Project Report

Credit Card Default Prediction

Using Classification and Risk-Based Techniques

Submitted to: Finance Club, IIT Roorkee

Name	:	Krish Singla
Branch	:	CSE 2nd Year
Enrollment	:	23114050
Email	:	krish_s@cs.iitr.ac.in

IIT Roorkee

Contents

1 Introduction

1.1 What is Credit Card Default and Why is it Important?

Credit card default occurs when a customer fails to repay their credit card balance, posing a financial risk to banks and lending institutions. Accurately predicting potential defaulters helps reduce losses, manage credit exposure, and enable proactive interventions.

By analyzing historical repayment behavior, credit usage, and demographic data, we aim to build a machine learning model that identifies high-risk customers in advance—facilitating smarter lending decisions and strengthening financial stability.

1.2 Business Objective

The goal of this project is to help financial institutions reduce credit risk by identifying customers likely to default on credit card payments in the upcoming month. Early detection of such defaulters allows lenders to minimize losses, optimize credit distribution, and preserve portfolio health.

To achieve this, we employ classification-based machine learning models enhanced with risk-aware strategies, including threshold tuning, recall-oriented metrics like F-score, and targeted feature engineering based on financial indicators such as credit utilization, delay streaks, and payment-to-bill ratios.

The model emphasizes capturing high-risk defaulters, even at the cost of some false positives—aligning with the business priority of minimizing missed defaults while maintaining efficiency.

1.3 Given Data

The dataset comprises historical credit and repayment information for a large number of credit card customers. Each row represents a unique customer, with columns capturing demographic details and financial behavior over a six-month period.

Key features include:

- **LIMIT_BAL** – Assigned credit limit.
- **SEX, EDUCATION, MARRIAGE, AGE** – Demographic attributes.
- **BILL_AMT1 to BILL_AMT6** – Monthly bill amounts for the last six months.
- **PAY_AMT1 to PAY_AMT6** – Monthly repayment amounts.
- **PAY_0 to PAY_6** – Payment delay status per month (e.g., -1 = early, 0 = on time, 1 = one month late, etc.).
- **default payment next month** – Target variable (1 = default, 0 = no default).

The training set contains labeled data used to build and validate the model. The test set includes similar customer records without the target label, and the objective is to accurately predict their default status.

1.4 Problem Statement

The goal of this project is to build a machine learning model that predicts whether a credit card customer will default on their payment in the upcoming month. Predictions will be based on historical behavior such as credit usage, repayment patterns, and demographic attributes.

This is a **binary classification** problem focused on identifying high-risk customers before they default. Accurate predictions enable financial institutions to proactively manage risk, minimize losses, and enhance credit decision-making.

The model should be optimized not just for accuracy, but also for **recall** and **F2-score**, to ensure that most defaulters are caught—even if it means allowing some false positives.

1.5 Approach

Our solution to the credit card default prediction task was built through a structured pipeline involving data analysis, feature engineering, model development, and evaluation. The key steps were:

- **Step 1: Data Exploration** – Performed exploratory analysis to understand feature distributions, detect patterns linked to default behavior, and assess class imbalance or missing values.
- **Step 2: Feature Engineering** – Created domain-informed features like credit utilization, payment-to-bill ratios, delay streaks, and repayment trends to better represent customer behavior.
- **Step 3: Handling Class Imbalance** – Addressed skewed class distribution using techniques such as SMOTE and class weighting to improve sensitivity to defaulters.
- **Step 4: Model Training & Evaluation** – Trained multiple models (Logistic Regression, Random Forest, LightGBM, XGBoost) and evaluated them using recall, F2-score, and AUC-ROC.
- **Step 5: Threshold Optimization** – Adjusted prediction thresholds to maximize the F2-score, prioritizing recall over precision.
- **Step 6: Feature Selection** – Leveraged feature importance scores from tree-based models to eliminate low-impact variables and enhance generalization.
- **Step 7: Final Prediction** – Used the best-performing model (XGBoost with optimized threshold and selected features) to predict defaults on the test set.

2 Data Preprocessing

Prior to model training, several preprocessing steps were applied to clean and prepare the dataset:

- **Missing Values:** The `AGE` column had 126 missing entries. These were imputed using the **median age** to maintain distribution symmetry and avoid bias.
- **Inconsistent Categorical Entries:** Some categorical columns included invalid or inconsistent values:
  - In the `MARRIAGE` column, values of `0` (an invalid category) were replaced with `3` (Others).
  - In the `EDUCATION` column, values `0`, `5`, and `6` were also invalid. These were grouped and replaced with `4` (Others). Since these accounted for over 1% of the data (311 entries), correction was preferred over removal.
- **Invalid Values in Precomputed Columns:** Two columns in the original dataset contained **negative values** that were logically inconsistent.

2.1 Demographic Analysis

Demographic features such as marriage, sex, education, and age were analyzed to explore their distribution and relationship with credit default behavior:

- **Marriage:** The data includes three categories — 1 (Married), 2 (Single), and 3 (Others). A slightly higher default rate was observed among married customers across all categories.
- **Sex:** Female customers show a slightly higher proportion of defaults compared to male customers.
- **Education:** Education levels are coded as 1 (Graduate School), 2 (University), 3 (High School), and 4 (Others). Default rates are highest for customers with a High School education, followed by University and Graduate School. The "Others" group has the lowest default rate.

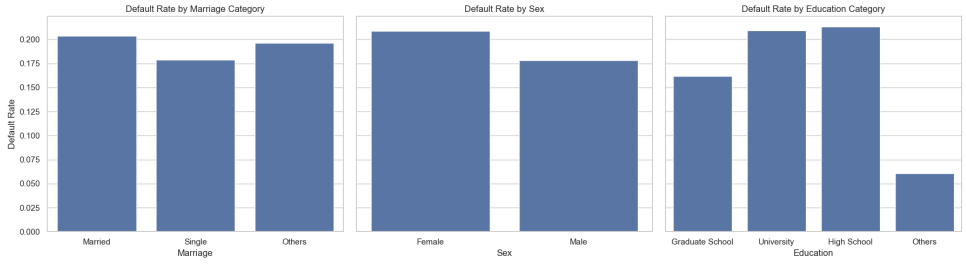


Figure 1: Default rate for marriage, sex and education

- **Age:** A KDE plot of age vs. default status shows that customers aged **25–35** have the highest default rate. The likelihood of default decreases with increasing age, indicating that younger customers are more prone to default.

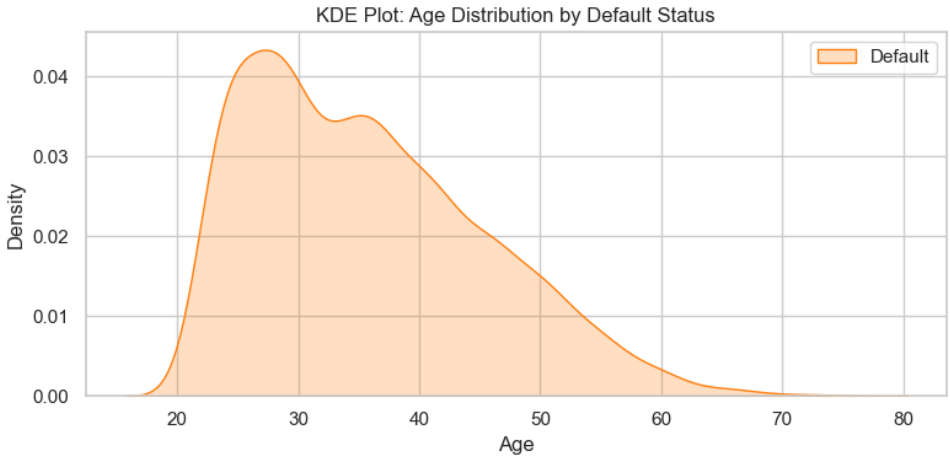


Figure 2: KDE - Age vs Default status

2.2 Financial Analysis

We conducted a detailed analysis of key financial features including credit limits, billing amounts, payments, and payment delay history to uncover their relationship with default behavior:

- **Credit Limit (LIMIT_BAL):**

  KDE analysis reveals that customers with lower credit limits (especially under 100,000) are more prone to default. The distribution is heavily skewed toward lower limits, with a long tail of high-credit customers. Although these upper outliers exist, we retained them due to their potential significance—representing high-value or premium users.

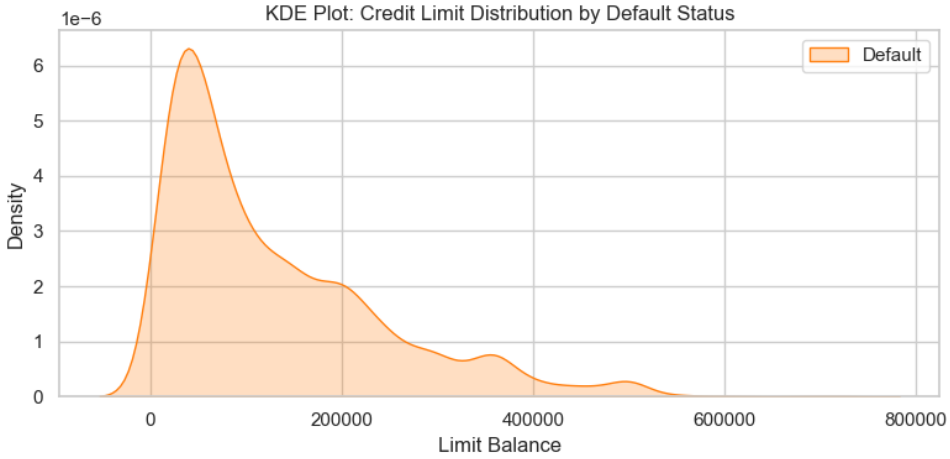


Figure 3: Credit Card limit distribution for defaults

- **Monthly Bill Amounts (BILL_AMT1 to BILL_AMT6):**

  Across all six months, defaulters generally show lower bill amounts compared to non-defaulters. This may imply conservative credit usage or financial constraints among high-risk customers.

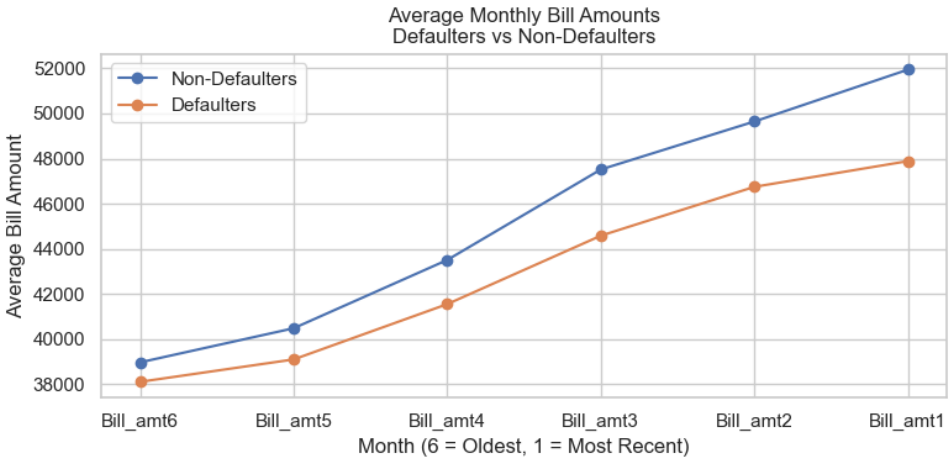


Figure 4: Last 6 months bills - Defaults vs Non Defaults

- **Average Bill Amount (AVG_BILL_AMT):**

  The trend continues with the average bill amount over six months—defaulters consistently spend less on average. While the data is left-skewed and contains over 2,000 high-value outliers, we preserved them to capture real-world financial extremes.

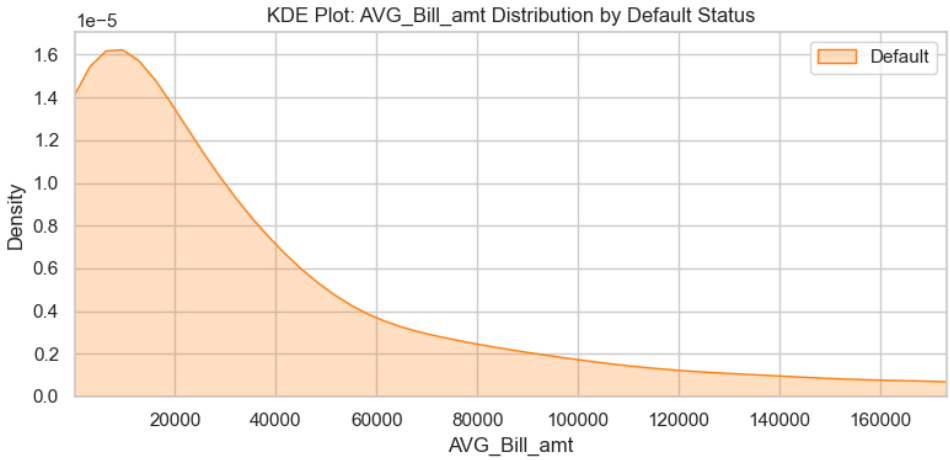


Figure 5: Average bills - Defaults

- **Monthly Payment Amounts (PAY_AMT1 to PAY_AMT6):**

  Defaulters tend to make significantly smaller payments than non-defaulters, suggesting either reduced ability or willingness to repay credit balances. This is a key differentiator in risk profiling.

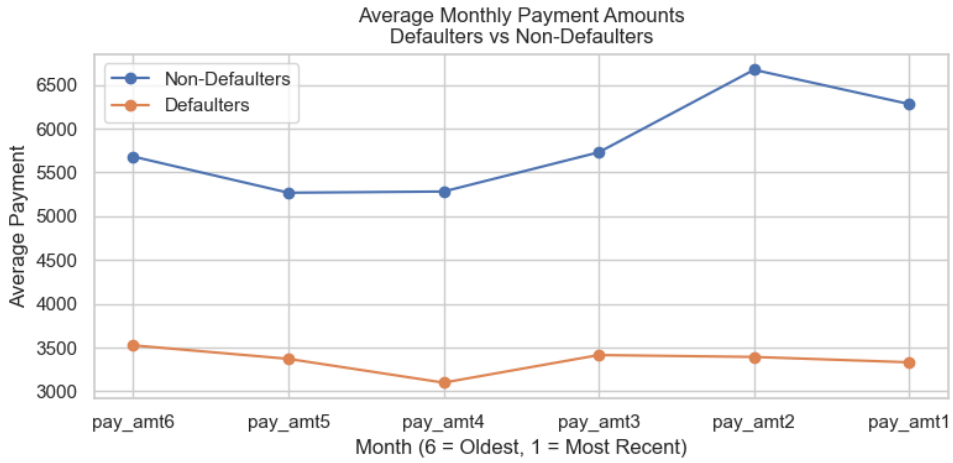


Figure 6: Average pay amounts - Defaults vs Non defaults

- **Payment Delay Status (PAY_0 to PAY_6):**

  Delay history strongly correlates with default risk. Defaulters show a higher frequency of delayed payments across all six months, reinforcing that sustained delinquency is a major red flag for credit risk.

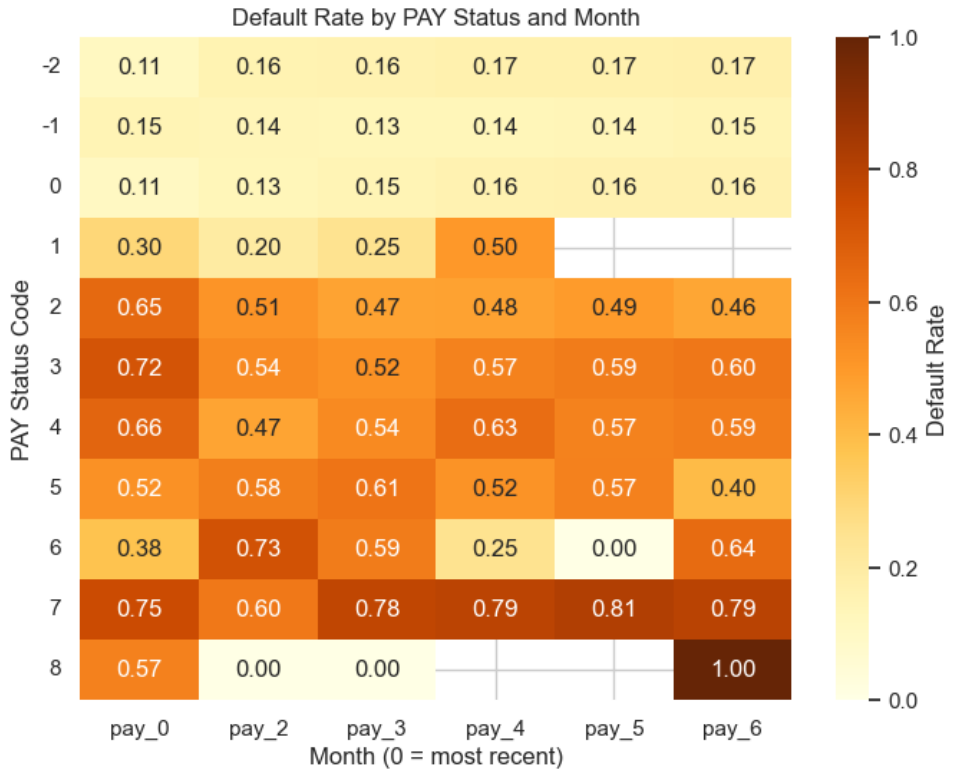


Figure 7: Payment delays relation with default

- **Payment-to-Bill Ratio:**

  This derived feature, representing how much of the billed amount was actually repaid, shows a clear pattern: defaulters typically have lower ratios, indicating partial or insufficient repayments. This

behavior aligns closely with increasing financial stress.

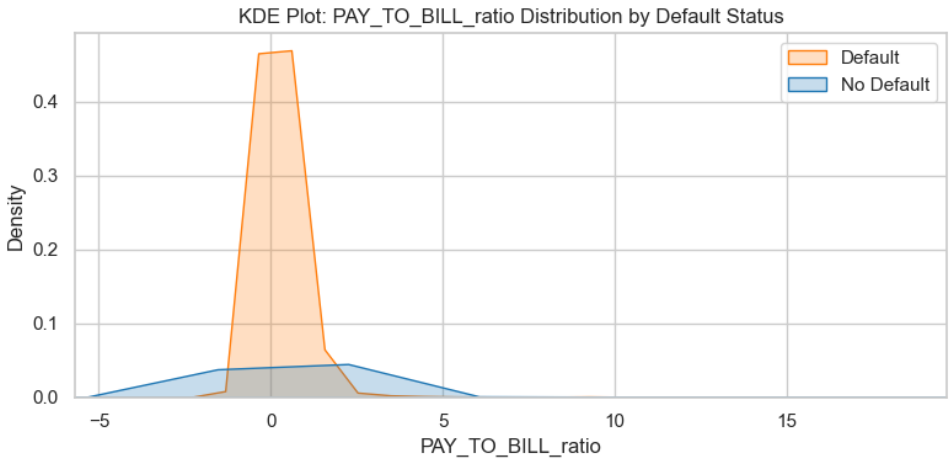


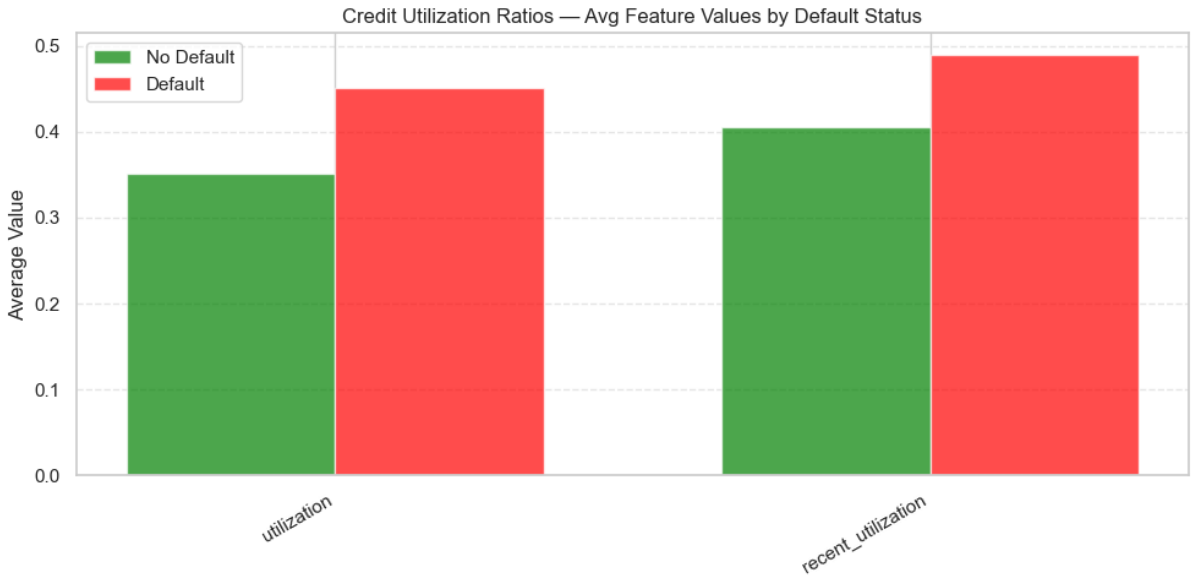
Figure 8: Pay to bill ratios - Default vs Non Default

Insight: Financial behaviors like consistent low repayments, high delay frequencies, and low payment-to-bill ratios are stronger indicators of default than the absolute value of bills or payments alone. These patterns help surface hidden risk in seemingly stable accounts.

3 Feature Engineering from Behavioral Analysis

This section outlines behavior-driven features derived from customer repayment activity, bill dynamics, and delinquency patterns. These features aim to mimic the risk assessment strategies used by financial institutions.

3.1 Credit Utilization Ratios

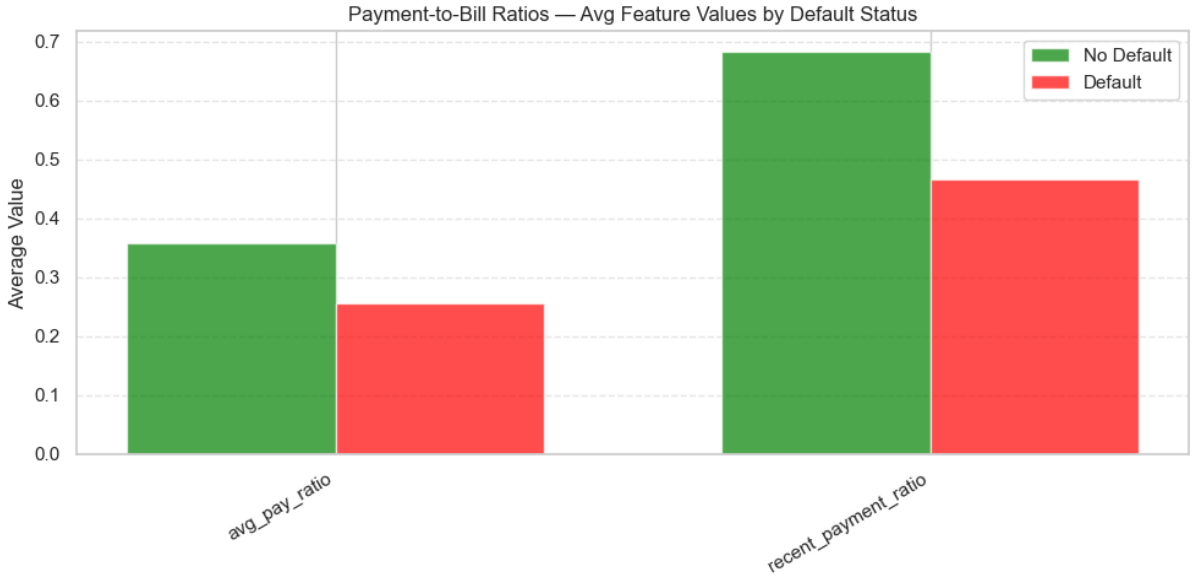


Observation: Both `utilization` and `recent_utilization` are significantly higher for defaulters.

Explanation: Even though bill amounts are similar across both groups, defaulters have lower credit limits. This leads to a higher utilization ratio.

Conclusion: *High credit utilization indicates financial stress and is strongly associated with default behavior.*

3.2 Payment-to-Bill Ratios

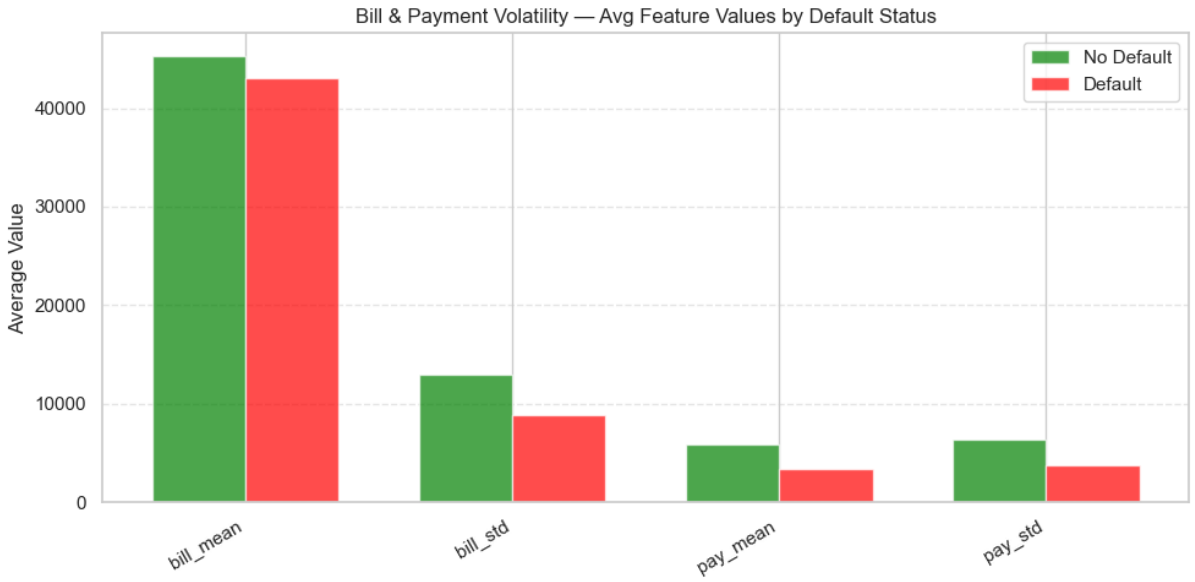


Observation: Defaulters have lower values across all ratios — `avg_pay_ratio`, `recent_payment_ratio`, and `PAY_TO_BILL_ratio`.

Explanation: This suggests they consistently pay less of their billed amount, possibly delaying or skipping payments.

Conclusion: *Poor repayment ratios are a key indicator of financial default.*

3.3 Bill and Payment Volatility

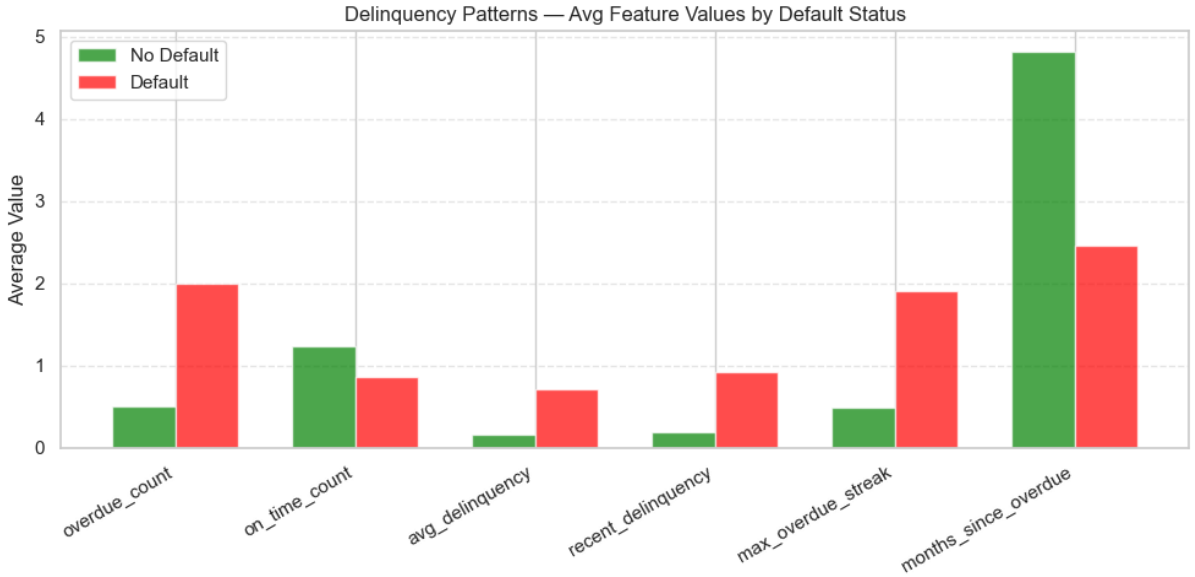


Observation: Non-defaulters show higher variability in payments and bills. Defaulters have lower `pay_mean` and `pay_std`.

Explanation: Defaulters make smaller, more consistent (but inadequate) payments, while non-defaulters are more adaptive.

Conclusion: *Payment volatility reflects active repayment behavior, while consistently low payments may suggest financial trouble.*

3.4 Delinquency Patterns

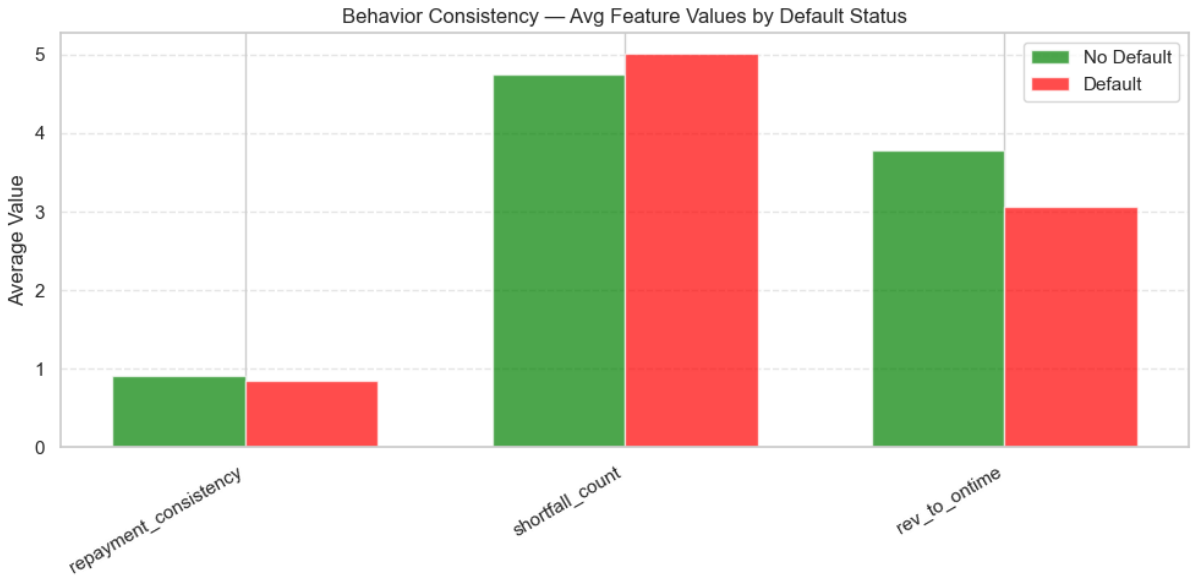


Observation: Defaulters have more overdue events, shorter time since last delinquency, and higher max overdue streaks.

Explanation: Chronic or recent delinquency signals a higher likelihood of default.

Conclusion: *Delinquency history is a strong predictive signal of default risk.*

3.5 Behavior Consistency

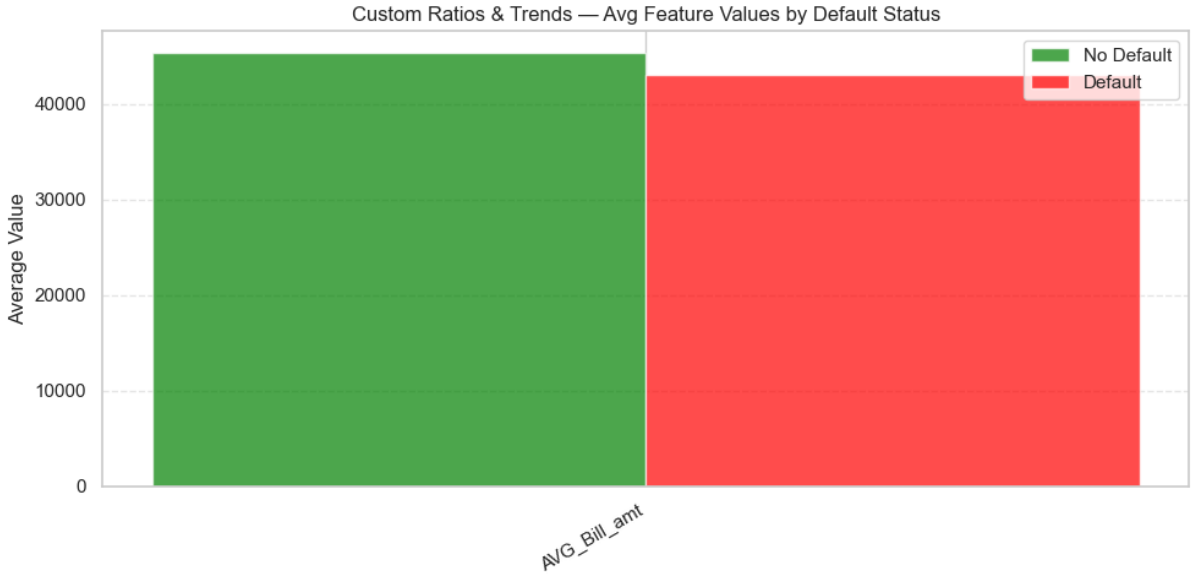


Observation: `shortfall_count` is higher and `rev_to_ontime` is lower for defaulters.

Explanation: This shows repayment irregularity and inability to repay even during credit revolving months.

Conclusion: *Inconsistent or insufficient payments are indicative of impending default.*

3.6 Custom Ratios & Trends



Observation: `AVG_Bill_amt` is almost the same for both groups.

Explanation: While billed amounts don't vary much, defaulters' lower credit limits cause high utilization.

Conclusion: *Bill size alone doesn't distinguish defaulters. Contextual ratios matter more.*

4 Feature Selection

After deriving a wide range of behavioral and demographic features, we performed feature selection to retain only the most impactful variables for default prediction. This process enhances model performance by removing noise, reducing overfitting, and improving interpretability.

4.1 Feature Importance Analysis

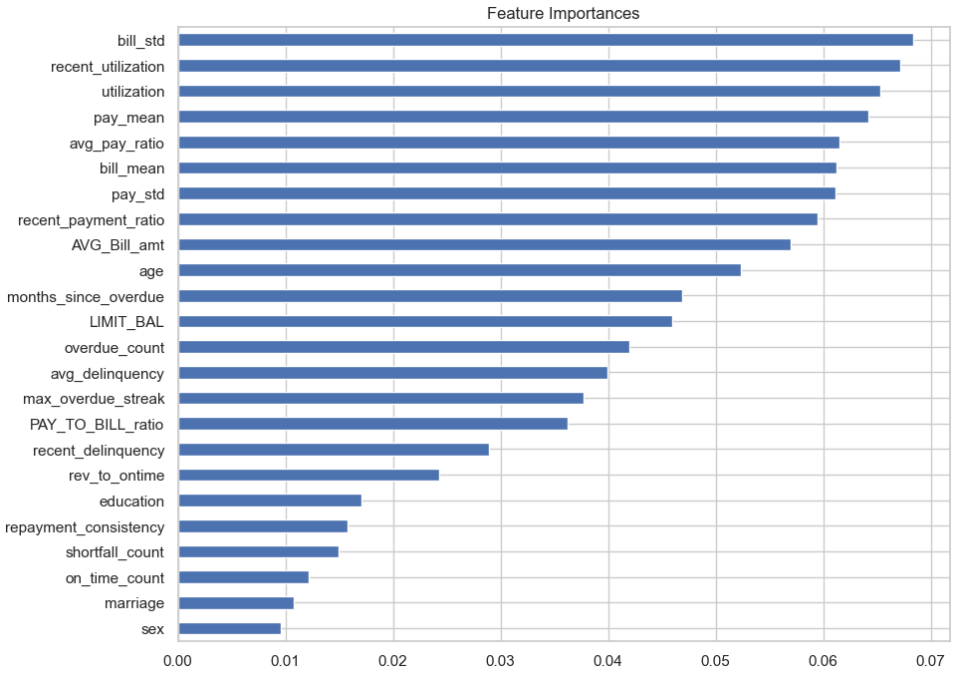


Figure 9: Feature Importances from Random Forest Model

We used a Random Forest Classifier to assess feature importance, leveraging its built-in ability to evaluate the contribution of each feature based on information gain. The importance values were visualized as shown in Figure 9.

- **bill_std, bill_mean:** These features capture volatility and average spending behavior. High fluctuations are often linked to financial instability, making these critical predictors.
- **recent_utilization, utilization:** Reflect a customer's immediate and long-term credit usage. High utilization is a strong signal of overleveraging and potential default.
- **pay_mean, pay_std:** Provide insights into repayment volume and consistency. Irregular payments indicate risk.

- **avg_pay_ratio, recent_payment_ratio:** Indicate how effectively a customer manages bill obligations. Low values suggest underpayment.
- **AVG_Bill_amt:** Reflects the scale of the customer's spending activity.
- **age:** Younger customers may exhibit more volatile financial behavior compared to older individuals.
- **months_since_overdue, overdue_count:** Indicate repayment discipline and recency of financial issues.
- **LIMIT_BAL:** Acts as a baseline financial capacity indicator.
- **avg_delinquency, max_overdue_streak:** Capture the average delay and longest continuous default streak—key indicators of financial stress.
- **PAY_TO_BILL_ratio:** A cumulative measure of a customer's ability to match repayments with spending.

These 20 features were selected based on their predictive strength as illustrated in the feature importance chart (Figure 9). They collectively capture a wide range of behavioral, financial, and demographic characteristics that are essential for robust credit risk modeling.

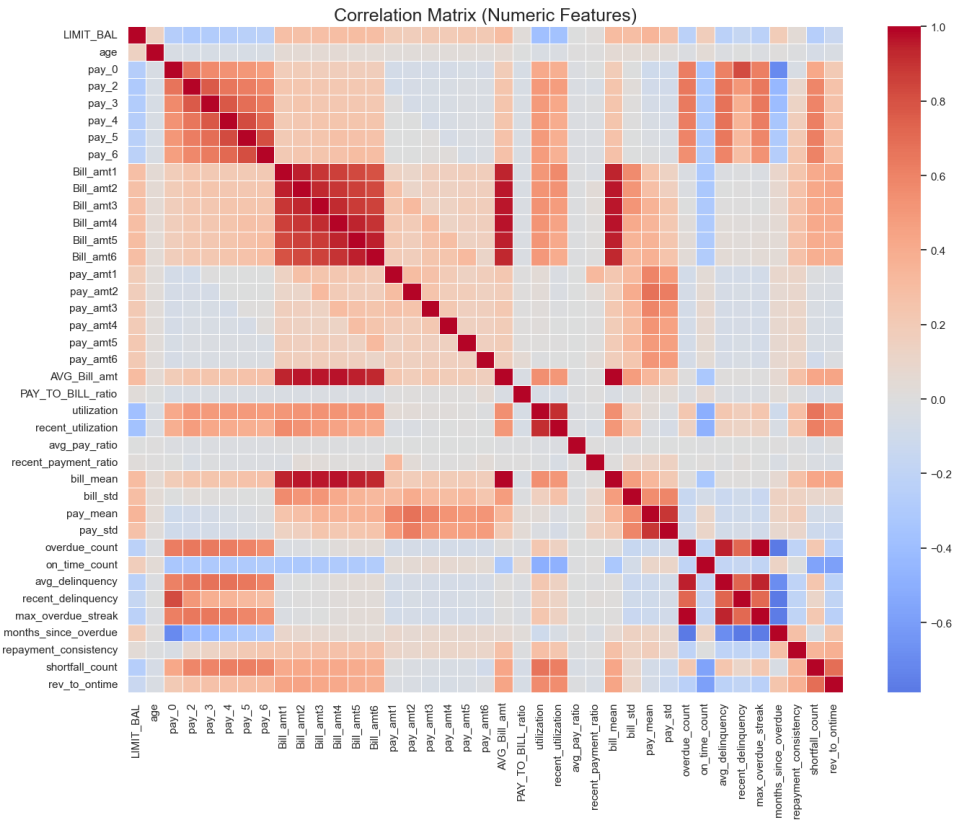


Figure 10: Features Correlation Heatmap

4.2 Final Feature Set

The final 20 features used for training are:

```
['bill_std', 'recent_utilization', 'utilization', 'pay_mean', 'avg_pay_ratio', 'bill_mean',
'pay_std', 'recent_payment_ratio', 'AVG_Bill_amt', 'age', 'months_since_overdue',
'LIMIT_BAL', 'overdue_count', 'avg_delinquency', 'max_overdue_streak', 'PAY_TO_BILL_ratio',
'recent_delinquency', 'rev_to_ontime', 'education', 'repayment_consistency']
```

These features provide comprehensive coverage of customer behavior, repayment discipline, and financial risk, and were retained for subsequent modeling.

5 Class Imbalance Handling and Model Training

Multiple machine learning models were trained and evaluated to predict credit card defaults. Each model was tested with different strategies for handling class imbalance and optimizing performance.

5.1 Stratified Train–Test Split

We first split the data into training and test sets while preserving the default rate:

```
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.3,
    stratify=y,
    random_state=42
)
```

Categorical columns were converted to integer codes, and any missing values were filled with zeros.

5.2 Class Imbalance Handling

Credit card default datasets are highly imbalanced. We employed two main techniques:

- **Class Weights:** Models were initialized with `class_weight='balanced'` (or `scale_pos_weight` for XGBoost) to penalize misclassification of the minority class more heavily.
- **SMOTETomek:** We applied SMOTE combined with Tomek links to oversample defaulters and clean borderline examples:

```
from imblearn.combine import SMOTETomek

smk = SMOTETomek(random_state=42)

X_train_res, y_train_res = smk.fit_resample(X_train, y_train)
```

```
Class distribution in y_train before resampling:
Counter({0: 14307, 1: 3365})
Class distribution in y_train after resampling:
Counter({0: 13473, 1: 13473})
```

Figure 11: SMOTE - Before vs After counts

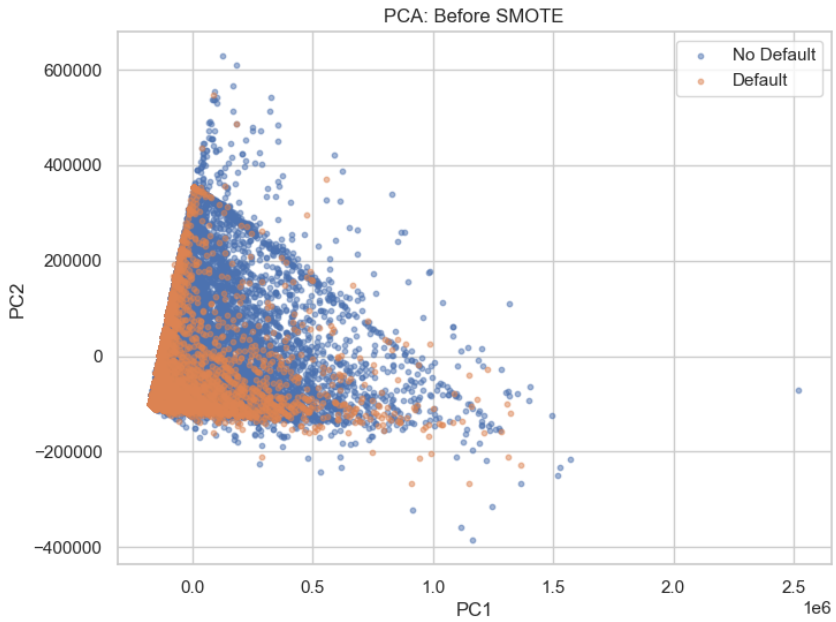


Figure 12: Before SMOTE

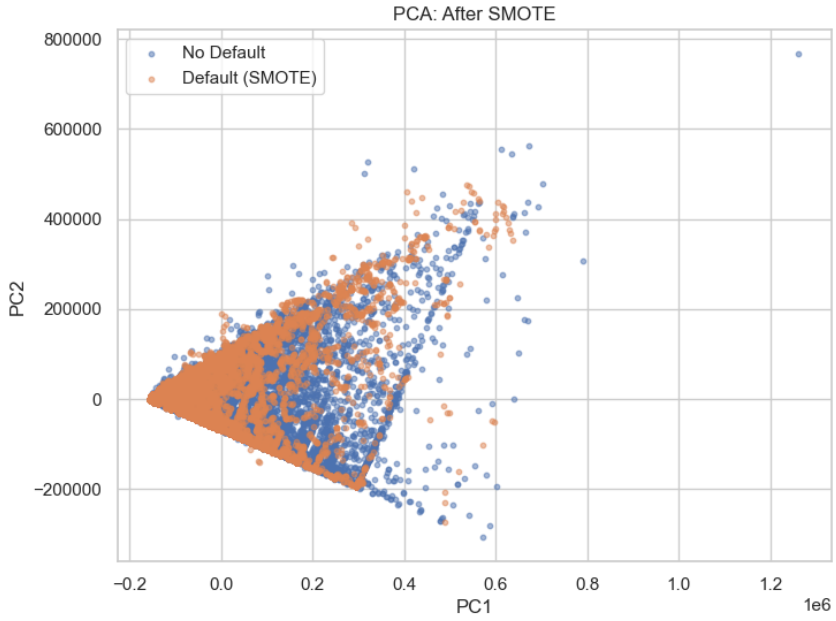


Figure 13: After SMOTE

5.3 Models and Hyperparameters

- **Logistic Regression:** `class_weight='balanced'`, `max_iter=1000`
- **Random Forest:** `class_weight='balanced'`, `n_estimators=200`, `max_depth=10`
- **XGBoost:** `scale_pos_weight = len(y_train==0) / len(y_train==1)`, `eval_metric='logloss'`

5.4 Threshold Tuning and Evaluation

We tuned the decision threshold to maximize the F_2-score. For each model:

1. Fit on $\{X_{\text{train\_res}}, y_{\text{train\_res}}\}$.
2. Predict probabilities $\hat{p} = P(y = 1)$ on X_{test}.
3. For t in $[0.10, 0.11, \ldots, 0.89]$, compute:
   - Accuracy
   - Precision
   - Recall
   - F_1-score
   - F_2-score
   - AUC–ROC
4. Select t^* that yields the highest F_2-score.

```
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score,
    f1_score, fbeta_score, roc_auc_score
)


def evaluate_model(name, model):
    model.fit(X_train_res, y_train_res)
    y_proba = model.predict_proba(X_test)[:,1]
    thresholds = np.arange(0.1, 0.9, 0.01)
    f2_scores = [fbeta_score(y_test, y_proba>=t, beta=2) for t in thresholds]
    best_t = thresholds[np.argmax(f2_scores)]
    y_pred = (y_proba >= best_t).astype(int)

    return {
        'Model': name,
        'Threshold': best_t,
        'Accuracy': accuracy_score(y_test, y_pred),
        'Precision': precision_score(y_test, y_pred),
        'Recall': recall_score(y_test, y_pred),
        'F1-score': fbeta_score(y_test, y_pred, beta=1),
        'F2-score': fbeta_score(y_test, y_pred, beta=2),
        'AUC-ROC': roc_auc_score(y_test, y_proba)
    }
```

5.5 Final Model Comparison

After evaluating all models, we assemble the results into a table sorted by F_2-score:

```
results_df = pd.DataFrame(results).set_index('Model')
results_df.sort_values('F2-score', ascending=False)
```

Table 1: Model Performance Comparison on Test Set

Model	Threshold	Accuracy	Precision	Recall	F1	F2	AUC-ROC
Logistic Regression	0.34	0.6712	0.3325	0.7219	0.4553	0.5849	0.7578
Random Forest	0.19	0.5807	0.2915	0.8405	0.4329	0.6105	0.7849
XGBoost	0.32	0.6256	0.3066	0.7663	0.4380	0.5895	0.7546

Key Insights:

- *Random Forest* achieved the highest F_2-score (0.6105) and recall (0.8405).
- *Logistic Regression* plateaued at F_2=0.5849.
- *SMOTE* variants improved recall but sometimes reduced overall F_2 due to calibration issues.
- *Threshold tuning* is essential to balance recall and precision according to business priorities.

6 Model Selection

After training and evaluating multiple classifiers, we selected the final model and decision threshold based on a balance of F_2-score and overall performance metrics.

6.1 Threshold Optimization

To align our predictions with the business objective of minimizing missed defaulters, we optimized the probability cutoff for the Random Forest model by maximizing the F_2-score on the validation set. Recall that

$$F_2 = (1 + 2^2) \frac{\text{Precision} \times \text{Recall}}{2^2 \,\text{Precision} + \text{Recall}}$$

weights recall twice as heavily as precision.

We evaluated thresholds from 0.10 to 0.90 in steps of 0.01. At each threshold t, we computed:

- Accuracy
- Precision
- Recall
- F_1-score
- F_2-score
- Number of predicted positives

The highest F_2-score (0.6105) occurred at $t = 0.19$. However, this low threshold produced a large number of positives (4,158) and relatively low accuracy (0.581), which may increase false alarms.

6.2 Trade-off and Final Threshold

To improve overall accuracy and precision while still maintaining strong recall, we selected a slightly higher threshold of $t = 0.29$. At this cutoff, the Random Forest model achieves:

$$\begin{aligned} \text{Accuracy} = 0.731, \quad & \text{Precision} = 0.382, \\ \text{Recall} = 0.673, \quad & F_1\text{-score} = 0.488, \\ F_2\text{-score} = 0.584, \quad & \#\text{Positives} = 2537. \end{aligned}$$

This threshold represents a practical compromise:

- It reduces false positives from 4,158 at $t = 0.24$ to 2,537 at $t = 0.29$.
- It boosts accuracy from 0.581 to 0.731.
- It incurs only a modest drop in F_2-score ($0.611 \rightarrow 0.584$).

6.3 Final Model Choice

- **Random Forest** was chosen because it achieved the highest F_2-score (0.611 at $t = 0.19$) among all models and demonstrated robust recall (0.84) and AUC-ROC (0.785).
- By adjusting the threshold to 0.29, we improved accuracy and precision, yielding a more balanced operational profile while still prioritizing recall of high-risk customers.

These decisions ensure the deployed model aligns with the bank's risk appetite: catching as many defaulters as possible (high recall and F_2-score) while controlling the rate of false alarms through a higher decision threshold.

7 Final Predictions and Conclusion

Using the optimized Random Forest model and a decision threshold of $t = 0.35$, we generated default probabilities for each of the 5,016 customers in the unseen validation dataset. Figure 6 illustrates the overall probability distribution, with the red dashed line marking our chosen cutoff.

After applying the threshold, we obtained the following class proportions:

Class	Proportion (%)	Count
Non-default (0)	66.76	3,350
Default (1)	33.24	1,667

Table 2: Final Predictions on Validation Set (N = 5,016)

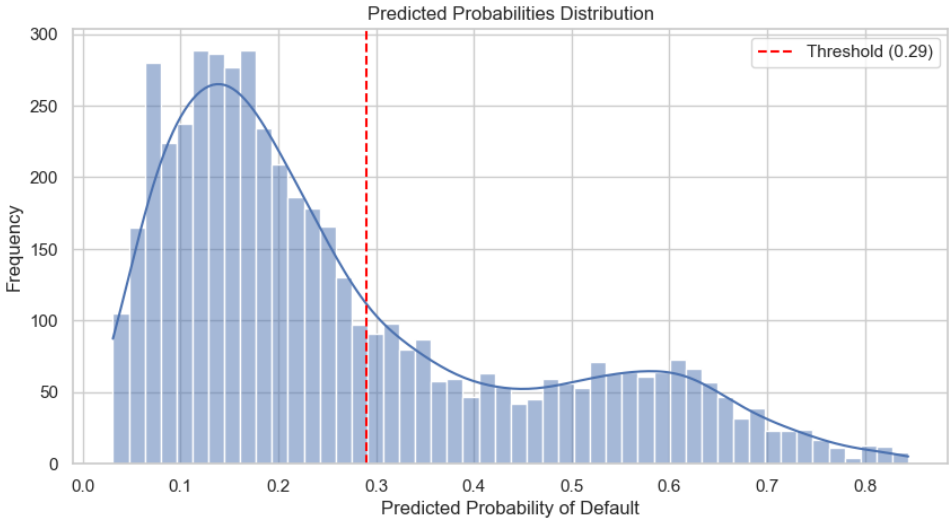


Figure 14: Predicted Probability Distribution of Default

These predictions were saved to `submission_23114050.csv` in the required format [Customer ID, next_month_default]. The full code, data processing scripts, and output files are available in our GitHub repository: `https://github.com/krishsingla06/Credit-Card-Default-Prediction-23114050`.

Conclusion. By combining robust feature engineering, careful threshold optimization, and a Random Forest classifier tuned for F_2-score, we built a model that balances the need to catch high-risk defaulters (recall) against the operational cost of false positives (precision). This solution meets the business requirement of proactive credit risk management while maintaining transparency and reproducibility.

Thank you!