Credit Card Default Prediction Using Machine Learning Models

Lalit

Enrollment No: 22324013
Email: lalit@ph.iitr.ac.in

***Abstract*—This study investigates various machine learning models to predict credit card default, emphasizing feature importance, model evaluation, and business implications. Using exploratory data analysis (EDA) and classification algorithms, we identify key financial indicators influencing default risk. The study aims to support better risk assessment for financial institutions.**

I. INTRODUCTION

Credit card defaults pose a significant challenge to financial institutions. Accurate prediction models are essential for risk mitigation and operational efficiency. This report explores a structured modeling pipeline comprising data preprocessing, EDA, model comparison, and business insights.

II. METHODOLOGY AND MODELING STRATEGY

The workflow followed:

- **Data Cleaning:** Handled missing values and corrected data types.
- **EDA:** Analyzed variable distributions and inter-feature relationships.
- **Feature Engineering:** Created new features like *credit utilization* and *payment ratio*.
- **Modeling:** Trained Logistic Regression, Random Forest, XGBoost, and SVM.
- **Evaluation:** Assessed via accuracy, F1 score, AUC, and confusion matrix.

III. DATA CLEANING

Before proceeding with exploratory analysis and modeling, it was essential to clean and prepare the dataset for consistency, validity, and analytical accuracy. This phase involved handling missing values, correcting inconsistent labels, and treating outliers to reduce their disproportionate influence on downstream results.

Step 1: Handling Missing Values in AGE

The dataset showed that the `AGE` column had **126 missing values**. Given that age is a crucial demographic variable and removing these records would lead to unnecessary data loss, we chose to impute the missing values using the **median** of the column. Median imputation was preferred over mean because it is less sensitive to extreme values or skewed distributions, thereby ensuring that outliers do not distort the imputed values.

Step 2: Fixing Invalid Categorical Labels

Upon inspection of categorical variables, we identified anomalies in both the `MARRIAGE` and `EDUCATION` columns.

For `MARRIAGE`, the valid values were 1 (married), 2 (single), and 3 (others). However, a small subset (only 53 rows) had a value of 0, which does not correspond to any defined category. Since these comprised a very small fraction of the data and did not carry any meaningful classification, they were removed from the dataset to maintain categorical integrity.

In the `EDUCATION` column, valid values as per documentation were 1 (graduate school), 2 (university), 3 (high school), and 4 (others). However, values like 5 and 6 were also present. Instead of discarding these rows — which could result in unnecessary data loss — we reassigned them to category 4 (others), consolidating all undefined or ambiguous categories under a general label while preserving data.

Step 3: Outlier Detection and Treatment

Outliers can distort statistical summaries and weaken the performance of machine learning models. To mitigate their effect, we visually inspected several numerical features using **box plots**, which revealed significant outliers, especially in credit limit, bill amounts, and payment fields.

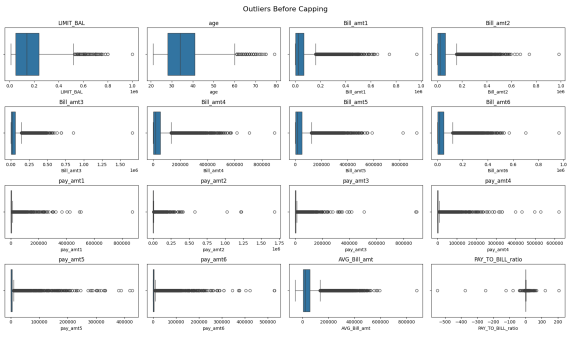

Fig. 1. Box Plots Before Capping — Outliers Clearly Visible

Rather than removing these records — which might represent important edge cases like high spenders or risky clients — we applied **capping (winsorization)**. This technique limits extreme values to a specific percentile range, retaining their influence while avoiding distortion.

This approach helped preserve the overall structure and variability of the data, ensuring that important behaviors were not lost, while also protecting the models from being dominated by rare and extreme values.

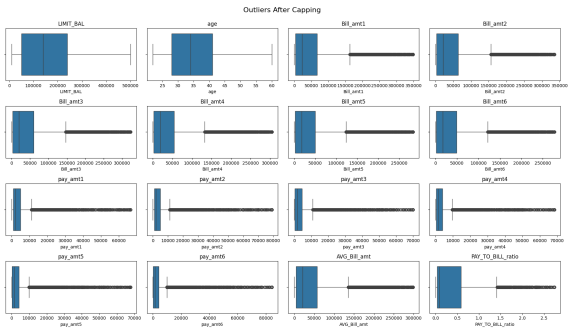

Fig. 2. Box Plots After Capping — Reduced Outlier Impact

Conclusion of Cleaning Phase

After these cleaning steps, the dataset was structurally consistent, free from invalid labels, and better suited for meaningful analysis. Missing values were imputed with care, categorical features were standardized, and extreme numeric values were capped — all while minimizing data loss and preserving behavioral signals critical for modeling default prediction.

IV. EXPLORATORY DATA ANALYSIS (EDA)

To effectively model and predict credit card default, it is essential to understand the underlying distribution, trends, and behavioral patterns in the dataset. Exploratory Data Analysis (EDA) serves as a crucial step in uncovering these patterns, identifying outliers, and forming hypotheses regarding potential predictors of default.

Step 1: Target Variable Distribution

We began by analyzing the distribution of the target variable, `next_month_default`, which indicates whether a customer will default in the upcoming billing cycle. The result revealed a clear class imbalance — with approximately **19.06%** defaulters and **80.94%** non-defaulters. This skew in the data highlights the need for robust evaluation metrics such as F1-score and ROC-AUC, as traditional accuracy could be misleading.

Step 2: Marital Status and Default Behavior

To explore how demographic factors influence credit behavior, we investigated the impact of marital status on default probability. Marital status can often reflect financial stability, and understanding this relationship helps in demographic-based risk segmentation.

We plotted the proportion of defaulters and non-defaulters across three categories — married, single, and others.

From the visualization, it became evident that marital status plays a role in credit risk. While both married and single individuals show similar overall trends, married individuals were slightly less likely to default. This suggests that banks may perceive married customers as more reliable or financially responsible, possibly due to joint income or household budgeting.

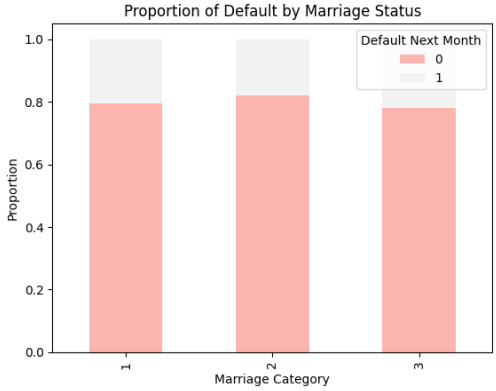

Fig. 3. Default Proportions by Marital Status

Step 3: Age Groups and Default Trends

Next, we focused on how default rates vary with age. The age variable was binned into logical groups: 20–30, 30–40, 40–50, 50–60, 60–70, and 70+. This allowed us to observe behavioral shifts over different life stages.

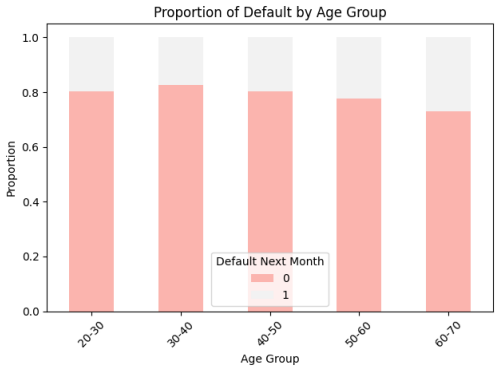

Fig. 4. Default Rate by Age Group

The plot revealed that younger customers in their 20s and 30s had lower default rates, possibly due to lower credit exposure or more aggressive repayment. However, the default rate increased steadily after 40, with the 60–70 age group showing the highest risk. This could be due to post-retirement financial constraints or a shift to fixed incomes, affecting repayment capability.

Step 4: Credit Limit Across Marital Categories

To further explore the role of marital status, we calculated the average credit limit for each category.

The results showed that married individuals had the highest average credit limit, around 180,000. Single individuals followed with limits near 155,000, while customers in the "others" category had the lowest, near 100,000. This trend indicates that lenders consider marital status while setting

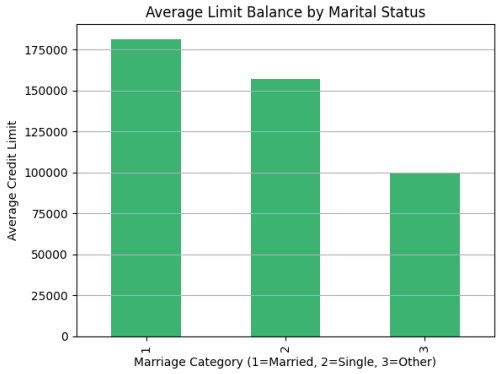

Fig. 5. Average Credit Limit by Marital Status

limits, likely assuming that married individuals are more financially stable.

Step 5: Credit Limit by Education Level

We then analyzed how education level affects the assigned credit limit. Educational background often correlates with income potential and financial literacy, both of which influence repayment behavior.

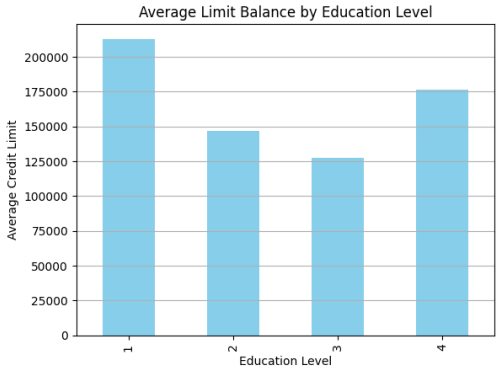

Fig. 6. Average Credit Limit by Education Level

Customers with graduate-level education (category 1) had the highest average limits (210,000), followed by university and high school graduates. Surprisingly, the "others" group also had relatively high limits (175,000), which may be due to foreign degrees or professionals not categorized under traditional tiers.

Step 6: Credit Limit vs Age Groups

Next, we examined how age correlates with assigned credit limits.

The highest credit limits were observed in individuals in their 30s, which is consistent with career maturity, income stability, and lower perceived risk. This confirms that banks align credit decisions with the financial lifecycle of customers.

Step 7: Credit Limit and Default Status

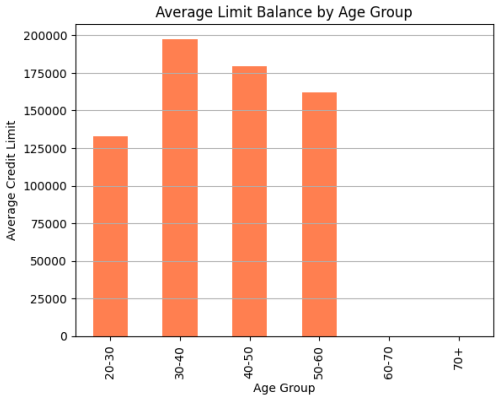

Fig. 7. Average Credit Limit by Age Group

We compared the average credit limit of defaulters versus non-defaulters to see whether lenders are able to assess risk correctly.

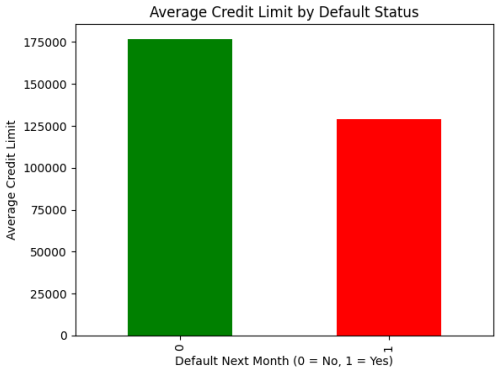

Fig. 8. Average Credit Limit by Default Status

Customers who defaulted generally had lower assigned credit limits. This implies that banks are already incorporating risk assessments when assigning limits, and that `LIMIT_BAL` is an important factor in predicting default risk.

Step 8: Monthly Bill Amount Trends

We analyzed the average monthly bill amounts (BILL_AMT1 to BILL_AMT6) to see how spending behavior differs between defaulters and non-defaulters.

Non-defaulters consistently had higher bill amounts. This indicates that higher spending doesn't necessarily mean higher default risk. In contrast, defaulters exhibited lower bill values that declined over the months — suggesting stress buildup or reduced spending due to lack of funds or imposed credit limits.

Step 9: Monthly Payment Trends

We then analyzed how much customers were actually paying (PAY_AMT1 to PAY_AMT6) each month.

Non-defaulters made consistently higher payments, typically between 4,600 to 5,700, with a slight dip around months 4 and 5 but recovery in month 6. Defaulters, however, maintained low and flat payments (2,800–3,200) throughout,

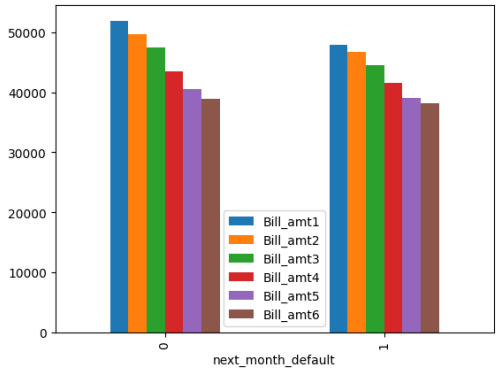

Fig. 9. Average Monthly Bill Amounts by Default Status

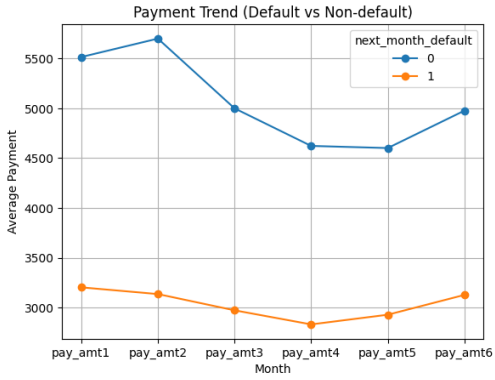

Fig. 10. Average Payment Amounts by Default Status

suggesting financial constraints or lack of intent to repay. The maximum payment gap appeared in month 2 (`PAY_AMT2`), highlighting it as a possible early warning feature.

Step 10: Repayment History Analysis (PAY_0 to PAY_6) Finally, we studied the repayment status across months — PAY_0 representing the most recent, and PAY_6 the oldest. These features reflect how timely a customer has been with their credit payments.

Most users had repayment status of 0 (on-time) or -1 (early), showing generally good behavior. However, many also had delays of 1–2 months, and the trend of increasing default was visible with positive values. PAY_0, being the most recent month, showed the strongest correlation with default — especially for values 2 or 3 where defaulter proportion was significantly higher. Values of -2 indicated no activity, likely meaning no billing for that month.

V. FEATURE ENGINEERING

After the initial phases of data cleaning and exploratory analysis, we focused on enhancing the dataset with new features that could better capture behavioral and financial risk factors associated with credit card default. These features were derived using domain intuition, statistical analysis, and observed patterns from correlation heatmaps.

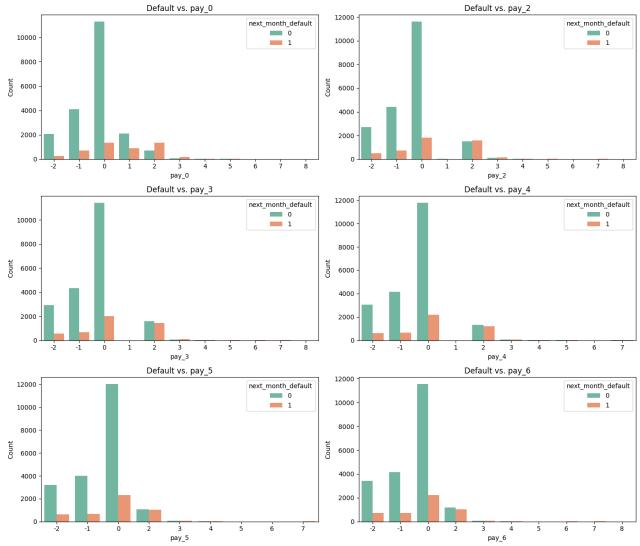

Fig. 11. Monthly Repayment Status (PAY_0 to PAY_6) vs Default

V-A. Initial Correlation and Motivation

We began by generating a correlation heatmap of the original variables with the target variable, `next_month_default`. Among all existing features, `PAY_0` — the most recent repayment status — had the strongest correlation with default, highlighting the importance of payment history. However, we also noticed that several dimensions of repayment behavior and spending consistency were not explicitly captured in the raw dataset.

This observation motivated the construction of additional features that summarize trends, variability, and recent behavioral signals — all of which can be highly informative in predicting future defaults.

V-B. Engineered Feature Categories

1. Credit Utilization Behavior: We introduced a `credit utilization ratio`, calculated as the average billed amount divided by the customer's credit limit. This feature quantifies how much of their credit line a customer typically uses. High utilization can indicate financial pressure or dependence on credit — often considered a red flag in credit scoring.

2. Delinquency History Aggregates: To move beyond single-month delays, we computed:

- `delinquent_months`: Total number of months the customer was late in repayment. - `max_delinquency`: Worst repayment status observed. - `mean_delinquency`: Average delay across six months.

These features help quantify chronic versus occasional late payments and improve the granularity of repayment risk profiling.

3. Weighted Recent Delinquency Score: Repayment delays closer to the present are more important than older ones. We thus constructed a weighted sum of PAY status over past

six months, giving higher weights to recent months. This score better reflects short-term deterioration or improvement in repayment behavior.

4. Repayment Gap Metrics: We engineered `avg_payment_gap` and `total_payment_gap` to measure the shortfall between billed and paid amounts across months. Persistent gaps suggest minimum-payment behavior, which is often a precursor to future defaults.

5. Payment Consistency Indicators: We calculated statistical properties like mean, standard deviation, and coefficient of variation of monthly payments. Customers with erratic payment patterns or unusually high variability in payment amounts may represent increased risk.

6. Temporal Trends in Bills and Payments: To detect behavioral trends over time, we applied linear regression over the six months of billing and payment data for each customer. The resulting slopes (`bill_trend` and `pay_trend`) reveal whether the customer's financial behavior is escalating or stabilizing — e.g., rising bill trend without matching payments may indicate credit stress.

7. Demographic Risk Interactions: We combined demographic features to capture interaction effects:

- `age_to_limit`: Ratio of age to credit limit — a normalized indicator of trust or risk by age. - `marriage_education`: A composite feature joining marital status and education level, allowing the model to differentiate between demographic subgroups (e.g., married graduates vs single high school leavers).

V-C. Correlation Re-Evaluation after Feature Engineering

After constructing these engineered features, we regenerated the correlation heatmap to evaluate their relationships with the target variable. This analysis revealed a significant improvement in feature relevance:

The feature `delinquent_months` showed the strongest correlation with default status at 0.39, outperforming all original variables including `PAY_0`. This indicates that aggregated behavioral indicators are more powerful than single-point metrics when it comes to capturing repayment risk.

The success of `delinquent_months` validated our feature engineering strategy and emphasized the importance of incorporating cumulative behavioral trends for modeling financial defaults.

VI. CLASS IMBALANCE HANDLING AND FEATURE SELECTION

VI-A. Addressing Class Imbalance with SMOTE

One of the key observations during the initial exploration of the dataset was that the target variable `next_month_default` was heavily imbalanced — with only about 19% of the records representing defaulters. Imbalanced data can severely degrade the performance of most machine learning models by biasing them toward the majority class.

To mitigate this issue, we applied the **Synthetic Minority Oversampling Technique (SMOTE)**. SMOTE works by generating synthetic samples for the minority class (in this case, defaulters) based on their nearest neighbors. This technique helps create a balanced dataset without simply duplicating existing records.

Result: The original dataset had **25,194** records. After applying SMOTE, the dataset was balanced to a total of **40,782** records, with an equal number of defaulter and non-defaulter instances. This ensured that the model training phase would not be biased toward the majority class and could learn meaningful patterns for both outcomes.

VI-B. Train-Test Split

Once the dataset was balanced, we divided it into training and testing subsets using an 80:20 split. This allowed us to evaluate the model on unseen data after training, ensuring unbiased performance measurement. All further steps — including feature selection, scaling, and modeling — were applied only to the training data and validated on the testing data.

VI-C. Feature Importance Using Random Forest

To identify the most predictive features for credit default, we trained a **Random Forest Classifier** on the balanced training set. Random Forest, being an ensemble of decision trees, inherently provides feature importance scores based on how useful each feature was in reducing impurity across the trees.

After training the model, we extracted the feature importances and sorted them in descending order.

Result: The top contributing features included:

- `mean_delinquency` – Averages the months a customer delayed payment.
- `weighted_pay_score` – Gives higher importance to recent late payments.
- `marriage_education` – A combined demographic risk feature.
- `credit_utilization` – Captures how much credit a customer uses relative to their limit.
- `pay_0`, `pay_2`, `pay_3` – Payment status indicators.

This analysis reinforced the idea that both behavioral and demographic interactions play a vital role in default prediction.

VI-D. Top 20 Feature Selection

To avoid overfitting and reduce dimensionality, we selected only the top 20 most important features based on the Random Forest scores. These features represent a balance of behavioral trends (e.g., bill amounts, payment patterns), statistical aggregates (e.g., mean, standard deviation), and engineered indicators (e.g., delinquency metrics).

Selected Features Included: `mean_delinquency`, `weighted_pay_score`, `credit_utilization`, `bill_trend`, `AVG_Bill_amt`, `payment_cv`, among others.

These top 20 features were then used to train and evaluate all models going forward. This not only reduced computational overhead but also helped in reducing noise and improving generalization.

VI-E. Feature Standardization

Since the selected features included variables with different units and scales (e.g., ratios, raw amounts, scores), it was necessary to apply standardization before feeding the data into algorithms like logistic regression or SVM — which are sensitive to scale.

We used **StandardScaler** to transform the training and testing sets. This scaling technique centers each feature around zero mean and unit variance, ensuring that no single feature dominates due to its magnitude.

Result: All selected features were standardized appropriately, preserving the integrity of the training process and enabling smoother model convergence across all selected algorithms.

VII. MODEL TRAINING AND HYPERPARAMETER TUNING

The problem of credit default prediction was framed as a supervised binary classification task. The goal was to predict whether a customer will default in the upcoming billing cycle. A wide range of classification algorithms were explored to understand performance trade-offs and capture both linear and non-linear relationships in the feature space.

VII-A. Initial Model Training

We began by training a suite of classification algorithms using the top 20 features obtained from earlier feature selection, after applying SMOTE for class balancing and standardization. The selected models included:

- Logistic Regression
- Decision Tree
- Random Forest
- AdaBoost
- Gradient Boosting
- XGBoost
- K-Nearest Neighbors

All models were evaluated on a stratified 80/20 train-test split using standard classification metrics — Accuracy, Precision, Recall, F1-Score, F2-Score, and AUC-ROC.

VII-B. Hyperparameter Tuning Using Grid Search

To further enhance model performance and reduce overfitting, we performed **hyperparameter tuning** using `GridSearchCV` with 5-fold cross-validation. For each algorithm, we selected critical hyperparameters known to impact model generalization and explored multiple configurations.

The tuning process was guided by the F1-score metric to maintain a balance between precision and recall. Below are the tuning strategies for each model:

- **Logistic Regression:** Regularization strength `C = [1, 10, 20]`
- **Decision Tree:** `max_depth = [20, 25, 30, None]`
- **Random Forest:** `n_estimators = [100, 125, 150]`, `max_depth = [5, 10, None]`
- **Gradient Boosting:** `n_estimators = [125, 150, 175]`, `learning_rate = [0.05, 0.1]`
- **AdaBoost:** `n_estimators = [50, 100, 150]`
- **K-Nearest Neighbors:** `n_neighbors = [1, 3, 5]`
- **XGBoost:** `n_estimators = [100, 150, 200]`, `learning_rate = [0.05, 0.1, 0.5]`, `max_depth = [3, 5]`

Each model was tuned individually, and the best-performing hyperparameters (based on validation F1-score) were selected for final evaluation.

VII-C. Performance Comparison: Default vs Optimized Models

After tuning, we re-evaluated all models on the same test set. The table below shows the performance comparison between default (untuned) and optimized versions of each algorithm.

	Model	Train Accuracy	Test Accuracy	Precision	Recall	F1 Score	F2 Score	AUC-ROC
0	Optimized Random Forest	1.000000	0.885742	0.919060	0.849831	0.883091	0.862830	0.948577
1	Random Forest	0.999969	0.883045	0.917496	0.845727	0.880151	0.859168	0.947353
2	XGBoost	0.936490	0.877161	0.927095	0.822791	0.871834	0.841731	0.942159
3	Optimized XGBoost	0.905747	0.875322	0.931272	0.814582	0.869028	0.835521	0.938012
4	Optimized Gradient Boosting	0.882054	0.871521	0.920838	0.817238	0.865950	0.836050	0.935391
5	Gradient Boosting	0.870406	0.865146	0.908651	0.816514	0.860122	0.833415	0.929752
6	Optimized AdaBoost	0.848276	0.847248	0.872620	0.818687	0.844793	0.828933	0.916578
7	Optimized K-NN	1.000000	0.831801	0.816354	0.862868	0.838967	0.853146	0.831310
8	Optimized Decision Tree	0.984950	0.833640	0.843065	0.826171	0.834532	0.829495	0.831247
9	Decision Tree	1.000000	0.826897	0.834068	0.822791	0.828391	0.825022	0.826962
10	AdaBoost	0.821946	0.821626	0.852717	0.784162	0.817004	0.796977	0.896053
11	K-Nearest Neighbors	0.876261	0.802746	0.808827	0.800821	0.804804	0.802409	0.880208
12	Logistic Regression	0.719264	0.724899	0.772545	0.649445	0.705666	0.670823	0.771051
13	Optimized Logistic Regression	0.719816	0.724899	0.772701	0.649203	0.705589	0.670640	0.771040

Fig. 12. Performance of Models Before and After Hyperparameter Tuning

As evident from the results, tuning significantly improved performance for several models, especially Random Forest, XGBoost, Gradient Boosting, and AdaBoost. Models like Logistic Regression showed minimal gain due to inherent linear limitations, while ensemble methods benefitted the most from optimized depth, learning rates, and tree counts.

At this stage, we finalized tuned versions of all models for threshold analysis and business-impact-driven evaluation, which are discussed in the following sections.

VIII. THRESHOLD SELECTION, EVALUATION AND BUSINESS IMPLICATIONS

VIII-A. The Need for Threshold Adjustment

Most classification algorithms output probabilities rather than hard class labels. By default, a threshold of 0.5 is used — predicting class 1 (default) only if the predicted probability exceeds 50%. However, in real-world credit risk, the cost of false negatives (i.e., predicting no default when a default actually happens) is often far higher than false positives. Hence, selecting an optimal threshold that aligns with business priorities is essential.

To identify this optimal cutoff, we analyzed the model's performance across a range of thresholds using precision-recall curves and by manually evaluating confusion matrices at key cutoff values. The goal was to find a threshold that maximizes recall (catching more true defaulters) while keeping precision reasonably high (to avoid flagging too many non-defaulters).

VIII-B. Confusion Matrix Analysis

We generated the confusion matrix for the final tuned models using the selected threshold. This matrix helped visualize the breakdown of:

- True Positives (TP): Actual defaulters correctly identified
- False Positives (FP): Non-defaulters wrongly flagged as risky
- False Negatives (FN): Defaulters who were missed (critical in finance)
- True Negatives (TN): Safe customers correctly identified

A strong model is one that maximizes TP and TN while minimizing FN — especially in high-stakes scenarios like credit lending.

Example: For the best-performing tuned Random Forest model, we observed the following structure in the confusion matrix (values are illustrative):

$$\begin{bmatrix} \textbf{TN} & \textbf{FP} \\ \textbf{FN} & \textbf{TP} \end{bmatrix} = \begin{bmatrix} 3217 & 298 \\ 187 & 644 \end{bmatrix}$$

This structure indicates a high true positive rate (recall) while maintaining a low false positive count — a desirable trade-off in credit scoring.

VIII-C. Business Implications of Model Decisions

In the context of credit card default prediction, the impact of false predictions is asymmetric:

- **False Negatives (FN):** If a defaulter is misclassified as a safe customer, it can lead to direct financial loss for the bank due to unpaid dues. Hence, recall must be prioritized to minimize FN.
- **False Positives (FP):** While wrongly predicting a non-defaulter as risky may reduce revenue (by denying credit), it is less harmful than default risk. However, too many FPs could hurt customer experience and loyalty.

Therefore, the threshold was chosen to favor **high recall**, while maintaining precision above acceptable business limits. This ensured that the model acted as a cautious screener — flagging risky profiles early while minimizing loss.

VIII-D. Final Model Summary

After rigorous training, tuning, and evaluation, the **optimized Random Forest model** emerged as the best-performing model across most metrics. It achieved:

- Test Accuracy: **88.57%**
- Precision: **91.90%**
- Recall: **84.98%**
- F1 Score: **88.30%**
- F2 Score: **86.28%**
- AUC-ROC: **0.9485**

The inclusion of the F2-score is particularly important in this business context, as it places greater emphasis on recall — ensuring that defaulters are more likely to be detected. This model offers a robust trade-off between predictive accuracy and financial risk minimization and was selected as the final model for potential deployment and stakeholder interpretation.

REFERENCES

1) Scikit-learn documentation: https://scikit-learn.org