Technical Report

Yuanhan Peng
1005354055

Introduction

The primary objective of this project is to develop a predictive model using customer data to forecast the likelihood of customers responding to promotional offers. This project involves utilizing historical transactional data and promotional response records to train a model. The effectiveness of this model will be assessed through a blind test, determining the model's predictive accuracy in unknown scenarios.

Given the training data's imbalance, with only 20% of customers responding to promotions, model performance is primarily assessed using the Receiver Operating Characteristic (ROC) Area Under the Curve (AUC). This metric measures the model's ability to distinguish between responders and non-responders, where a higher AUC score indicates better discrimination. Additionally, precision and recall are evaluated to ensure comprehensive performance assessment: precision quantifies the accuracy of positive predictions, while recall measures the model's ability to identify actual positives.

Dataset Overview

The dataset for this project is structured into four main components, provided in CSV format compressed with gzip for efficiency:

1. *Transactions.csv.gz:* This file contains transaction records for all customers spanning from March 2012 to February 2013. There are records transactional details for all customers over a specified period, including what was bought, where, and its category and manufacturer details.
2. *Promos.csv.gz:* This table provides metadata about each promotion, including the category, quantity and manufacturer of items involved, as well as the financial value of the promotion.
3. *Train_history.csv.gz:* This dataset contains data on promotions given to a subset of customers, including where and when they received these promotions. This dataset forms the training data where customer response is known.
4. *Test_history.csv.gz:* Similar to the train history but for a different subset of customers promotions data from April 2013, this dataset is used for testing the model's predictions on new, unseen data. The model's performance will be evaluated based on its predictions here, which are to be formatted as probabilities (ranging from 0 to 1) indicating the likelihood of customer response.

Exploratory Data Analysis

The transactions dataset features records from 23,347 manufacturers and 24,174 brands across 829 categories, spanning 116 stores. This diversity enables comprehensive analysis of market trends and consumer behaviors within the data science project. The boxplot (Figure 1.) displays

transaction amounts with a median near zero and a narrow interquartile range, indicating primarily low-value transactions. Numerous outliers suggest infrequent high-value transactions and potential refunds. This variability necessitates further analysis to understand diverse consumer spending patterns.

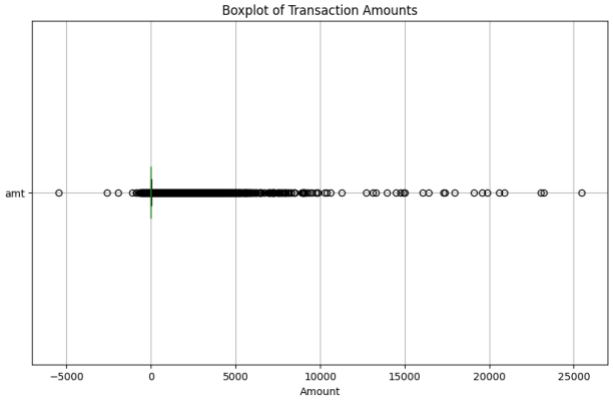

Figure 1.

Figure 2.

The above time series plot (Figure 2.) reveals seasonal sales fluctuations over a year, with a prominent peak suggesting a surge in holiday shopping in November, followed by a sharp decline. Overall, the sales demonstrate seasonality, with the potential for a cyclical pattern that could inform feature engineering strategies.

The promotion dataset consists of 24 unique promotional entries, spanning 13 categories and 11 manufacturers, with 12 distinct brands represented. Each promo record includes details such as promotional ID and value, with promotional quantities uniformly logged as '1' across all entries. This dataset highlights the frequency of promotions, with the most frequent manufacturer appearing in 7 entries and the top category featured in 5 out of 24 records.

Methodology

Feature Engineering

There are 12 effective features used in the final model.

- General information extracted from the promotion information and transaction history:
  1. '*weekday'*: Due to the seasonality noticed before, the weekday of each promotion was extracted, where Monday starts at 0.
  2. *'promoval'*: The dollar value of each promotion.
  3. *'category_rank'* and *'manufacturer_rank'*: Evaluated the popularity of each category and manufacturer based on their frequency of appearance in transaction history.
  4. '*store_size'*: defined by the number of unique product types available, determined by unique combinations of 'category', 'manufacturer', and 'brand'. Larger stores, with more diverse product offerings, generally attract higher customer traffic.
  5. *'region_encoded'*: Label encoded region ID.
- Customer related features:

1. *'recency', 'monetary' and 'frequency': 'Recency'* refers to the time between the promotion date and the nearest purchase date for each customer, indicating engagement levels. *'Frequency'* and *'Monetary'* values measure how often a customer makes a purchase and the total amount spent, respectively, revealing customer loyalty and spending power.
2. *'customer_category_favor', 'customer_manufacturer_favor' and 'customer_brand_favor'*: These features show a customer's loyalty by the percentage of their purchases from preferred category, manufacturers or brand. The assumption is if a customer frequently purchases from a particular category, manufacturer, and brand, they are more likely to respond to promotions involving that manufacturer's products.

Moreover, numerous features tested proved to be ineffective, ultimately lowering the AUC score. For example, *'brand_rank'* is the popularity of each brand based on their frequency of appearance in transaction history. *'store_performance'* is calculated by dividing a store's total sales by its *'store_size'*, assessing how effectively a store uses its product diversity to generate sales, providing insights into the store's sales efficiency relative to its inventory variety. Also, the features *'std_transaction_customer'* and *'return_behavior'* illuminate customer sensitivity to price by calculating the standard deviation of transaction amounts and the average return amount per transaction, respectively, thereby highlighting variability in spending and return habits.

Model Building

The train_history dataset comprises 20,000 rows, with an 80/20 split between the training and testing datasets. Given that 15,950 transactions involve customers not responding to promotions, this creates a highly imbalanced dataset. Initially, the SMOTE method was applied to the training dataset only to address this imbalance. However, it resulted in at least a 2% decrease in model performance (AUC score), leading to its exclusion from the final model.

Then, several classifier models, including Artificial Neural Network (ANN), K-Nearest Neighbor (KNN), Random Forest, and Extreme Gradient Boosting (XGBoost), were initially tested without hyperparameter tuning, where ANN and KNN were tested with standardized data. The XGBoost classifier emerged as the top performer and was selected as the final model. Initially, the model utilized around 20 features. However, when the backward feature selection algorithm was applied to improve accuracy, it unexpectedly lowered both precision and recall without enhancing the AUC score, indicating a decline in model performance. Therefore, in the feature selection phase, a set of 12 effective features introduced above was carefully selected based on the model's feature importance, combining subjective insights with extensive manual testing of various feature combinations. The removal of *'brand_rank'* is an example. This strategic selection optimized the model's performance.

Overall, the initial performance of the XGBoost model before hyperparameter tuning achieved a test accuracy of 80.025% and an AUC of 0.67885. The model effectively identifies non-

responders, evidenced by high precision (0.82) and recall (0.96) for the majority class, yet it indicates a need for improved accuracy in predicting positive cases.

Next, *GridSearchCV* is employed to systematically explore multiple combinations of hyperparameters within a predefined grid, aiming to optimize the model performance. Key parameters such as *'max_depth'*, *'n_estimators'*, *'learning_rate'*, and *'scale_pos_weight'* are adjusted to improve the model's complexity, learning speed, and handling of class imbalances. Importantly, this process incorporates cross-validation (using a 5-fold strategy) to ensure that the model's performance is robust across different subsets of the data. After hyperparameter tuning, the best model demonstrated an improvement, achieving a test accuracy of 79.75% and an AUC of 0.70858. The precision for non-responders slightly improved to 0.85, with recall dropping to 0.91, showcasing enhanced effectiveness in identifying the majority class. However, despite a slight improvement in precision for positive cases to 0.49, the low recall of 0.33 indicates that accuracy in predicting positive responses still requires further enhancement.

Then, built a model with the parameters from previous best model, where *'max_depth'* = 3, *'n_estimators'* = 100, *'learning_rate'* = 0.1, and *'scale_pos_weight'* = 2.5, and trained it with all data from train_history dataset. The training accuracy is 79.535%, AUC is 0.72873. Then, we prepared the data from test_history the same way as we process train_history. The best model was applied on the test_history dataset and make predictions on the likelihood of provided customers responding to their promotional offers.

Future Directions

1. Advanced Feature Engineering:
   Explore additional features that could capture more nuances in the data. For example, creating interaction terms between features or extracting new features from existing data through techniques like Principal Component Analysis (PCA) could reveal hidden patterns.
2. Deeper Hyperparameter Optimization:
   Although the project has implemented hyperparameter tuning, leveraging advanced techniques such as Bayesian optimization could yield more nuanced adjustments and improved results. Expanding the search to include a broader spectrum of values and additional hyperparameters, such as 'colsample_bytree' and 'subsample,' may enhance the model's precision and stability.