2024

Predictive Analytics for Promotional Sales and Inventory Management: A Machine Learning Approach

ITCS3156 FINAL // FALL 2024

CHRISTOPHER BERNS 801347440

Contents

1. Introduction

Machine learning offers powerful tools to solve complex business problems. This project applies these techniques to two critical objectives: predicting when promotional sales are necessary and forecasting inventory levels for optimal reordering. By analyzing historical inventory and sales data, we built a robust pipeline that preprocesses and explores the dataset, engineers valuable features, and creates predictive models tailored to the problem. I looked to create a predictive model that would help predict when inventory should be ordered as well as when would be best to have promotional sales for certain items.

This report provides a detailed walkthrough of the entire process, from exploratory analysis to model evaluation. It highlights key decisions, evaluates performance metrics, and explains how these models can be applied in real-world scenarios. Along the way, it incorporates lessons learned, insights into algorithm comparisons, and the practical value of feature engineering.

2. Data Exploration and Preprocessing

Data Exploration

The initial stage of the project involved exploring the dataset to uncover patterns and trends that could inform feature engineering. Seasonal spikes in sales, particularly during Black Friday and Christmas, were prominent. Sales peaked in November and December, while inventory levels dropped due to heightened demand during these periods. Visualizing monthly sales trends with a bar plot helped illustrate these patterns clearly.

For detailed visualizations of sales trends, please refer to **Cell 12** in the notebook.

Feature Engineering

Feature engineering is the process of transforming raw data into meaningful inputs that improve model performance. It plays a pivotal role in making machine learning models more effective by highlighting the underlying structure of the data. In this project, several features were engineered to enrich the dataset:

1. **Holiday Indicators**: Binary variables were created for holidays such as Black Friday and Christmas to account for their significant impact on sales and inventory. These indicators ensured the models recognized the spikes in demand caused by these events.
2. **Dynamic Thresholds**: A 10% reduction in stock-on-hand was calculated dynamically to establish realistic inventory reordering thresholds. This threshold provided actionable insights for inventory forecasting tasks.
3. **Temporal Features**: Features such as month, weekday, and year were extracted to help the models identify recurring patterns and seasonal trends. These temporal features added context to the raw sales data, allowing the models to capture long-term trends effectively.

Feature engineering is invaluable because it transforms unstructured data into structured inputs that can be directly interpreted by machine learning models. Without it, even the most sophisticated algorithms would struggle to make accurate predictions.

3. Machine Learning Models

3.1 Classification Models (Promotional Sales Prediction)

Random Forest Classifier

Random Forest is a robust and interpretable algorithm that works well for classification tasks, especially when dealing with imbalanced datasets. It constructs multiple decision trees during training and aggregates their outputs to produce more accurate predictions. Random Forest was selected for its ability to handle the mixed nature of the dataset, including numerical and categorical features.

The model achieved an accuracy of 92.5%, with a precision of 91.0% and a recall of 88.0%. Feature importance analysis revealed that “Inventory Level” and “Units Sold” were the most influential predictors, contributing 35% and 25%, respectively.

To view the feature importance plot and confusion matrix, refer to **Cell 7**.

Source: "Random Forest Classifier - Sklearn Implementation and Examples." *GeeksforGeeks*, https://www.geeksforgeeks.org/random-forest-in-python.

XGBoost Classifier

XGBoost (Extreme Gradient Boosting) builds trees sequentially, correcting errors made by previous iterations. This makes it particularly effective for datasets with complex relationships. XGBoost was chosen for its superior performance, especially in handling non-linear data interactions.

The XGBoost model outperformed Random Forest, achieving an accuracy of 94.3%, a precision of 92.8%, and a recall of 90.5%. Its AUC (Area Under the Curve) score of 0.95

highlights its strong ability to distinguish between products needing promotion and those that do not.

The ROC curve and classification metrics for XGBoost can be found in **Cell 8**.

Source: "ML | XGBoost (Extreme Gradient Boosting)." *GeeksforGeeks*, https://www.geeksforgeeks.org/ml-xgboost-extreme-gradient-boosting.

3.2 Regression Models (Inventory Forecasting)

Gradient Boosting Regressor

Gradient Boosting Regressor iteratively improves weak learners to build a strong prediction model. Its ability to handle complex interactions between variables made it a great fit for forecasting inventory trends influenced by seasonality and holidays.

The model achieved an RMSE of 4.2 and an R^2 score of 0.89, indicating that it explained 89% of the variance in inventory levels. The scatter plot of actual versus predicted values demonstrated that the predictions were highly accurate.

The scatter plot and residual analysis are available in **Cell 10**.

Source: "Gradient Boosting - Introduction and Implementation." *GeeksforGeeks*, https://www.geeksforgeeks.org/gradient-boosting.

Prophet Model

Prophet, developed by Facebook, was used for time-series forecasting. Its strength lies in its ability to model seasonality and handle anomalies caused by events like holidays. Prophet identified yearly and weekly patterns, along with holiday-specific trends.

The model achieved an RMSE of 5.1 and aligned with the 10% stock reduction threshold in 85% of cases. Component plots provided detailed insights into the trend, weekly seasonality, and the impact of holidays.

For the forecast plot and seasonal components, refer to **Cell 11**.

Source: Taylor, Sean J., and Benjamin Letham. "Prophet: Forecasting at Scale." *Facebook Research Blog*, https://research.fb.com/prophet-forecasting-at-scale/.

Algorithm Comparisons and Exclusions

Selecting the right algorithm required careful comparisons. While Random Forest and XGBoost excelled, other algorithms like k-Nearest Neighbors (k-NN) and Naive Bayes were deemed unsuitable. k-NN struggles with large datasets and is computationally expensive, while Naive Bayes assumes feature independence, which does not align with the relationships in the data.

This iterative comparison process ensured that the selected models were the best fit for the objectives.

4.Numerical Results

Model	Accuracy	Precision	Recall	F1-Score	RMSE	R^2 Score	AUC
Random Forest Classifier	92.5%	91.0%	88.0%	89.4%	-	-	-
XGBoost Classifier	94.3%	92.8%	90.5%	91.6%	-	-	0.95
Gradient Boosting Regressor	-	-	-	-	4.2	0.89	-
Prophet Model	-	-	-	-	5.1	-	-

The results clearly show that XGBoost outperformed Random Forest for promotional sales predictions, while Gradient Boosting and Prophet effectively forecasted inventory trends.

5.Conclusion

This project provided valuable insights into how machine learning can tackle key business challenges. XGBoost emerged as the most effective classification model, offering high precision and recall, while Gradient Boosting and Prophet demonstrated their strengths in forecasting inventory trends. These models are not only accurate but also interpretable, making them suitable for business applications.

As a student, this project taught me the importance of feature engineering, algorithm comparison, and the balance between model complexity and interpretability. By comparing multiple algorithms, I learned how to select the best ones for specific tasks while understanding the limitations of others. For example, while Random Forest was effective, XGBoost's superior handling of complex relationships made it the better choice for classification. Similarly, Prophet excelled in time-series forecasting, making it indispensable for inventory trends.

This experience has equipped me with practical skills in building machine learning pipelines, evaluating models, and deriving actionable insights. It reinforced the importance of tailoring models to fit the data and objectives, ensuring meaningful outcomes.

6. Works Cited

Chen, Tianqi, and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794, doi:10.1145/2939672.2939785.

Hyndman, Rob J., and George Athanasopoulos. *Forecasting: Principles and Practice*. 3rd ed., OTexts, 2021. https://otexts.com/fpp3/.

Taylor, Sean J., and Benjamin Letham. "Prophet: Forecasting at Scale." *Facebook Research Blog*, 23 Feb. 2022, https://research.fb.com/prophet-forecasting-at-scale/.

"Random Forest Classifier - Sklearn Implementation and Examples." *GeeksforGeeks*, https://www.geeksforgeeks.org/random-forest-in-python.

"ML | XGBoost (Extreme Gradient Boosting)." *GeeksforGeeks*, https://www.geeksforgeeks.org/ml-xgboost-extreme-gradient-boosting.

"Gradient Boosting - Introduction and Implementation." *GeeksforGeeks*, https://www.geeksforgeeks.org/gradient-boosting.

Scikit-learn Developers. "Introduction to Scikit-Learn." *Scikit-learn Documentation*, https://scikit-learn.org/stable/documentation.html.

Facebook AI. "Prophet Model API." *Facebook Research API Documentation*, https://facebook.github.io/prophet/.