
Credit Card Behaviour Score: Model Development Documentation

Introduction

At Bank A, we recognize the growing importance of refining our risk management strategies. To better anticipate customer defaults and manage portfolio risk, we've developed the "Behaviour Score"—a predictive model designed to estimate the likelihood of default on credit card accounts. By leveraging historical data and customer attributes, the model can help proactively identify high-risk accounts, allowing us to make informed decisions that ensure profitability while minimizing risk.

Problem Statement

The goal of this project is to create a model capable of predicting whether a credit card customer is likely to default. To achieve this, the model will:

- Utilize a dataset consisting of credit card account details and a "bad_flag" indicating whether or not a customer has defaulted.
- Produce a set of predictions for unseen data, returning the account number and the predicted probability of default.

Datasets Provided

We received two datasets for model development and validation:

1. **Development Data**:
    - A total of 96,806 records, including features such as:
        - **Account number**: A unique identifier for each customer.
        - **Bad_flag**: Indicates if the customer defaulted (1) or not (0).
        - **Various customer attributes**, including credit limits, transaction counts, delinquencies, and inquiry history.
2. **Validation Data**:
    - Contains 41,792 records, similar to the development data but without the "bad_flag". Our task here is to make predictions that will be submitted for evaluation.

Approach to Solving the Problem

To build an effective model, we followed these key steps:

1. Data Preparation

Before training the model, the dataset needed to be cleaned and transformed:

- **Missing Values**: We handled missing data by imputing the mean or median for numeric variables and the mode for categorical ones.
- **Feature Encoding**: Categorical features were encoded using one-hot encoding, ensuring that they could be interpreted by the model.
- **Feature Scaling**: We standardized the numerical variables to ensure they had equal weight in the model.
- **Class Imbalance**: The bad_flag variable was imbalanced, with more non-defaults than defaults. We addressed this by adjusting class weights to help the model learn the minority class more effectively.

2. Model Development

For the actual modeling, we selected **XGBoost**—a powerful and efficient classifier that excels at binary classification tasks. Some important configurations were:

- **Objective**: We used binary:logistic to output probabilities.
- **Evaluation Metric**: Log loss was chosen to assess how well the model predicts probabilities.
- **Hyperparameters**: We fine-tuned the model with settings like learning_rate=0.1, n_estimators=100, and max_depth=6 to balance performance and training time. To combat overfitting, we applied subsampling and column sampling (subsample=0.8 and colsample_bytree=0.8).
- **Class Weighting**: The scale_pos_weight parameter was adjusted to compensate for the imbalance in the target class.

3. Validation and Evaluation

We evaluated the model's performance on a held-out validation set. The key metrics used were:

- **Log Loss**: Measures the accuracy of the predicted probabilities.
- **ROC-AUC**: Evaluates the model's ability to distinguish between defaults and non-defaults.

- **Precision-Recall Curve**: Particularly useful due to the class imbalance, as it highlights the model's performance in detecting defaults.

The results showed that the model was well-calibrated and able to distinguish between the two classes effectively.

4. Prediction on Validation Data

Once the model was trained, it was applied to the validation dataset to generate predictions. The output consisted of two columns: the **account_number** and the **predicted_probability**. All probabilities were rounded to three decimal places for clarity, and the results were stored in a CSV file, ready for submission.

Key Insights and Observations

- **Feature Importance**: Some features were significantly more predictive of defaults than others. For instance, customer credit limits, historical delinquencies, and recent credit inquiries played a critical role.
- **Handling Imbalanced Data**: The imbalance between default and non-default cases was noticeable, but techniques like class weighting helped mitigate its impact on model performance.
- **Data Quality**: Missing values and outliers were addressed early on in preprocessing, allowing the model to focus on the more relevant features.
- **XGBoost's Effectiveness**: Compared to simpler models like logistic regression, XGBoost handled the non-linear relationships in the data much better, leading to improved performance.

Conclusion

The Behaviour Score model has proven to be an effective tool for predicting credit card defaults, showing promising results on both the training and validation datasets. By utilizing this model, Bank A can take proactive measures to minimize risk, such as:

- **Adjusting credit limits** for customers at higher risk of default.
- **Monitoring high-risk accounts** more closely.
- **Developing targeted retention strategies** for customers on the verge of default.

This model represents a significant step toward a more data-driven approach to risk management.

Submission Details

The predictions for the validation dataset have been compiled into a CSV file, validation_predictions.csv. The file contains the following columns:

- **Account_number**: A unique identifier for each customer.
- **Predicted_probability**: The probability that the customer will default, rounded to three decimal places.

The file has been thoroughly checked to ensure that the probabilities are between 0 and 1, as expected.

Future Recommendations

Looking ahead, there are several opportunities to enhance the model further:

1. **Feature Engineering**: More advanced features, such as time-series trends or behavioral patterns in transaction data, could provide additional predictive power.
2. **Model Enhancement**: Techniques like ensemble learning or stacking could help boost performance even further.
3. **Model Explainability**: Implementing tools like SHAP values could help provide insights into individual predictions, improving transparency.
4. **Monitoring and Retraining**: As customer behavior evolves, it will be important to periodically retrain the model to ensure its continued effectiveness.

With these improvements, the Behaviour Score model can continue to evolve and provide valuable insights into credit card risk management.
