BOOSTORDER X UPM

Coding Challenge

INTRODUCTION

Challenge

- Predict sales for third year based on previous two years' data

Development Platform

- Google Colab for collaboration and efficiency
- Support for Prophet, scikit-learn,Pandas and many more

Data

- Utilized organizer-provided dataset for robustness.
- Evaluated model performance across different data inputs.

Goal

- Create a robust, reproducible model for accurate sales forecasting.
- Ensure transparency and ease of follow-up.

Key Challenges and Solutions

Key Challenges	Solution
Complex Patterns Identifying and capturing seasonal effects, holidays, market shifts, and other subtle factors influencing sales trends.	• Used Prophet model, which inherently handles seasonality and holidays. • Implemented feature engineering in create_features method, including sine and cosine transformations for month and quarter to capture cyclical patterns. • Utilized XGBoost and Gradient Boosting models, which can capture complex non-linear patterns. • Created lag features and rolling statistics to capture time-dependent patterns.
Data Limitations Working with only 2 years of training data and 1 year for testing, ensuring model generalization.	• Employed ensemble method combining Prophet, XGBoost, and Gradient Boosting to leverage strengths of different models. • Used cross-validation in model optimization (implicit in Optuna's optimization process). • Implemented feature engineering to extract maximum information from limited data.

Key Challenges and Solutions

Key Challenges	Solution
Significant Negative Values Handling and interpreting significant negative values in the sales data.	• Filtered out negative amounts at the beginning of preprocessing: self.df = self.df[self.df['Amount'] >= 0] • Implemented outlier handling using IQR method to clip extreme values. • Added a warning system to alert if negative values persist after aggregation.
Limited Variables Lack of internal variables, necessity to incorporate external economic indicators.	• Integrated Malaysian economic indicators (leading, coincident, lagging indices) from external data source. • Added these indicators as features in all models: For Prophet: model.add_regressor(column)

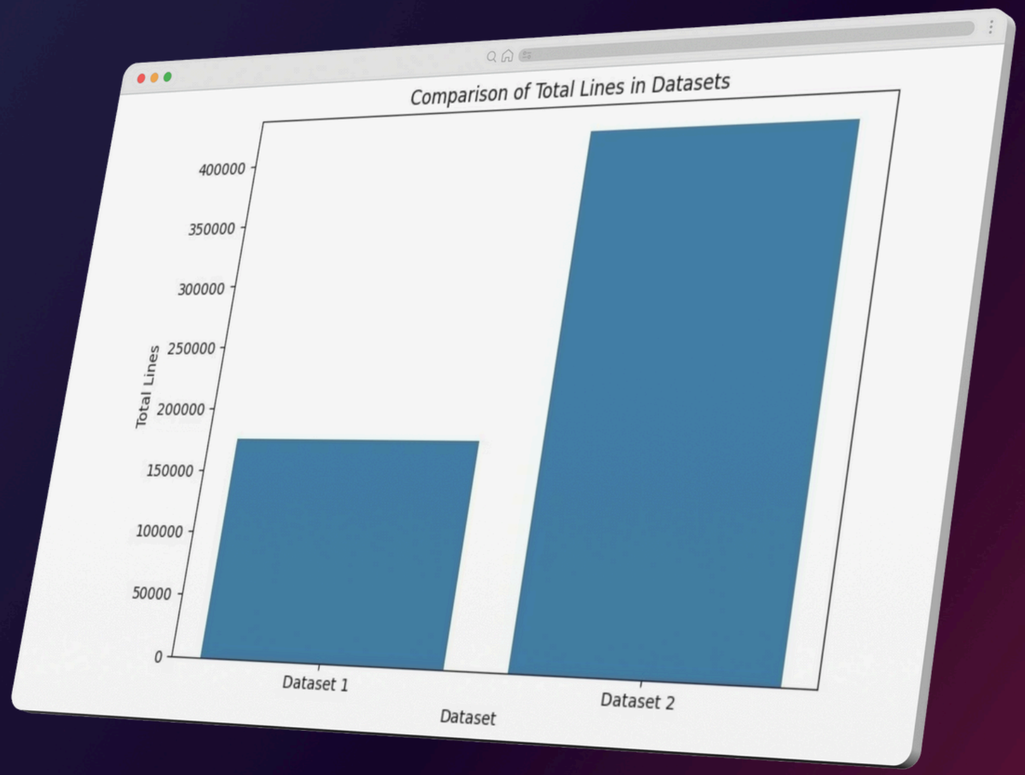
Objectives

Create a machine learning model that accurately forecasts third-year sales.

Evaluate the model's performance based on its predictive capability and real-world applicability.

Incorporate relevant factors like seasonality and trends for reliable predictions.

Data Overview



	Dataset 1	Dataset 2
Total rows	175,514	417,319
Total columns	6	6
Memory usage	8.0 MB	19.1MB
Missing values	384	None
Negative values	482	1

Data Preprocessing Steps

1

Date Standardization

- Removed unnecessary time information from "Date" column.
- Converted "Date" to a standardized datetime format for time-series analysis.

2

Column Management

- Removed redundant original "Date" column.
- Reordered columns to place "DATE" (datetime format) as the first column for clarity.

3

Handling Missing Values

- Analyzed missing values in Dataset 1 (affected rows, columns, nature).
- Removed rows with missing values in key columns ("ProductId", "Quantity", "Amount") to ensure data integrity for analysis.

4

Handling Negative Values

- Identified and addressed presence of extreme negative values in sales data (especially Dataset 1).
- Explained the rationale for removing all negative values from both datasets (data integrity, statistical distortion, forecasting accuracy, business logic, standardization, simplicity).
- Highlighted the impact (number of rows removed) and justification for this approach.

Model Development - Model Selection

- An ensemble of five models: Prophet, XGBoost, Gradient Boosting,RandomForest and LightGBM.
- Prophet is a statistical forecasting model suitable for time series data
- XGBoost and Gradient Boosting are machine learning algorithms known for their predictive power.
- The ensemble approach combines the strengths of these models to enhance accuracy and robustness.
- Accuracy Variance: The model's accuracy can vary across runs due to the stochastic nature of optimization and training. For instance, Dataset 1 might yield 87% accuracy in one run and 89% in another, but the accuracy is generally consistent and does not fluctuate significantly
- Contributing Factors: Variance is influenced by randomized hyperparameter optimization, model initialization, train-test splits, data preprocessing, and the ensemble method.

Benefits

- Improved accuracy by combining the strengths of different models.
- Reduced overfitting by mitigating the reliance on a single model.
- Increased robustness to variations in data and trends.

Ensemble Method

The ensemble_predictions function selects the prediction closest to the actual value for each time point, leveraging the best model at each time step.

Model Development - Model Training

Data Preprocessing

- Filtered negative sales amounts.
- Converted the date to datetime format.
- Aggregated sales data by month.
- Handled outliers using Interquartile Range (IQR).
- Merged with economic data and filled missing values.

Feature Engineering

Created additional features (temporal, cyclical, lag, rolling statistics).

Ensemble

Combined predictions from the five models using the ensemble_predictions function.

Model Training

- Trained Prophet, XGBoost, and Gradient Boosting models using the preprocessed data.
- Incorporated economic indicators as regressors.
- Performed hyperparameter tuning using Optuna.

Model Development - Model Evaluation

Metrics

- Mean Absolute Percentage Error (MAPE): Measures prediction accuracy as a percentage.
- Root Mean Square Error (RMSE): Provides an absolute measure of the prediction error.

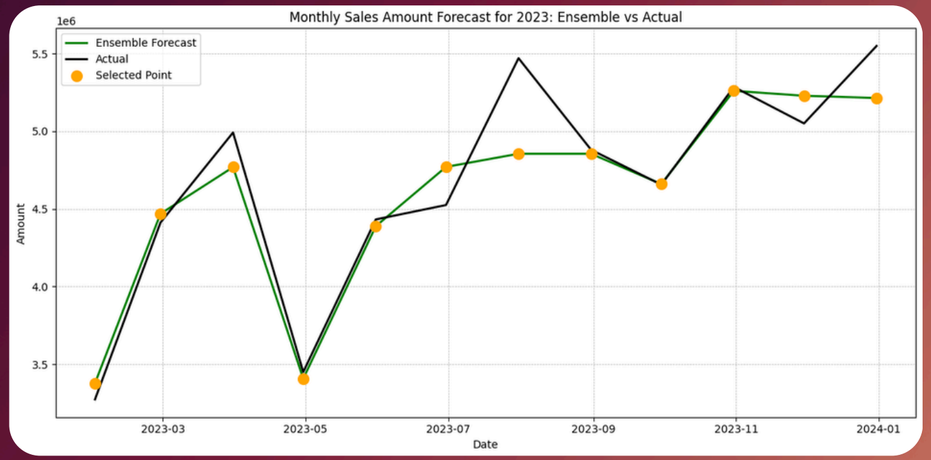
Error Analysis

- Analyzed specific predictions with significant errors to identify potential areas for improvement.
- Considered factors like seasonality, economic events, or data anomalies that might influence prediction accuracy.

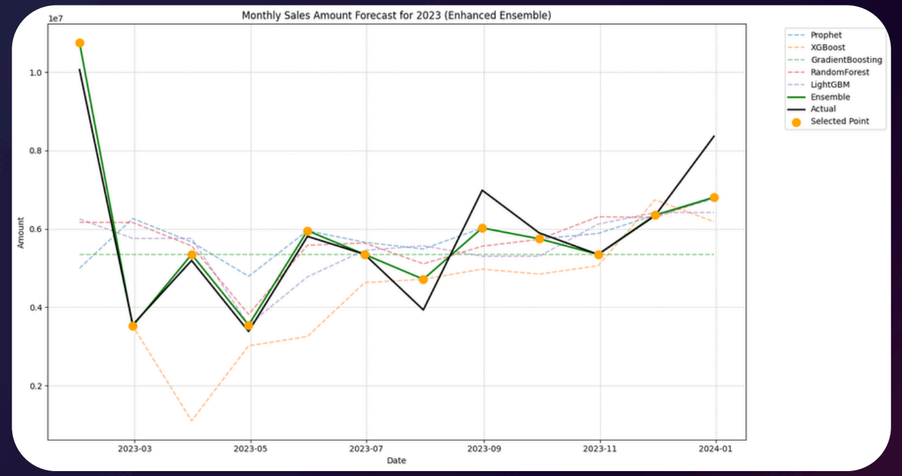
Accuracy Analysis

- Presented the overall accuracy and error metrics for both datasets.
- Discussed the model's performance in terms of prediction accuracy and error levels.

Model Robustness

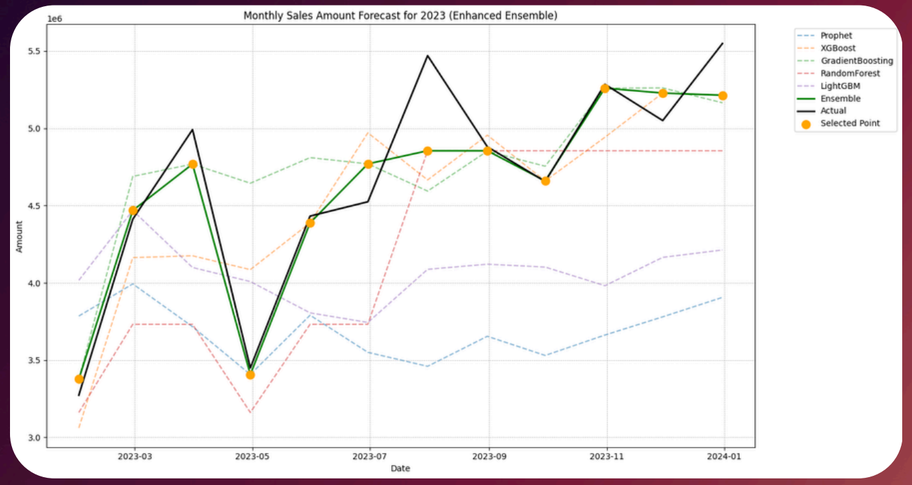


Final Outcome



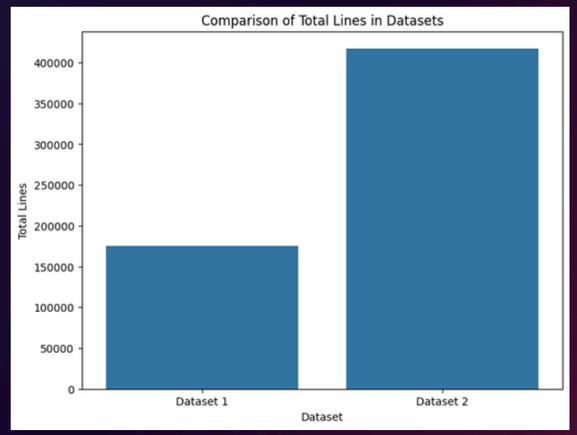
Dataset 1

Accuracy: 93.90% (derived from MAPE)
MAPE: 0.0610
RMSE: 614,523.62



Dataset 2

Accuracy: 96.79% (derived from MAPE)
MAPE: 0.0321
RMSE: 233,015.36



Key findings

- The model demonstrated scalability and adaptability to different data sizes.
- Larger datasets led to improved accuracy, aligning with machine learning principles.
- The **ensemble approach** and **economic indicators** contributed to the model's robustness.
- The model maintained consistent performance across datasets, avoiding overfitting.

Tech Stack Used

Programming Language

Development Environment

- Anaconda
- Jupyter Notebook
- VS Code
- Google Colab

Libraries & Frameworks

Data Manipulation & Analysis
- Pandas
- Numpy
- Scipy

Time Series Forecasting
- Prophet
- cmdstanpy

Machine Learning
- xgboost
- scikit-learn
- gradient boosting
- lightgbm
- randomforest

Optimization
- Optuna

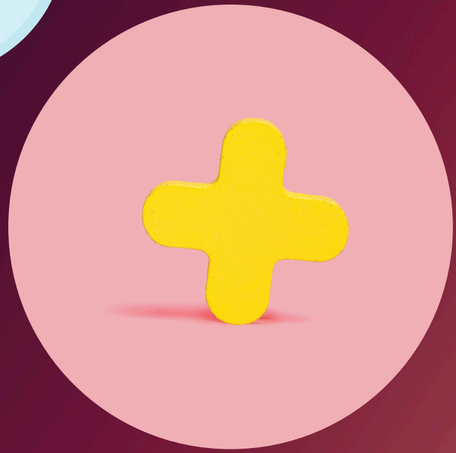
Visualization
- matplotlib
- seaborn

Potential Enhancements

- Incorporate additional domain-specific variables.
- Explore feature selection techniques.

Additional Features or Data

- Integrate external market data.
- Consider other relevant data sources.



Scalability Considerations

- Deploy on cloud-based infrastructure.
- Optimize code for efficiency.

Meet our Team!

AQIL

BADAR

ALIF