Prediction of Credit Card Behaviour Score

1 Introduction

Effective risk management is vital for the sustainability and profitability of financial institutions, especially in the credit card business. With evolving customer behaviors, a robust framework is required to assess the risk of default by existing credit card customers. This report introduces the *Behavior Score* model, a predictive tool designed to estimate the likelihood of future defaults, enhancing portfolio risk management at Bank A.

The model leverages a historical dataset of 96,806 credit card accounts with a `bad_flag` indicating past defaults, and a validation dataset of 41,792 accounts without the flag. These datasets include variables such as credit limits, transaction behaviors, and credit bureau attributes, providing a comprehensive foundation for predictive modeling.

This study details the methodology, analysis, and results of developing the Behavior Score model using advanced machine learning techniques. The insights generated are action-ready, supporting activities like credit limit adjustments, targeted collections, and portfolio optimization.

2. Data Preprocessing

Data preprocessing is a critical step in the machine learning pipeline, ensuring that the dataset is clean, well-structured, and suitable for modeling. This section outlines the steps taken during the preprocessing phase, which include handling missing values, removing irrelevant or redundant features, and applying dimensionality reduction techniques. The goal is to enhance the performance of machine learning algorithms by providing them with high-quality input data.

2.1 Removing Highly Correlated Features

- Columns with high correlation (i.e., correlation coefficient greater than 0.9) were identified and one of the correlated features was removed from each highly correlated pair.
- This step is essential to avoid multicollinearity, which can adversely affect the performance of some machine learning models.

2.2 Handling Missing Data

- Missing data were handled by applying Simple Imputer with the median strategy, filling missing values with the median of the respective column.

- Rows with more than 50% missing values were dropped from the dataset to maintain data integrity.

2.3 Feature Scaling and Transformation

- **Standardization:** Features with different scales were standardized using StandardScaler, ensuring that each feature contributes equally to the model.
- **Transformation:** Applied Yeo-Johnson transformation to correct skewed data and improve the model's performance by stabilizing variance.

2.4 Dimensionality Reduction: Principal Component Analysis (PCA)

- PCA was applied to reduce the dimensionality of the dataset, retaining 95% of the variance in the data.
- This technique helped in visualizing high-dimensional data in a lower-dimensional space and improved computational efficiency for downstream algorithms.

2.5 Feature Selection: Correlation-Based Feature Selection (CFS)

- Used Swarm Search Algorithm to select the top 30 features based on their relevance and redundancy.
- This step ensured that only the most important features were retained, improving the model's efficiency and interpretability.

3. Detailed Steps

3.1.1 Applied Algorithms

Naïve Bayes and Random Forest was applied to evaluate model performance on the above preprocessed data using feature engineering techniques. The results are summarized in Table given below.

Approach	**ROC-AUC Score**	**F1-Score**
PCA + Swarm Search + Naïve Bayes	0.75	0.081
PCA + Swarm Search + Random Forest	0.773	0.105

Table 1: Performance Comparison of the Two Approaches

4 Proposed Approach

4.1 Our preprocessing Technique

- **Double Ensemble Techniques for Feature Importance:**

  - XGBoost and Random Forest were used to rank feature importance.
  - Top 600 features were selected based on importance scores.

- **Handling Missing Data:** Rows with more than 40% missing values were removed in the third approach.
- **Feature Selection and Classification:** Voting Classifier combining Random Forest and LightGBM was used for optimal performance.

Performance Summary

Approach	ROC-AUC Score	F1-Score
Feature Importance + Random Forest	0.8119	0.3835
Feature Importance + Voting Classifier (**Best**)	0.8529	0.3944

Table 2: Performance Comparison of the Last Two Approaches

5. Key Insights

- **Imputation Challenges:** Simple imputation caused information loss and potential misinformation.
- **Feature Selection Impact:** Extracting feature importance from ensemble techniques significantly improved model performance.
- **Missing Data Handling:** Removing rows with substantial missing data yielded better results than imputation.
- **Model Combinations:** Voting Classifier with Random Forest and LightGBM achieved the highest metrics.

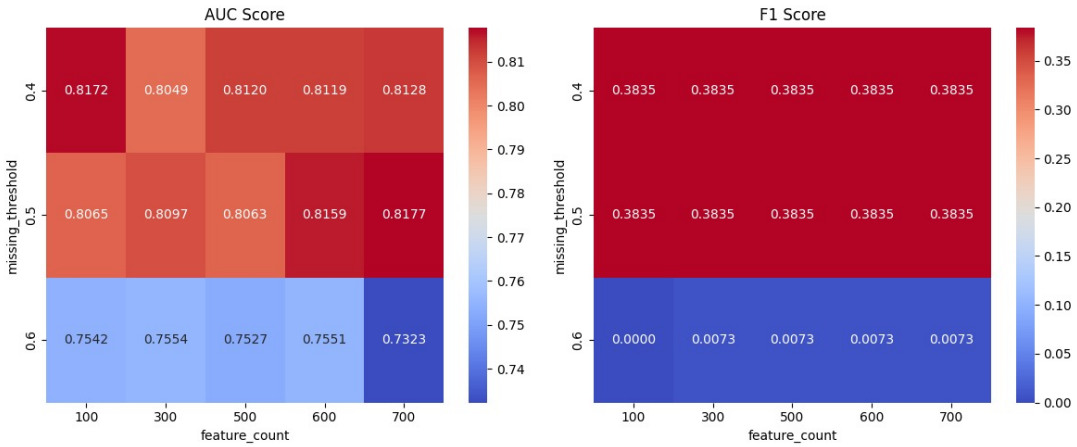

Figure 1: Performance based on no of features and missing values

6. Metrics Used

ROC-AUC Score

Measures the model's ability to distinguish between classes. Higher values indicate better classification performance.

F1-Score

Harmonic mean of precision and recall. Emphasizes the balance between false positives and false negatives.

Technique Used	ROC-AUC Score	F1-Score
XGB	0.805	0.3836
RF	0.311	0.3835
LGBM	0.833	0.3339
CB	0.8007	0.2831
XGB + RF	0.827	0.3884
RF + LGBM	**0.8529**	**0.3944**
RF + CB	0.8236	0.3753
XGB + CB	0.8261	0.3035
XGB + LGBM	0.8493	0.3077
LGBM + CB	0.8523	0.2701
XGB + LGBM + CB	0.8523	0.3339
RF + LGBM + CB	0.8533	0.3745
RF + XGB + CB	0.8301	0.3655
RF + XGB + LGBM	0.8504	0.3711
RF + XGB + CB + LGBM	0.8531	0.3614

Table 3: Performance Comparison of Techniques Used

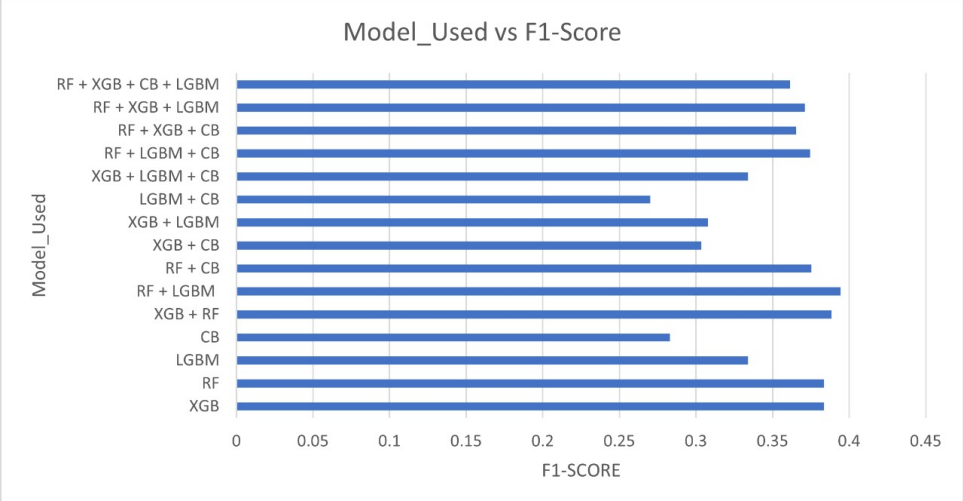

Figure 2: Model vs F1 Score

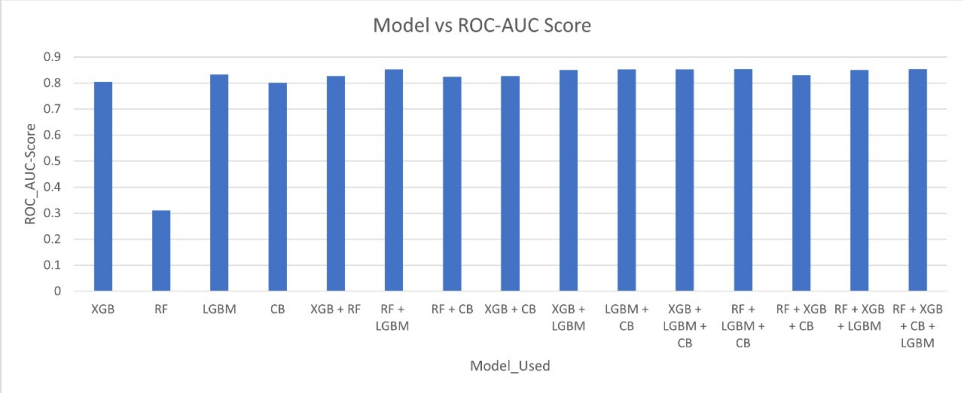

Figure 3: Model vs ROC-AOC Score

7. Conclusion and Recommendations

- **Optimal Approach:** The third approach (Feature Importance + Voting Classifier) demonstrated superior performance during training process.
- **Key Takeaways:**
  - Avoid imputation where feasible.
  - Use ensemble techniques for robust feature selection.
  - Leverage combinations of models in a Voting Classifier for better generalization.
- **Future Work:**
  - Explore hyperparameter tuning for further optimization.
  - Investigate additional ensemble combinations and advanced preprocessing techniques.
  - Address the issue of rows in the test data containing more than 40% missing values, which were removed during training but not in testing.

8. References

- Sudhansu R. Lenka, Sukant Kishoro Bisoy, Rojalina Priyadarshini, and Mangal Sain. (2022). Empirical Analysis of Ensemble Learning for Imbalanced Credit Scoring Datasets: A Systematic Review.
- Simon Fong, Yan Zhuang, Rui Tang, Xin-She Yang, and Suash Deb. (2013). Selecting Optimal Feature Set in High-Dimensional Data by Swarm Search.
- Chuheng Zhang, Yuanqi Li, Xi Chen, Yifei Jin, Pingzhong Tang, and Jian Li. (2020). DoubleEnsemble: A New Ensemble Method Based on Sample Reweighting and Feature Selection for Financial Data Analysis.