MMF2000 RISK MANAGEMENT

AML Risk Assignment

Shuqi (Serena) Chang
Yirong (Frederick) Weng
Xinyi (Cynthia) Shen
Songyuan (Yannis) Zhang

Master of Mathematical Finance

University of Toronto

November 28, 2024

Table of Contents

1 Introduction

This assignment is to build an Anti-money laundering (AML) client risk rating (CRR) model. A CRR model is a model that assigns a rating (high, moderate, low) to a customer at the time of customer onboarding and for periodic updates; the rating is from AML perspective, and considers customer KYC (know your customer) information and transaction patterns. We aim to build a model that assigns a rating to a client according to the features provided using statistical and machine learning models.

2 Data Preparation

Data is in the file `AML and ATF Modelling Assignment data_std_Updated.xlsx`. A data dictionary is enclosed in the same file for you to understand the meaning of the data elements. All features have already been standardized. Below is a summary of the type of data elements.

- **ID** (`cust_id_masked`): Unique Customer ID.
- **Target** (`rating`): This is the target of the model, or the true rating assigned by adjudicators. The 3 categories (high, moderate, and low risk) have been simplified to high and non-high, therefore, a binary classification model would suffice.
- **Features:** There are 149 features provided in the data. You do not need to include all of them in the final model. You can:
  i. Perform feature selection by machine learning methods, or
  ii. Make judgmental decisions on inclusion or exclusion based on your understanding of each feature's impact on a client's AML risk.

After we import necessary libraries in Python Jupyter Notebook, we load and prepare data by reading the `data` sheet in the Excel file `AML and ATF Modelling Assignment data_std_Updated.xlsx` and dropping the unnecessary column, `cust_id_masked`. By checking missing values, we have confirmed that the dataset is complete and does not require handling for missing data.

By checking normality of the whole dataset, we have found that there are no p-values greater than 0.05 using the Shapiro-Wilk test, therefore, none of the features in the dataset follow a normal distribution at the 5% significance level. This suggests that normal standardization is not suitable for this dataset, so we may need to investigate skewness and outliers, explore scaling methods, and consider data transformations for next steps. Before that, we first differentiate binary and non-binary data from the dataset for better investigations.

We load the `data dictionary` sheet from the same Excel file to understand feature meanings and types and to identify binary features from the dictionary by searching for "Binary" contained in the descriptions. Table 1 shows the selected binary features with descriptions. We also add our target binary variable `rating` to our binary data list, which contains 390 moderate or low-risk indicators and 198 high-risk indicators. Now, we have found 18 binary variables in total, and our binary variable list is as follows

```
['cust_cdn_resident',
 'prod_sav',
 'prod_cda',
 'prod_mor',
 'prod_llc',
 'prod_ted',
 'prod_crc',
 'geo_mail_addr_country_rate_high',
 'geo_mail_addr_country_rate_med',
 'geo_mail_addr_country_rate_low',
 'geo_domicile_country_rate_high',
 'geo_domicile_country_rate_med',
 'geo_domicile_country_rate_low',
 'in_person_onboard_f',
 'occu_h',
 'prod_med',
 'rating']
```

Type	Data Element	Description
Feature	cust_cdn_resident	Binary indicator whether the customer resides in Canada.
Feature	prod_sav	Binary indicator for savings account.
Feature	prod_cda	Binary indicator for chequing account.
Feature	prod_mor	Binary indicator for mortgage.
Feature	prod_llc	Binary indicator for loan and line of credit.
Feature	prod_ted	Binary indicator for term deposit.
Feature	prod_crc	Binary indicator for credit card.
Feature	geo_mail_addr_country_rate_high	Binary indicator for mailing address being high risk.
Feature	geo_mail_addr_country_rate_med	Binary indicator for mailing address being medium risk.
Feature	geo_mail_addr_country_rate_low	Binary indicator for mailing address being low risk.
Feature	geo_domicile_country_rate_high	Binary indicator for residence country being high risk.
Feature	geo_domicile_country_rate_med	Binary indicator for residence country being medium risk.
Feature	geo_domicile_country_rate_low	Binary indicator for residence country being low risk.
Feature	in_person_onboard_f	Binary indicator as to whether there ever was an in-person onboarding.
Feature	occu_h	Binary indicator of high-risk occupation.
Feature	prod_med	Binary indicator of medium-risk product.

Table 1: Filtered Binary Feature Description Table

Then we plot distributions for 134 non-binary features group by rating in Figure 1, 2, 3, 4, 5, 6.

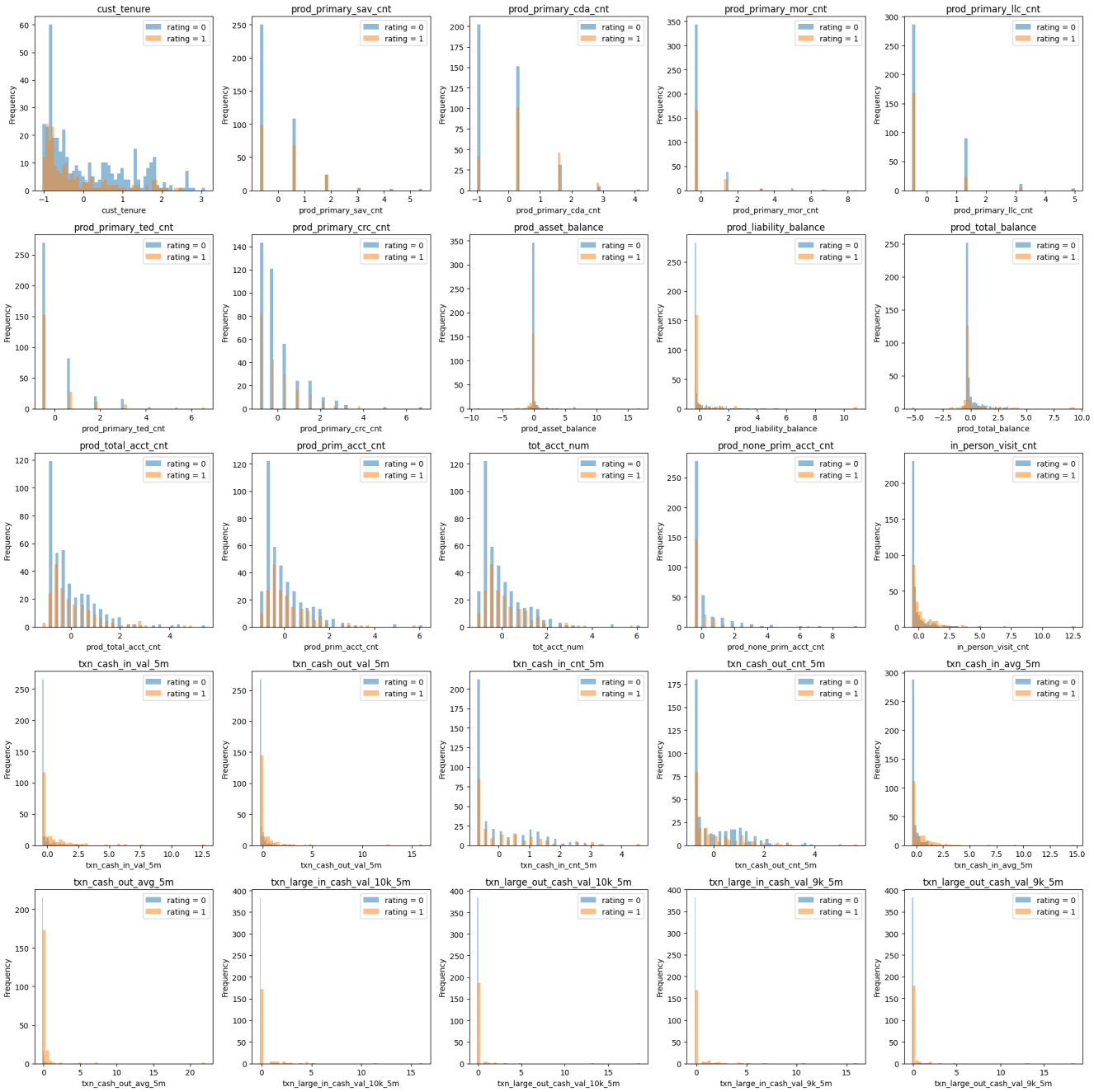

Figure 1: Distributions for non-binary features group by rating from 1st to 25th

Figure 2: Distributions for non-binary features group by rating from 26th to 50th

Figure 3: Distributions for non-binary features group by rating from 51st to 75th

Figure 4: Distributions for non-binary features group by rating from 76th to 100th

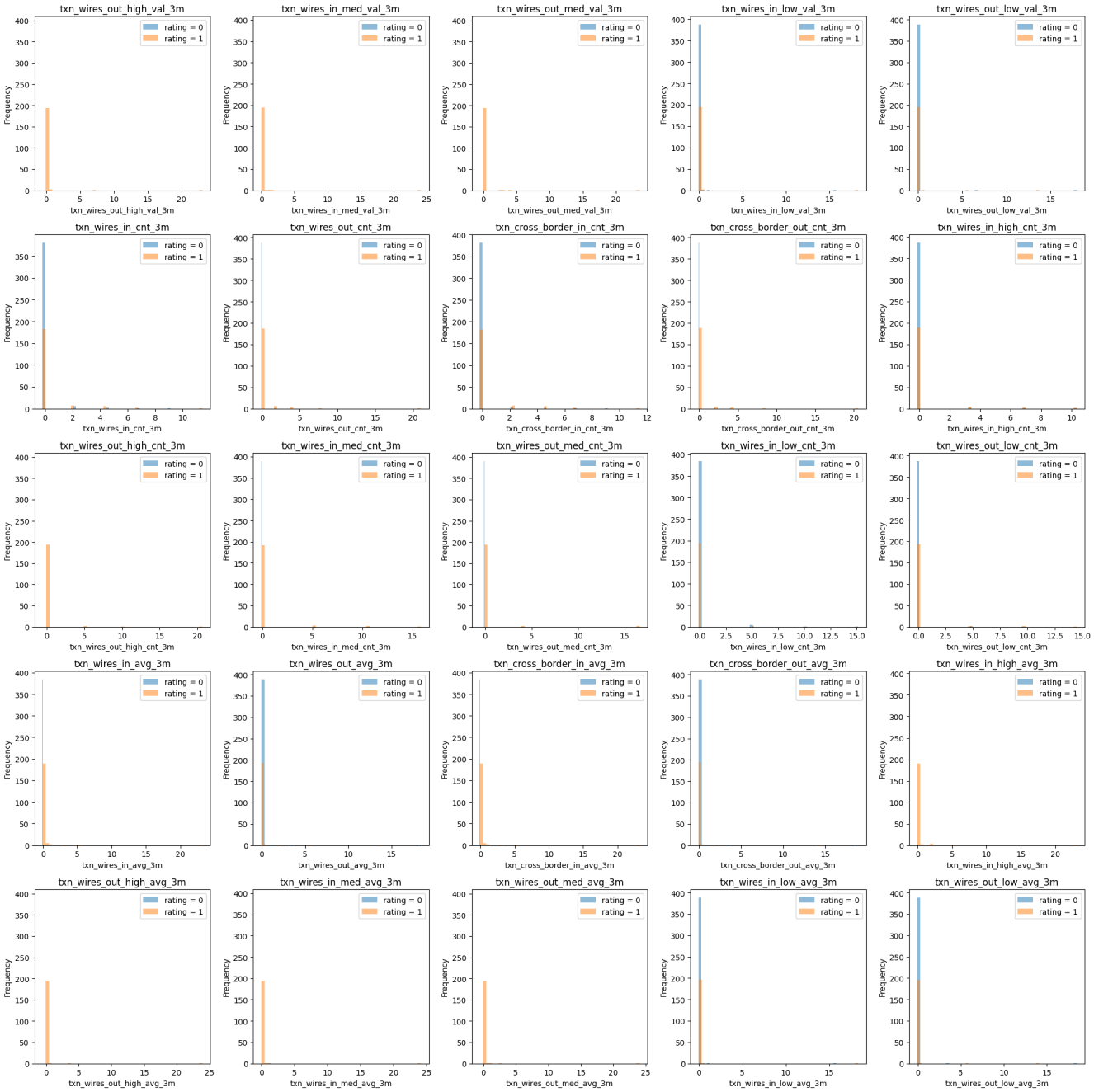

Figure 5: Distributions for non-binary features group by rating from 101st to 125th

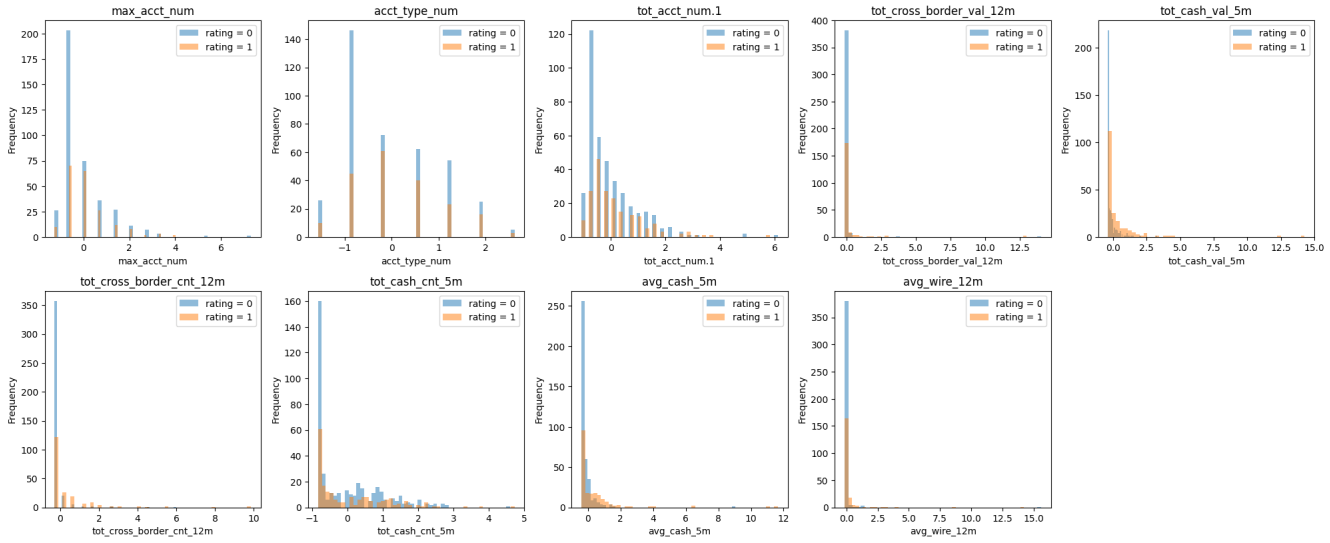

Figure 6: Distributions for non-binary features group by rating from 126th to 134th

Upon examining the histograms provided in Figures 1 to 6 illustrate the distribution of 134 non-binary features, grouped by the binary target variable, `rating`, the following features demonstrate significant differentiation across the `rating` groups and thus have predictive potential:

1. **`txn_cash_out_cnt_3m`:** Higher frequencies of cash withdrawals over 3 months correlate strongly with higher risk ratings.
2. **`tot_cash_val_5m`:** Large aggregate cash transaction values over 5 months are more prevalent among high-risk ratings.
3. **`prod_primary_llc_cnt`:** Indicates specific product preferences tied to higher-risk profiles.
4. **`geo_mail_addr_country_rate_high`:** Strongly differentiates high-risk clients based on location-based risk indicators.
5. **`txn_large_in_cash_val_10k_3m`:** Large cash inflows over 10k in the past three months show a noticeable difference between the `rating` groups.

These features align with domain-specific expectations that high cash transactions, product-specific usage, and geographic risks are key indicators of AML risk.

3 Data Transformation

Based on observations from the data preparation phase, the features in the dataset operate on vastly different scales. This can lead to issues in machine learning models like SVMs, PCA, or neural networks, which assume equal feature contributions. Without proper scaling, features with larger ranges can dominate and distort model predictions or slow convergence in optimization-based models. As a result, we apply the Min-Max Scaling method in the pipeline to transform each feature to a range of $[0, 1]$, making them comparable in magnitude. It avoid dominance by features with larger scales, ensuring balanced feature contributions. The formula is

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)},$$

where

- x is the original value.
- x' is the scaled value.
- $\min(x)$ and $\max(x)$ are the minimum and maximum values of the feature.

In this dataset, while binary features and the target variable (`rating`) are already normalized, interactions between continuous features can also reveal meaningful relationships, thus providing more insight into a customer's risk profile than either feature alone. Hence, we also consider cross terms, i.e., the interactions between features, to capture relationships that might not be apparent in individual features. Considering the compounded effects of features may improve the model's ability to explain and predict complex patterns in the data, making the dataset become richer and leading to more nuanced and accurate predictions.

Therefore, feature engineering and data transformation are very necessary in order to train this dataset. For feature engineering, we created interaction terms between all the non-binary features, resulting in approximately 9,000 individual and combined features. Then, we applied Min-Max Scaling to the training dataset to normalize feature values and ensure consistency in data transformation. The same scaling parameters derived from the training data were then applied to the testing dataset to prevent data leakage and maintain uniformity between datasets.

4 Splitting Training and Testing Datasets

To split the dataset into training and testing sets, we utilized the `train_test_split` function from `sklearn.model_selection`. By setting `stratify=y`, we separate the target variable (`rating`) from the features before splitting, and then we allocate 80% of the data to the training set and 20% to the testing set, ensuring that the class distribution of the target variable is preserved in both training ans testing datasets. The stratified splitting maintains the balance between the "high-risk" and "non-high-risk" categories across both training and testing datasets.

To address class imbalance within the training set, we apply SMOTE (Synthetic Minority Oversampling Technique), which generates synthetic examples for the minority class. This ensures that the model had an equal representation of both classes during training, allowing it to better learn the distinguishing features of each class. Following this, we apply Min-Max Scaling to both the training and testing datasets. The scaler is first fitted on the training data and then applied to the test set to prevent data leakage.

The combined approach of stratified splitting, resampling with SMOTE, and Min-Max Scaling provides robust training while preventing overfitting or bias due to data imbalance or feature scale differences.

In the following feature selection section, we select features base on the training set.

5 Feature Selection

It is necessary to perform feature selection, especially for this dataset, which contains 149 features. Feature selection improves model interpretability, reduces computational complexity, and mitigates overfitting. Many features in the dataset may have little to no predictive power or may be highly correlated with one another, leading to redundancy. Retaining irrelevant or redundant features can degrade model performance by introducing noise, inflating variance, and increasing the risk of overfitting.

5.1 Forward and Backward Selection

5.1.1 Forward Selection and Backward Elimination based on p-value

For one thing, we apply the iterative forward selection and backward elimination methods based on p-values from Ordinary Least Squares (OLS) regression to evaluate the contribution of each feature to the target variable. Forward selection begins with no features in the model and adds the features with low p-values below the threshold of `threshold_in=0.01` to the model stepwise. Backward elimination starts with all features included in the model and removes the features with high p-values above the threshold of `threshold_in=0.05` from the model stepwise. Using this approach, we identified 20 key features in Table 2.

Feature	**p-value**
prod_cda	5.7706×10^{-12}
txn_cash_out_cnt_3m	3.49468×10^{-10}
tot_cash_val_5m	1.0477×10^{-6}
geo_mail_addr_country_rate_high	1.70195×10^{-5}
prod_primary_mor_cnt*txn_wires_in_cnt_12m	1.40764×10^{-5}
geo_domicile_country_rate_high	7.34945×10^{-6}
prod_llc	0.000178232
cust_tenure*max_acct_num	7.48937×10^{-5}
prod_primary_ted_cnt*txn_cash_out_cnt_3m	2.56502×10^{-5}
prod_none_prim_acct_cnt*in_person_visit_cnt	0.000118662
txn_cash_in_cnt_3m	0.00053505
txn_wires_in_low_cnt_3m*txn_large_in_cash_val_10k_3m	8.26595×10^{-5}
in_person_visit_cnt*txn_large_out_cash_val_10k_3m	0.000300527
prod_ted	0.00121884
txn_cash_in_val_3m*avg_cash_5m	0.00196187
prod_primary_crc_cnt*tot_cash_val_5m	0.00346313
prod_primary_llc_cnt*txn_cash_out_cnt_5m	0.00617786
txn_cash_in_cnt_5m	0.00780432
txn_wires_out_med_val_12m*txn_cross_border_in_avg_12m	0.0047765
txn_cash_out_avg_3m*txn_wires_out_val_6m	0.00762383

Table 2: Selected Features using Forward Selection and Backward Elimination based on p-values

5.1.2 Forward Selection based on BIC

For another, we apply Forward selection based on and then Bayesian Information Criterion (BIC) to identify features that provides the best model fit with balanced model complexity. While Akaike Information Criterion (AIC) measures the trade-off between the goodness of fit of the model and the complexity of the model, BIC applies a stronger penalty for model complexity, leading to a simpler model.

The selected features and the best model summary are in Table 3.

Feature	**Coefficient**	**Std. Error**	**P-value**
Intercept	1.3178	0.4190	0.0020
prod_cda	0.3190	0.0370	<0.0001
txn_cash_out_cnt_3m	-1.0073	0.1190	<0.0001
tot_cash_val_5m	2.3501	0.2670	<0.0001
geo_mail_addr_country_rate_high	0.3202	0.0730	<0.0001
prod_primary_mor_cnt*txn_wires_in_cnt_12m	-1.8476	0.3000	<0.0001
prod_llc	-0.2084	0.0430	<0.0001
cust_tenure*max_acct_num	1.1264	0.2610	<0.0001
prod_primary_ted_cnt*txn_cash_out_cnt_3m	0.7661	0.1750	<0.0001
prod_none_prim_acct_cnt*in_person_visit_cnt	-0.7782	0.2580	0.0030
txn_cash_in_cnt_3m	-2.7120	0.7170	<0.0001
txn_wires_in_low_cnt_3m*txn_large_in_cash_val_10k_3m	0.8864	0.2940	0.0030
in_person_visit_cnt*txn_large_out_cash_val_10k_3m	-1.3472	0.3690	<0.0001
prod_ted	-0.1557	0.0410	<0.0001
txn_cash_in_val_3m*avg_cash_5m	-1.8625	0.4680	<0.0001
prod_primary_crc_cnt*tot_cash_val_5m	-1.0989	0.3200	0.0010
prod_primary_llc_cnt*txn_cash_out_cnt_5m	0.6711	0.2270	0.0030
txn_cash_in_cnt_5m	2.0622	0.7400	0.0050
txn_wires_out_med_val_12m*txn_cross_border_in_avg_12m	-1.1445	0.3220	<0.0001
txn_cash_out_avg_3m*txn_wires_out_val_6m	0.8414	0.3140	0.0080

Table 3: Selected Features using Forward Selection based on BIC

5.2 Embedded Method using LASSO Regression

Then we apply the embedded method, Least Absolute Shrinkage and Selection Operator (LASSO) regression , to inherently perform feature selection by penalizing less important features during the optimization process. This ensures that the model selects the most relevant features while fitting the data. LASSO regression adds the L1 penalty to the loss function, which encourage sparsity in the model's coefficients. The optimization objective in LASSO regression is

$$\min_{\beta} \left(\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{p} |\beta_j| \right),$$

where

- β_j: Coefficients of the features.

- α: Regularization parameter controlling the trade-off between fitting the data and penalizing the model's complexity.
- $|\beta_j|$: Absolute value of feature coefficients, encouraging some coefficients to become exactly zero.

By shrinking some coefficients to zero, LASSO effectively removes irrelevant or less significant features from the model. Selected features are in Table 4

Selected Features using LASSO Regression (α = 0.005)
prod_cda
prod_primary_cda_cnt
prod_mor
prod_llc
prod_ted
prod_crc
geo_mail_addr_country_rate_high
geo_domicile_country_rate_high
txn_cash_out_cnt_3m
occu_h
prod_med
max_acct_num
tot_cash_val_5m
tot_cross_border_cnt_12m
tot_cash_cnt_5m
txn_wires_out_med_avg_6m*txn_wires_in_high_cnt_12m
prod_primary_ted_cnt*txn_cash_out_cnt_3m
prod_primary_ted_cnt*tot_cash_cnt_5m
txn_wires_out_high_cnt_12m*tot_cash_cnt_5m
txn_wires_out_high_cnt_12m*txn_wires_in_med_cnt_6m

Table 4: Selected Features using LASSO Regression (α = 0.005)

5.3 Cross Comparison with PCA

To complement the previous feature selection methods of stepwise selection and LASSO regression that rely solely on statistical significance or regularization penalties, we apply the cross-comparing Principal Component Analysis (PCA) method to select the most significant features based on the power of explaining model variability.

According to the plot of explained variance ratio by principal component in Figure 7 and the cumulative explained variance in the following output, we use the 70% cumulative explained variance as a threshold to ensures a significant portion of the variance is captured while avoiding overfitting by discarding PCs that explain less variance likely noise or redundancies.

```
Cumulative Explained Variance:
[0.15965508 0.25361351 0.32818784 0.38638144 0.43267099 0.47589221
 0.51691984 0.5560108  0.59489266 0.62288662 0.64458796 0.66568111
```

```
0.68398239 0.69905108 0.71320051 0.72679515 0.73941346 0.75137194
0.763168   0.77437276 0.78457356 0.79411795 0.80334597 0.81233541
0.8210709  0.82936375 0.83748739 0.84516569 0.85264897 0.85979733]
```

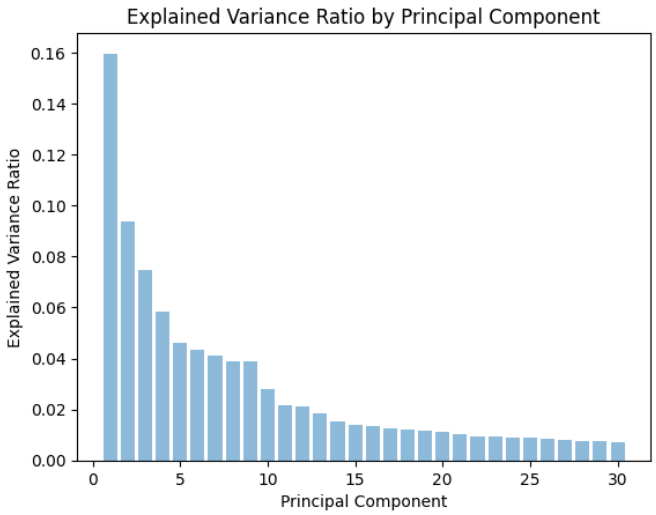

Figure 7: Explained Variance Ratio by Principal Component

5.3.1 Individual Principal Component Contributions (Threshold: 0.1)

To identify features that contribute significantly to individual principal components, we set the threshold for significant contributions of features as 0.1. The selected features are in Table 5.

Selected Features by Individual PC Contributions
acct_type_num
cust_cdn_resident
geo_domicile_country_rate_high
geo_domicile_country_rate_low
geo_mail_addr_country_rate_high
geo_mail_addr_country_rate_low
in_person_onboard_f
occu_h
prod_cda
prod_crc
prod_llc
prod_med
prod_sav
prod_ted
tot_cash_cnt_5m

Table 5: Features Selected Based on Individual PCA Contributions (Threshold: 0.1)

5.3.2 Total Contributions Across All Principal Components (Threshold: 0.4)

To identify features with significant overall contributions across all PCs, features with cumulative loadings greater than 0.4 are considered important and retained in the selected feature list in Table 6.

Selected Features by Total PC Contributions
cust_cdn_resident
prod_sav
prod_cda
prod_primary_cda_cnt
prod_mor
prod_llc
prod_ted
prod_crc
prod_primary_crc_cnt
prod_prim_acct_cnt
tot_acct_num
geo_mail_addr_country_rate_high
geo_mail_addr_country_rate_low
geo_domicile_country_rate_high
geo_domicile_country_rate_low
in_person_onboard_f
txn_cash_in_cnt_5m
txn_cash_out_cnt_5m
txn_cash_in_cnt_3m
occu_h
prod_med
acct_type_num
tot_acct_num.1
tot_cash_val_5m
tot_cash_cnt_5m

Table 6: Features Selected Based on Total PCA Contributions (Threshold: 0.4)

5.4 Final Selection excluding High Correlated Features

Combining features selected using forward and backward selection, LASSO regression, individual principal component loading, and total contributions across all principal components, there are 43 unique features in Table 7.

We check the paired correlations among features in Figure 8.

Combined Feature Selection before Removing High Correlated Features
acct_type_num
avg_cash_5m*txn_cash_in_val_3m
cust_cdn_resident
cust_tenure*max_acct_num
geo_domicile_country_rate_high
geo_domicile_country_rate_low
geo_mail_addr_country_rate_high
geo_mail_addr_country_rate_low
in_person_onboard_f
in_person_visit_cnt*prod_none_prim_acct_cnt
max_acct_num
occu_h
prod_cda
prod_crc
prod_llc
prod_med
prod_mor
prod_prim_acct_cnt
prod_primary_cda_cnt
prod_primary_crc_cnt
prod_primary_llc_cnt*txn_cash_out_cnt_5m
prod_primary_ted_cnt*tot_cash_cnt_5m
prod_primary_ted_cnt*txn_cash_out_cnt_3m
prod_sav
prod_ted
tot_acct_num
tot_acct_num.1
tot_cash_cnt_5m
tot_cash_val_5m
tot_cash_val_5m*prod_primary_crc_cnt
tot_cross_border_cnt_12m
txn_cash_in_cnt_3m
txn_cash_in_cnt_5m
txn_cash_out_avg_3m*txn_wires_out_val_6m
txn_cash_out_cnt_3m
txn_cash_out_cnt_5m
txn_large_in_cash_val_10k_3m*txn_wires_in_low_cnt_3m
txn_large_out_cash_val_10k_3m*in_person_visit_cnt
txn_wires_in_cnt_12m*prod_primary_mor_cnt
txn_wires_out_high_cnt_12m*tot_cash_cnt_5m
txn_wires_out_high_cnt_12m*txn_wires_in_med_cnt_6m
txn_wires_out_med_avg_6m*txn_wires_in_high_cnt_12m
txn_wires_out_med_val_12m*txn_cross_border_in_avg_12m

Table 7: Combined Feature Selection before Removing High Correlated Features

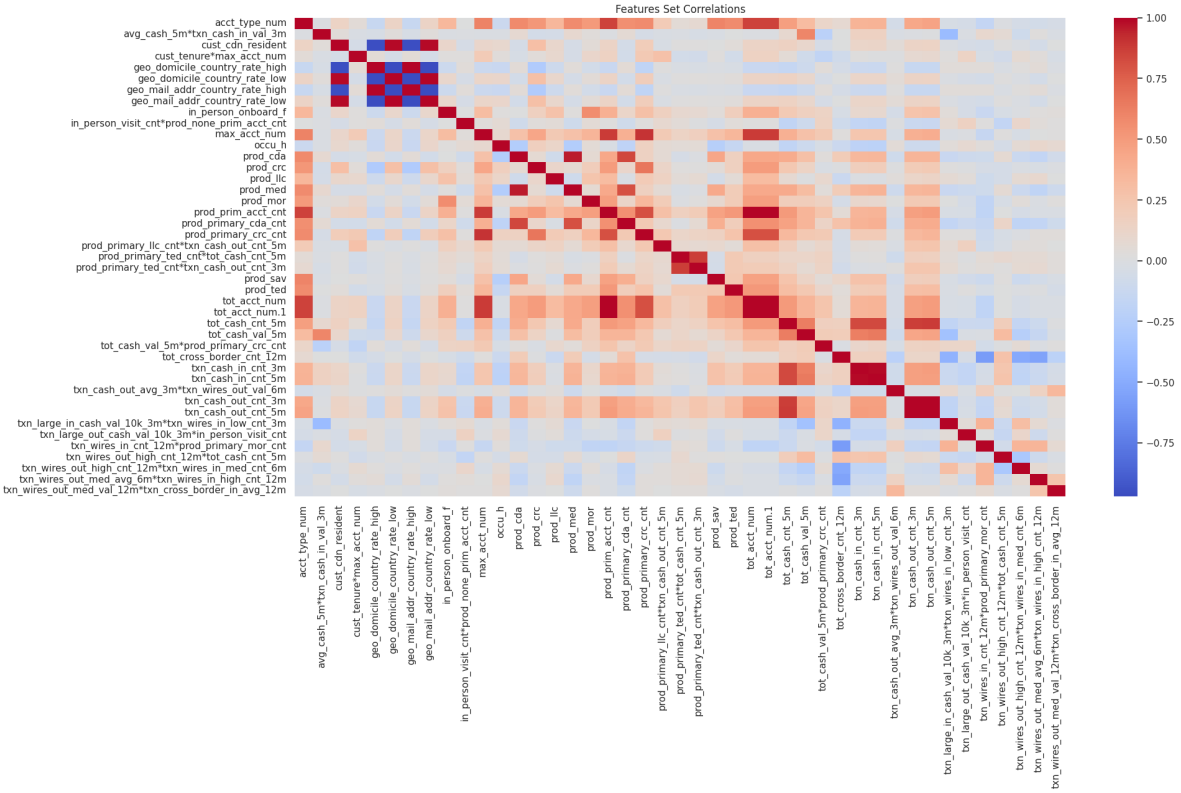

Figure 8: Correlation Heatmap before Removing High Correlated Features

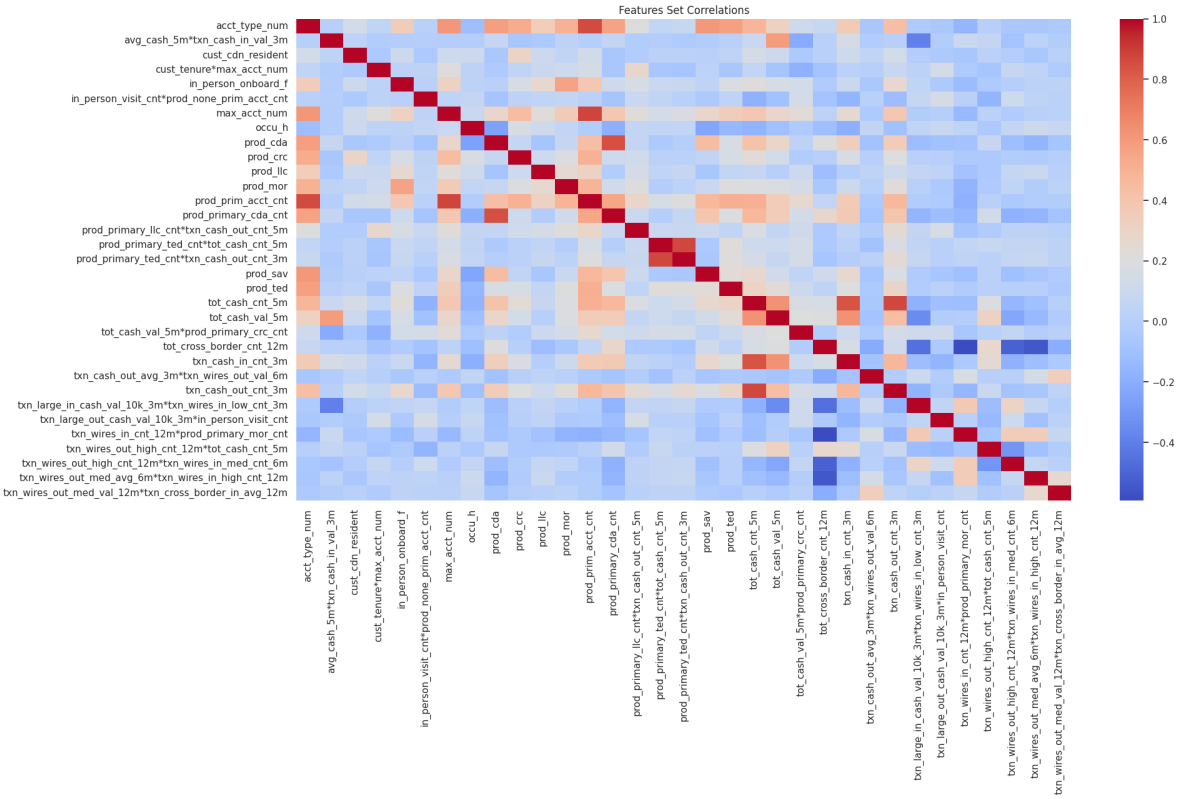

Figure 9: Correlation Heatmap after Removing High Correlated Features

To remove features with high correlation, a correlation matrix was computed for the selected features, and a threshold of 0.9 was applied to identify pairs of features with a strong linear relationship. We systematically identify and remove highly correlated features by iterating through features, marking those with high correlations for removal while skipping already-reviewed ones to avoid redundancy. This approach ensures that multicollinearity, which can negatively impact the interpretability and stability of machine learning models, is minimized. By reducing redundancy, the process helps create a more robust and parsimonious feature set, improving model performance and generalizability. Figure 9 shows the correlation heatmap after removing high correlated features. Table 8 is our final feature selection.

Final Feature Selection after Removing High Correlated Features
acct_type_num
avg_cash_5m*txn_cash_in_val_3m
cust_cdn_resident
cust_tenure*max_acct_num
in_person_onboard_f
in_person_visit_cnt*prod_none_prim_acct_cnt
max_acct_num
occu_h
prod_cda
prod_crc
prod_llc
prod_mor
prod_prim_acct_cnt
prod_primary_cda_cnt
prod_primary_llc_cnt*txn_cash_out_cnt_5m
prod_primary_ted_cnt*tot_cash_cnt_5m
prod_primary_ted_cnt*txn_cash_out_cnt_3m
prod_sav
prod_ted
tot_cash_cnt_5m
tot_cash_val_5m
tot_cash_val_5m*prod_primary_crc_cnt
tot_cross_border_cnt_12m
txn_cash_in_cnt_3m
txn_cash_out_avg_3m*txn_wires_out_val_6m
txn_cash_out_cnt_3m
txn_large_in_cash_val_10k_3m*txn_wires_in_low_cnt_3m
txn_large_out_cash_val_10k_3m*in_person_visit_cnt
txn_wires_in_cnt_12m*prod_primary_mor_cnt
txn_wires_out_high_cnt_12m*tot_cash_cnt_5m
txn_wires_out_high_cnt_12m*txn_wires_in_med_cnt_6m
txn_wires_out_med_avg_6m*txn_wires_in_high_cnt_12m
txn_wires_out_med_val_12m*txn_cross_border_in_avg_12m

Table 8: Final Feature Selection after Removing High Correlated Features

6 Machine Learning Training

6.1 Overview of Methods

For this analysis, we decided to use two widely recognized machine learning algorithms: **Random Forest (RF)** and **XGBoost**. Both algorithms belong to the category of ensemble methods, which combine multiple base models to improve prediction accuracy and robustness. However, RF and XGBoost employ different ensemble strategies—**bagging** and **boosting**, respectively—making them suitable for different scenarios. Our primary objective is to compare the performance of these two methods to determine which is optimal for the AML/ATF client risk rating task.

6.1.1 Random Forest (RF)

Random Forest is an ensemble learning algorithm based on **bagging** (Bootstrap Aggregation). In this approach:

- Multiple decision trees are trained on bootstrapped subsets of the dataset, with random sampling of features at each split.
- The final prediction is obtained by averaging (for regression) or majority voting (for classification) across all the trees.

Advantages:

- Robustness: Reduces the risk of overfitting by averaging predictions from multiple trees.
- Feature Handling: Handles high-dimensional data and multi-collinearity well.
- Interpretability: Provides feature importance scores, aiding interpretability.

These properties make Random Forest an excellent baseline model for tasks like AML/ATF classification, where high-dimensional and noisy data are common.

6.1.2 XGBoost

XGBoost (**eXtreme Gradient Boosting**) is an implementation of the gradient boosting algorithm, which is based on the **boosting** paradigm. In this approach:

- Weak learners (shallow decision trees) are sequentially trained, with each tree improving on the errors of the previous one.
- Model training focuses on minimizing a custom loss function using gradient descent, making it highly efficient and flexible.

Advantages:

- Performance: Known for its superior performance on structured data and imbalanced datasets.
- Flexibility: Supports various regularization techniques (L1, L2) to prevent overfitting.
- Scalability: Optimized for speed with parallel processing and distributed computing support.

XGBoost's ability to capture complex patterns and its robustness to imbalanced data make it a compelling choice for AML/ATF tasks.

6.2 Comparison Objective

The goal of this analysis is to train and evaluate both RF and XGBoost models using the same dataset and evaluate their performance using metrics such as F1-score, precision, recall, and ROC AUC. By comparing their results, we aim to determine the most suitable method for the client risk rating task, considering both classification accuracy and interpretability.

The **Random Forest Classifier** was chosen for its ability to handle high-dimensional data, capture non-linear relationships, and effectively manage imbalanced datasets. This model is particularly suitable for AML/ATF tasks where feature interaction and noise are common challenges.

The **XGBoost Classifier** was selected for its efficiency and flexibility. Its ability to focus on misclassified instances during training makes it well-suited for tasks with imbalanced datasets, such as AML/ATF client risk classification. Additionally, XGBoost's superior performance on structured data offers a competitive edge over other ensemble methods.

Both RF and XGBoost bring unique strengths to this task. RF excels in its interpretability and stability, while XGBoost offers enhanced performance and flexibility, particularly on imbalanced datasets. The subsequent sections will delve into the training process, hyperparameter tuning, and performance evaluation for each model, followed by a comparison of their results to identify the optimal method for this specific use case.

7 Hyperparameter Tuning and Threshold Management

7.1 Random Forest Model

7.1.1 Hyperparameters and Selection

The Random Forest model has several key hyperparameters that significantly affect its performance. To identify the optimal settings, we conducted a comprehensive **grid search** using **5-fold cross-validation**, with F1 score and ROC AUC as the scoring metric. The F1 score was chosen because it balances precision and recall, making it particularly suited for this imbalanced binary classification task where both false positives and false negatives have significant consequences.

The grid search explored the following hyperparameters:

- `n_estimators`: [25, 50, 100, 200, 300] - Number of decision trees in the forest.
- `max_depth`: [5, 10, 15, None] - Maximum depth of each tree, controlling model complexity.
- `min_samples_split`: [2, 5, 10] - Minimum number of samples required to split an internal node.
- `min_samples_leaf`: [1, 2, 4] - Minimum number of samples required to be at a leaf node.
- `max_features`: ['sqrt', 'log2', None] - Number of features considered for splitting at each node.

After grid search, the optimal hyperparameters by setting F1 score as the scoring metric were identified as:

- `n_estimators`: 300
- `max_depth`: 15
- `max_features`: `sqrt`
- `min_samples_split`: 5
- `min_samples_leaf`: 1

and the optimal hyperparameters by setting ROC AUC as the scoring metric were identified as:

- `n_estimators`: 300
- `max_depth`: 10
- `max_features`: `sqrt`
- `min_samples_split`: 2
- `min_samples_leaf`: 1

These hyperparameter values were chosen based on their ability to balance overfitting and underfitting, ensuring robust performance on the validation folds.

7.1.2 Threshold Management

The Random Forest algorithm inherently determines the decision thresholds based on class probabilities during the prediction process. No additional threshold optimization was required, as the model's internal mechanism effectively handled the classification of `high` and `non-high` clients.

7.2 XGBoost Model

7.2.1 Hyperparameters and Selection

The XGBoost model has several hyperparameters that require careful tuning to achieve optimal performance. Similar to the Random Forest model, we used **grid search** with **5-fold cross-validation**, with F1 score and ROC AUC as the scoring metric. The F1 score was chosen since it ensures the model is optimized for balanced precision and recall, critical for addressing the imbalance in the dataset.

The grid search explored the following hyperparameters:

- `n_estimators`: [100, 200, 300] - Number of boosting rounds.
- `learning_rate`: [0.01, 0.1, 0.2] - Step size for updating weights during training.
- `max_depth`: [3, 5, 7] - Maximum depth of a tree, controlling model complexity.
- `colsample_bytree`: [0.6, 0.8, 1.0] - Fraction of features sampled per tree split.
- `subsample`: [0.6, 0.8, 1.0] - Fraction of training samples used per boosting round.
- `scale_pos_weight`: [1, 5, 10] - Balances the positive class weight to handle imbalanced data.

After grid search, the optimal hyperparameters by setting F1 score as the scoring metric were identified as:

- `n_estimators`: 300
- `learning_rate`: 0.1
- `max_depth`: 5
- `colsample_bytree`: 0.8
- `subsample`: 0.8
- `scale_pos_weight`: 5

and the optimal hyperparameters by setting ROC AUC as the scoring metric were identified as:

- `n_estimators`: 100
- `learning_rate`: 0.1
- `max_depth`: 7
- `colsample_bytree`: 0.6
- `subsample`: 0.8
- `scale_pos_weight`: 1

These hyperparameter values were selected to balance the model's learning capacity, prevent overfitting, and address the imbalanced dataset.

7.2.2 Threshold Management

Similar to the Random Forest model, XGBoost manages thresholds internally during prediction by default. No manual optimization for thresholds was required for separating `high` and `non-high` clients.

7.3 Summary of Optimization

Both Random Forest and XGBoost underwent rigorous hyperparameter tuning using **5-fold cross-validation** with dual scoring metrics—**F1 score** and **ROC AUC**. These configurations highlight how the models adapt to different evaluation criteria:

- **F1 score optimization:** Focuses on balancing precision and recall, making it suitable for imbalanced datasets where both false positives and false negatives are critical.
- **ROC AUC optimization:** Prioritizes the overall ranking of predictions, ensuring robust discrimination across all probability thresholds.

After all, the F1 score was selected as the evaluation metric since it provides a more balanced measure.

Additionally, threshold management was handled internally by both algorithms, simplifying the classification process.

8 Performance Testing

8.1 Random Forest Model

To evaluate the performance of the Random Forest model, we compare four classification reports generated under different conditions:

- **With Feature Engineering and Feature Selection and ROC AUC as Scoring Metric**
- **With Feature Engineering and Feature Selection and F1 Score as Scoring Metric**
- **With Feature Engineering and Without Feature Selection and ROC AUC as Scoring Metric**
- **With Feature Engineering and Without Feature Selection and F1 Score as Scoring Metric**

8.1.1 With Feature Engineering and Feature Selection and ROC AUC as Scoring Metric (Best Performance Model)

The classification report for this configuration is presented in Table 9. Feature selection slightly reduces recall for the minority class (`high`), but the overall accuracy remains stable. The weighted average F1 score is 0.83, and the macro-average F1 score is 0.81.

Table 9: Classification Report for RF (With FS, ROC AUC)

Class	Precision	Recall	F1-score	Support
`non-high`	0.90	0.82	0.86	78
`high`	0.70	0.82	0.76	40
Accuracy		0.82 (118 samples)		
Macro Avg.	0.80	0.82	0.81	118
Weighted Avg.	0.83	0.82	0.83	118

8.1.2 With Feature Engineering and Feature Selection and F1 Score as Scoring Metric (Best Performance Model)

The classification report for this configuration is shown in Table 10. Feature selection with F1 optimization achieves the highest F1 score for the minority class (`high`), at 0.79, and maintains a robust macro-average F1 score of 0.84. The overall accuracy is also improved to 0.85.

Table 10: Classification Report for RF (With FS, F1)

Class	Precision	Recall	F1-score	Support
`non-high`	0.92	0.85	0.88	78
`high`	0.74	0.85	0.79	40
Accuracy		0.85 (118 samples)		
Macro Avg.	0.83	0.85	0.84	118
Weighted Avg.	0.86	0.85	0.85	118

8.1.3 With Feature Engineering and Without Feature Selection and ROC AUC as Scoring Metric (Best Performance Model)

The classification report for this configuration is presented in Table 11. Without feature selection, the model captures more relationships but risks redundancy, which slightly decreases precision for the minority class (`high`). The weighted average F1 score remains at 0.83.

Table 11: Classification Report for RF (No FS, ROC AUC)

Class	Precision	Recall	F1-score	Support
`non-high`	0.94	0.79	0.86	78
`high`	0.69	0.90	0.78	40
Accuracy		0.83 (118 samples)		
Macro Avg.	0.82	0.85	0.82	118
Weighted Avg.	0.86	0.83	0.83	118

8.1.4 With Feature Engineering and Without Feature Selection and F1 Score as Scoring Metric (Best Performance Model)

The classification report for this configuration is shown in Table 12. Without feature selection, the model achieves a slightly lower macro-average F1 score of 0.82 and an overall accuracy of 0.83, with no significant improvement in the F1 balance for the minority class.

Table 12: Classification Report for RF (No FS, F1)

Class	Precision	Recall	F1-score	Support
`non-high`	0.91	0.82	0.86	78
`high`	0.71	0.85	0.77	40
Accuracy		0.83 (118 samples)		
Macro Avg.	0.81	0.84	0.82	118
Weighted Avg.	0.84	0.83	0.83	118

8.1.5 Best Performance Selection

The model trained **With Feature Engineering and Feature Selection and F1 score as the scoring metric** achieves the best overall performance:

- Highest F1 score for the minority class (`high`) at 0.79.
- Balanced precision (0.74) and recall (0.85) for `high`.
- Robust macro-average and weighted-average F1 scores, ensuring stability across both classes.

By using feature selection, we reduce model complexity while retaining critical predictors. F1 optimization ensures balanced performance for both classes, addressing the class imbalance effectively.

8.1.6 Conclusion

Based on the comparison, **the Random Forest model With Feature Engineering and Feature Selection and F1 score as the scoring metric** is recommended as the optimal configuration for this binary classification task. With feature selection, the model provide more balanced recall and precision, but overall performance stays similar. It achieves the best trade-off between accuracy, precision, recall, and F1 score, making it suitable for AML/ATF risk classification.

8.2 XGBoost Model

To evaluate the performance of the XGBoost model, we compare four classification reports generated under different conditions:

- **With Feature Engineering and Feature Selection and ROC AUC as Scoring Metric**
- **With Feature Engineering and Feature Selection and F1 Score as Scoring Metric**
- **With Feature Engineering and Without Feature Selection and ROC AUC as Scoring Metric**
- **With Feature Engineering and Without Feature Selection and F1 Score as Scoring Metric**

8.2.1 With Feature Engineering and Feature Selection and ROC AUC as Scoring Metric (Best Performance Model)

The classification report for this configuration is shown in Table 13. While precision for the minority class (`high`) is moderate (0.62), the recall is relatively high at 0.85. The overall accuracy is 0.77, and the macro-average F1 score is 0.76.

Table 13: Classification Report for XGBoost (With FS, ROC AUC)

Class	Precision	Recall	F1-score	Support
`non-high`	0.90	0.73	0.81	78
`high`	0.62	0.85	0.72	40
Accuracy		0.77 (118 samples)		
Macro Avg.	0.76	0.79	0.76	118
Weighted Avg.	0.81	0.77	0.78	118

8.2.2 With Feature Engineering and Feature Selection and F1 Score as Scoring Metric (Best Performance Model)

The classification report for this configuration is presented in Table 14. This setup achieves balanced precision and recall for the minority class (`high`) with an F1 score of 0.70. However, the macro-average F1 score is lower than in the ROC AUC optimization scenario.

Table 14: Classification Report for XGBoost (With FS, F1)

Class	Precision	Recall	F1-score	Support
non-high	0.90	0.71	0.79	78
high	0.60	0.85	0.70	40
Accuracy		0.75 (118 samples)		
Macro Avg.	0.75	0.78	0.75	118
Weighted Avg.	0.80	0.75	0.76	118

8.2.3 With Feature Engineering and Without Feature Selection and ROC AUC as Scoring Metric (Best Performance Model)

The classification report for this configuration is shown in Table 15. Without feature selection, the model achieves slightly higher precision for the non-high class but sacrifices recall for the minority class (high), leading to a lower F1 score for that class.

Table 15: Classification Report for XGBoost (No FS, ROC AUC)

Class	Precision	Recall	F1-score	Support
non-high	0.83	0.79	0.81	78
high	0.63	0.68	0.65	40
Accuracy		0.75 (118 samples)		
Macro Avg.	0.73	0.73	0.73	118
Weighted Avg.	0.76	0.75	0.76	118

8.2.4 With Feature Engineering and Without Feature Selection and F1 Score as Scoring Metric (Best Performance Model)

The classification report for this configuration is presented in Table 16. The precision, recall, and F1 scores are generally balanced, but this setup fails to outperform the feature-selected configurations in terms of macro-average and weighted-average F1 scores.

Table 16: Classification Report for XGBoost (No FS, F1)

Class	Precision	Recall	F1-score	Support
non-high	0.87	0.76	0.81	78
high	0.62	0.78	0.69	40
Accuracy		0.76 (118 samples)		
Macro Avg.	0.74	0.77	0.75	118
Weighted Avg.	0.78	0.76	0.77	118

8.2.5 Best Performance Selection for XGBoost

The XGBoost model trained **With Feature Engineering and Feature Selection and ROC AUC as the scoring metric** achieves the best overall performance:

- High F1 score for the minority class (`high`) at 0.72.
- Balanced precision (0.62) and recall (0.85) for `high`.
- Robust macro-average and weighted-average F1 scores, ensuring stability across both classes.

8.3 Comparison Between Best RF and XGBoost Configurations

Comparing the best-performing configurations of Random Forest and XGBoost:

- **Random Forest (With FS, F1):** Achieves higher precision (0.74) and F1 score (0.79) for the minority class compared to XGBoost.
- **XGBoost (With FS, ROC AUC):** Achieves higher recall (0.85) for the minority class but lower F1 score (0.72) than Random Forest.
- Overall, the Random Forest model demonstrates a better trade-off between precision and recall, making it the preferred model for this task.

8.4 Conclusion

Based on the comparison, the **Random Forest model With Feature Engineering and Feature Selection and F1 score as the scoring metric** remains the optimal choice. It provides superior balance between accuracy, precision, recall, and F1 score, ensuring effective classification for both `high` and `non-high` classes.

9 Feature Importance and Model Interpretation

9.1 Feature Importance Analysis

The feature importance plot for the selected Random Forest model (with feature selection and F1 score as the scoring metric) is shown in Figure 10. The importance of a feature is measured by the average decrease in impurity it contributes when used for splitting decision trees. Features with higher importance values contribute more significantly to the model's predictions. The feature importance of the variables selected through feature selection appears strong, as shown in the following figures.

Key observations from the feature importance plot include:

- `txn_large_out_cash_val_10k_3m*in_person_visit_cnt`: This interaction feature has the highest importance, indicating that frequent in-person visits combined with large cash transactions are highly predictive of the client's risk rating. This aligns with domain knowledge, as such patterns could indicate suspicious financial behavior.
- `cust_tenure*max_acct_num`: The interaction between customer tenure and the maximum number of accounts suggests that long-standing customers with multiple accounts may exhibit distinct risk patterns, possibly due to prolonged transaction histories.

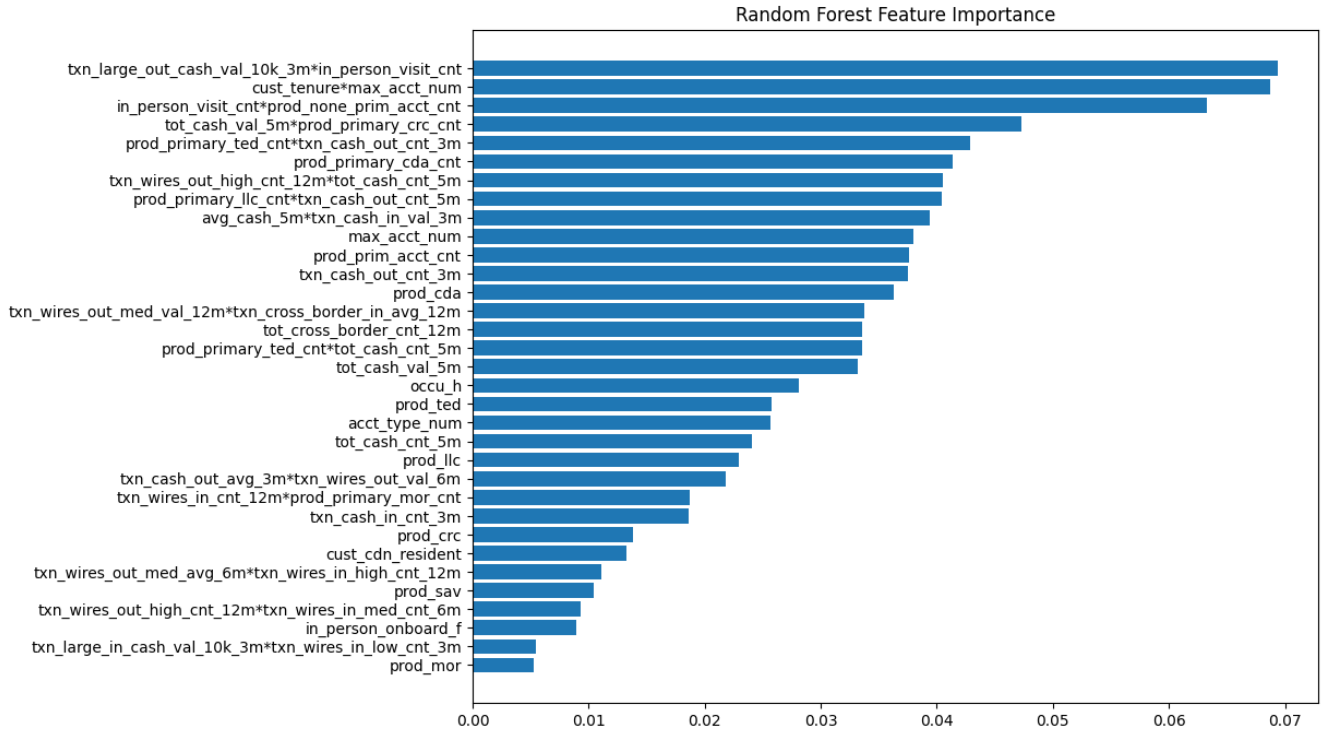

Figure 10: Feature Importance of Random Forest with Feature Selection and F1 Score as Metric

- `prod_primary_ted_cnt*txn_cash_out_cnt_3m`: The combination of product-based transaction counts and recent cash outflows is another significant indicator, hinting at the interplay between product utilization and cash activity in determining risk.

9.2 Counter-Intuitive Behavior

While most important features align with expectations, certain features display counter-intuitive behavior:

- `occu_h`: Occupation-related features such as `occu_h` (indicating higher-risk occupations) appear lower in the importance ranking than expected. This may suggest that other financial behavior patterns outweigh occupation when predicting client risk ratings.
- `cust_cdn_resident`: The binary indicator for Canadian residency shows moderate importance. However, a non-resident flag might intuitively correlate with higher risk due to reduced traceability, yet the model may attribute higher importance to transaction patterns rather than residency.

Counter-intuitive behavior may arise due to feature interaction effects, where certain features may interact with others in non-linear ways, obscuring their individual contribution to the model's predictions.

To address such issues, we propose:

- **Further Data Exploration:** Examine the distributions and relationships between counter-intuitive features and other variables to understand underlying patterns.
- **Incorporating Domain Knowledge:** Adjust the dataset or feature engineering process to ensure

that critical variables like occu_h receive appropriate weight.

- **Model Constraints:** Apply regularization techniques or domain-specific constraints to emphasize the importance of certain features.
- **Explainability Techniques:** Use SHAP (Shapley Additive Explanations) values to better understand the contribution of each feature at the individual prediction level, providing more granular insights into the model's behavior.

9.3 Conclusion

The Random Forest model demonstrates strong alignment with domain knowledge in its feature importance rankings, emphasizing transactional patterns and interaction effects. However, counter-intuitive results highlight the need for careful data exploration and incorporation of domain expertise to further refine the model's predictions.

10 Other Issue Consideration

10.1 Effect of Feature Selection and Cross Terms

One significant issue observed during the modeling process was the performance and efficiency of the Random Forest and XGBoost models when trained without feature selection or cross terms. Without these preprocessing steps, the models exhibited:

- **Lower performance:** The classification reports and evaluation metrics showed reduced precision, recall, and F1 scores compared to models trained with feature selection and cross terms.
- **High computational cost:** Training times were significantly longer due to the large number of features, many of which contributed minimally or not at all to the final predictions. For instance, the Random Forest model without feature selection took approximately 50% more time to train compared to the version with feature selection.

To address these issues:

- A **feature selection process** was implemented to retain the most important predictors while eliminating noise and redundant features. This not only improved performance metrics (e.g., F1 score and ROC AUC) but also reduced overfitting and sped up model training.
- **Cross terms** were created for certain features to capture complex interactions, which proved critical for enhancing the model's ability to identify subtle patterns indicative of high-risk behavior.

10.2 Threshold Management

Initially, efforts were made to optimize decision thresholds manually to balance false positives and false negatives. However, it was later observed that both the Random Forest and XGBoost algorithms effectively manage thresholds internally during prediction. As a result, manual threshold optimization was deemed unnecessary, simplifying the overall workflow.

10.3 Interpretability and Domain Knowledge Integration

While the models demonstrated strong performance, interpretability was a challenge due to the inherent complexity of ensemble methods like Random Forest and XGBoost. To address this:

- Feature importance rankings were analyzed to ensure that the top predictors aligned with domain knowledge in AML/ATF.
- Instances of counter-intuitive feature behavior, such as the lower importance of high-risk occupation (`occu_h`), were investigated. These were attributed to data distribution and the indirect capture of such information via transactional features.

Future enhancements could involve the use of explainability tools, such as SHAP (Shapley Additive Explanations), to provide granular insights into individual predictions and validate feature contributions against domain expectations.

10.4 Scalability and Efficiency

The scalability of the modeling approach was another important consideration:

- The feature selection process not only improved prediction accuracy but also significantly reduced training time, making the models more scalable for larger datasets.
- Hyperparameter tuning was conducted using **grid search** with 5-fold cross-validation, which, while comprehensive, was computationally expensive. Future iterations could explore **Bayesian optimization** or **random search** to enhance efficiency.

10.5 Conclusion

Addressing these issues through feature selection, cross terms, class balancing, and interpretability measures resulted in a robust and efficient modeling pipeline. These adjustments highlight the importance of tailoring machine learning workflows to specific datasets and domain requirements, ensuring both practical and theoretical challenges are effectively mitigated.