounds.
* **subsample**: The fraction of samples used for fitting the individual base learners.

This method allowed us to systematically evaluate the performance of the model with different parameter settings and identify the optimal values. As a result, the tuned XGBoost model achieved significantly better performance compared to the initial settings.

After tuning, we found the best hyperparameters for our XGBoost model, which were:

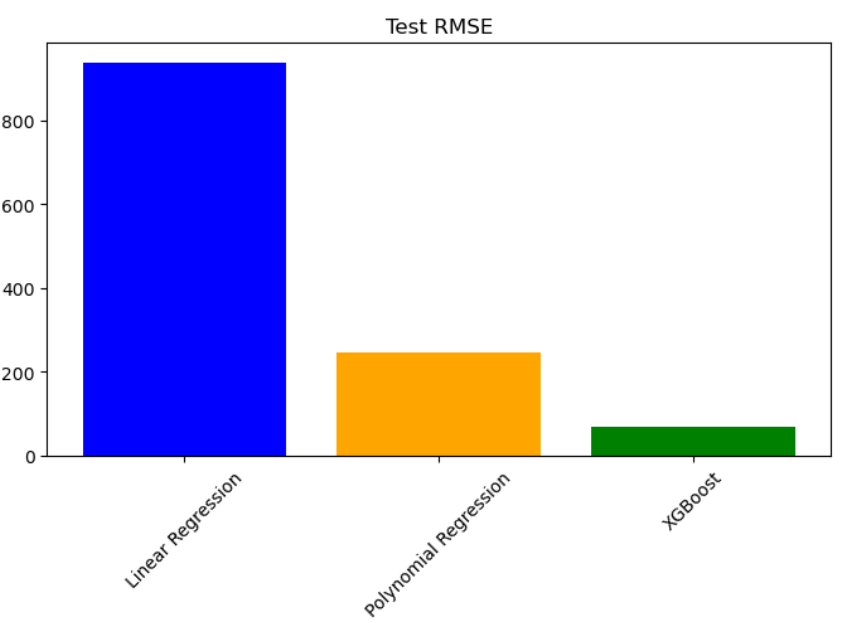
* **max\_depth**: 4
* **learning\_rate**: 0.02
* **n\_estimators**: 1000
* **subsample**: 0.8

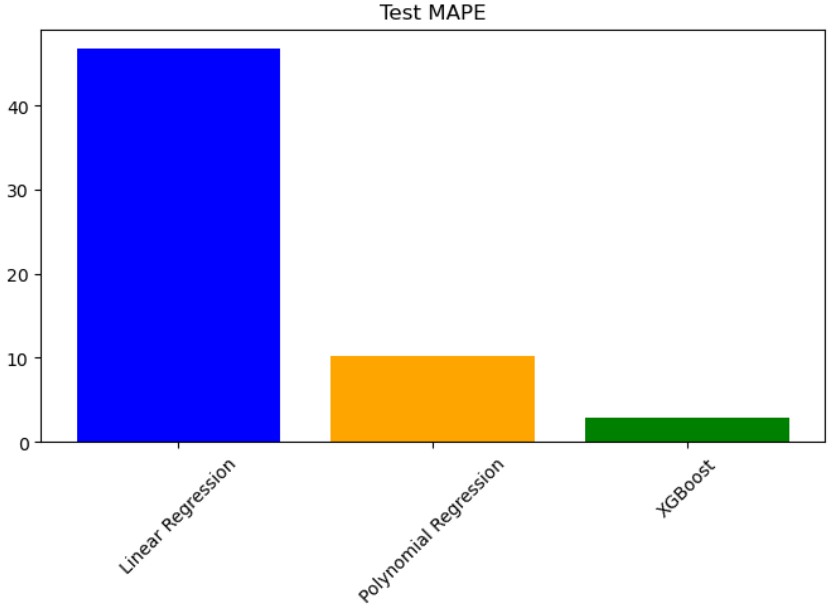
These optimized parameters were used to train the final model, leading to more accurate sales forecasts.

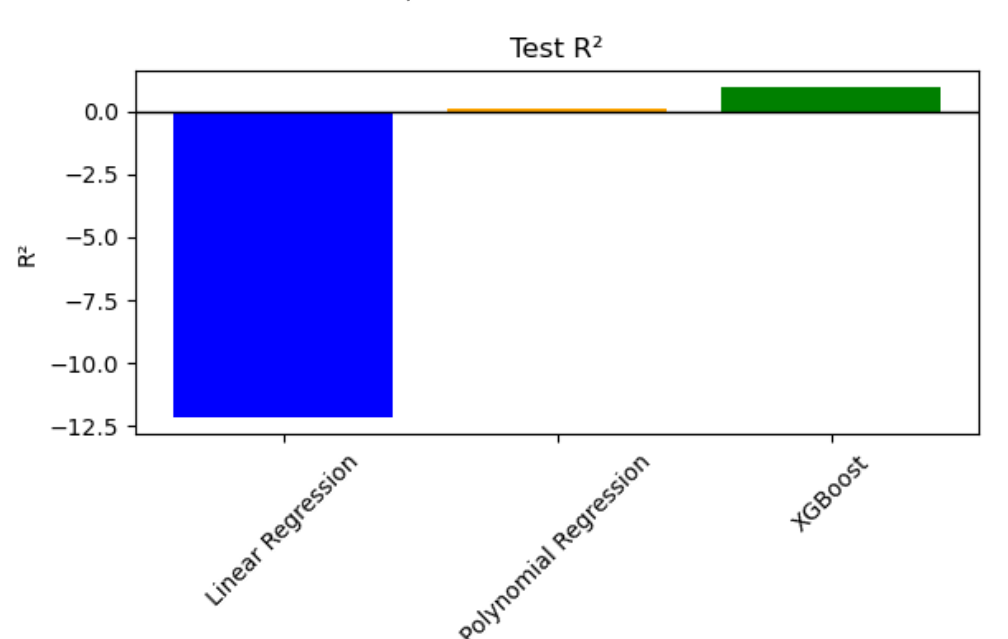
## XGBoost

The XGBoost model performed as follows:  
- Test RMSE: 68.06  
- Test MAE: 55.14  
- Test MAPE: 2.79%  
- Test R²: 0.93

# Models Evaluation

We evaluated the performance of each model using several metrics. The XGBoost model outperformed the others, achieving the lowest RMSE and MAE, and the highest R² score.





# Conclusion

In conclusion, our analysis and modeling efforts revealed that the XGBoost model is the most effective for forecasting sales in this dataset. Its superior performance metrics suggest that it can offer valuable insights for inventory management and strategic planning. Future work could focus on further refining the models and exploring additional features to enhance prediction accuracy.