International Institute of Information Technology
Bangalore

Machine Learning
AIM 511

Project Report

Team Members:

Ajitesh Kumar Singh (IMT2022559)
Ketan Ghungralekar (IMT2022058)
Abhinav Deshpande (IMT2022580)

February 17, 2025

iiit-b
ज्ञानमुत्तमम्

Data Analysis and Model Training Report

1. Preprocessing Steps

Data Loading and Feature Exploration

The training and testing datasets were loaded, and an initial feature analysis was conducted to examine the data structure, column types, and unique values.

- Missing values were checked, and none were found, ensuring a clean dataset for model training.

Flowchart of Process

The flowchart below visualizes the major steps in the data processing and modeling pipeline, from data preprocessing to model training and prediction.

Feature Selection

- **Categorical Variables:** Key categorical columns included `HasCoSigner`, `LoanPurpose`, `HasDependents`, `HasMortgage`, `MaritalStatus`, `EmploymentType`, and `Education`. These features were label-encoded to transform categorical values into numerical representations.
- **Numerical Variables:** All numerical columns, excluding `LoanID`, were retained as they could provide predictive insights.
- **Target Variable:** The target column `Default` indicated whether a loan was defaulted. This binary variable was used in **training** classification models.

Data Transformation

Label encoding was applied to the categorical columns in both training and test datasets to convert categorical features into numerical values. The `LoanID` column, which served as a unique identifier, was removed as it did not contribute to prediction.

Data Splitting

The training data was split into feature set X and target variable y (representing `Default`). 75% of the data was used for training, and 25% was reserved as a validation set for performance evaluation.

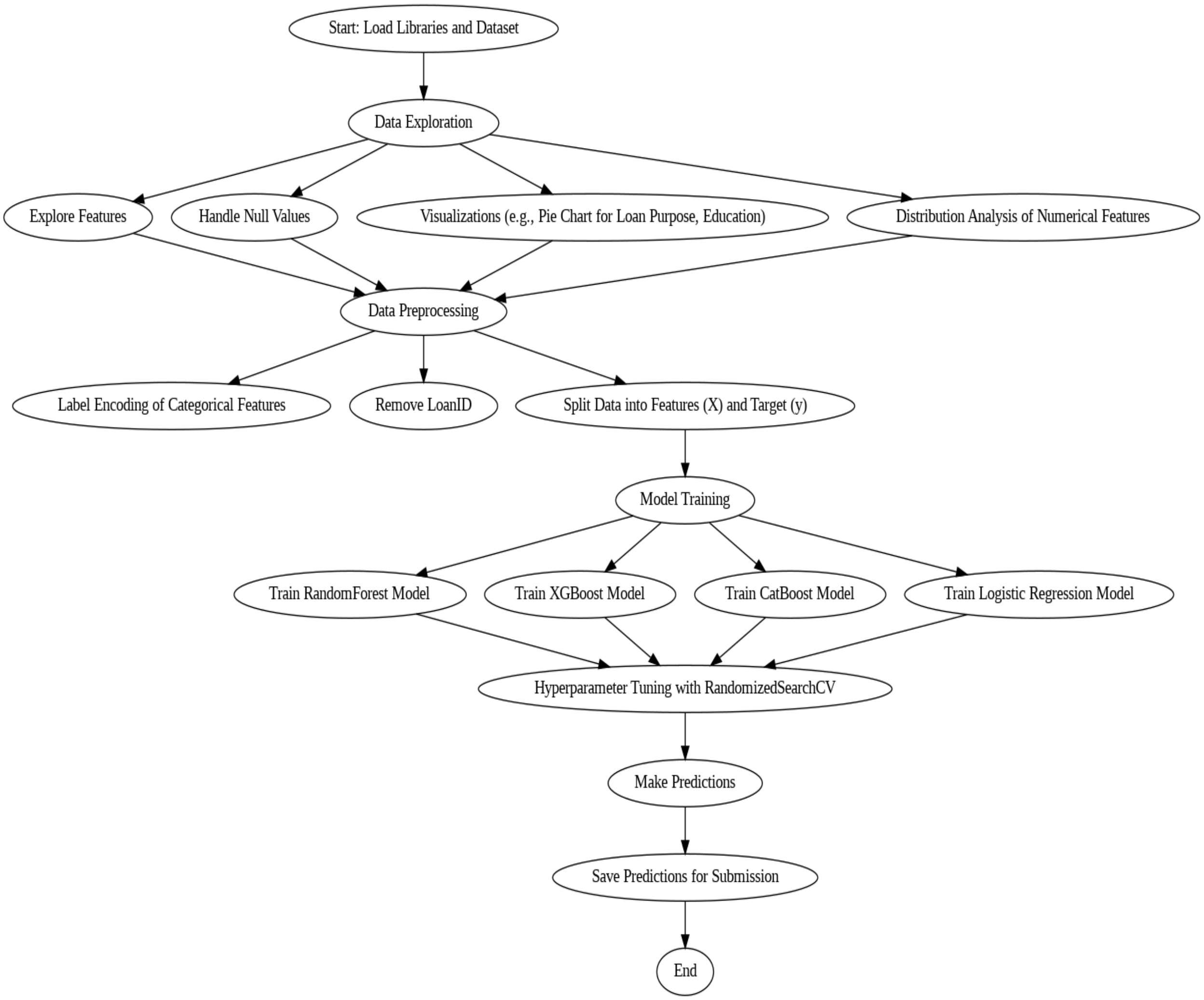
Start: Load Libraries and Dataset
Data Exploration
Explore Features
Handle Null Values
Visualizations (e.g., Pie Chart for Loan Purpose, Education)
Distribution Analysis of Numerical Features
Data Preprocessing
Label Encoding of Categorical Features
Remove LoanID
Split Data into Features (X) and Target (y)
Model Training
Train RandomForest Model
Train XGBoost Model
Train CatBoost Model
Train Logistic Regression Model
Hyperparameter Tuning with RandomizedSearchCV
Make Predictions
Save Predictions for Submission
End

Figure 1: Flowchart of Data Processing and Model Training Pipeline

2. Exploratory Data Analysis (EDA)

Distribution of Loan Purpose

The distribution of loan purposes is shown below. The pie chart provides an overview of the different loan purposes within the dataset.

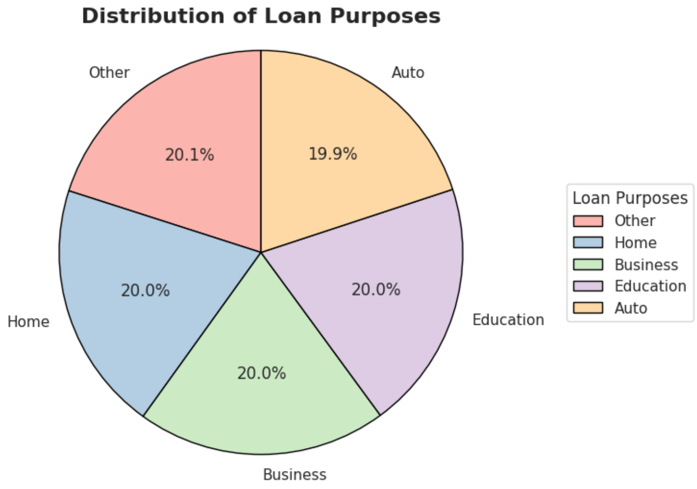

Figure 2: Distribution of Loan Purposes

Distribution of Education Levels

The distribution of education levels among the loan applicants is shown in the following pie chart.

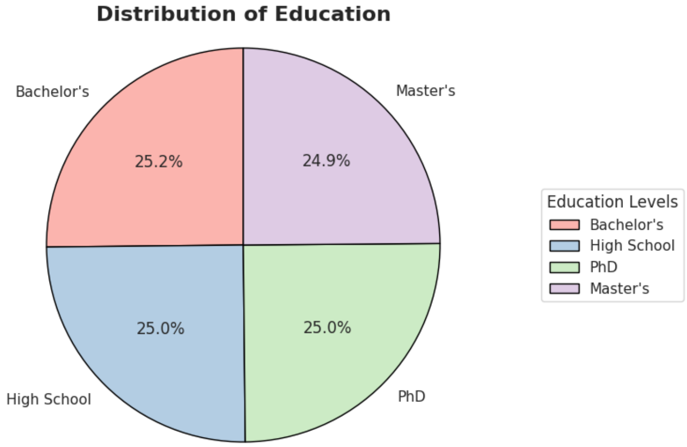

Figure 3: Distribution of Education Levels

Validity Checks for Numerical Columns

Each numerical column was checked to ensure values fell within realistic ranges, identifying potential data anomalies. For example:

- **Age:** Checked for values between 0 and 100.
- **Income:** Expected range of 0 to 500,000.
- **LoanAmount:** Expected range 0 to 1,000,000.
- **CreditScore:** Checked for values between 300 and 850.

Result: All values were found in valid range.

Feature Distributions

Histograms and density plots were generated for numerical features to examine data distribution and central tendencies.

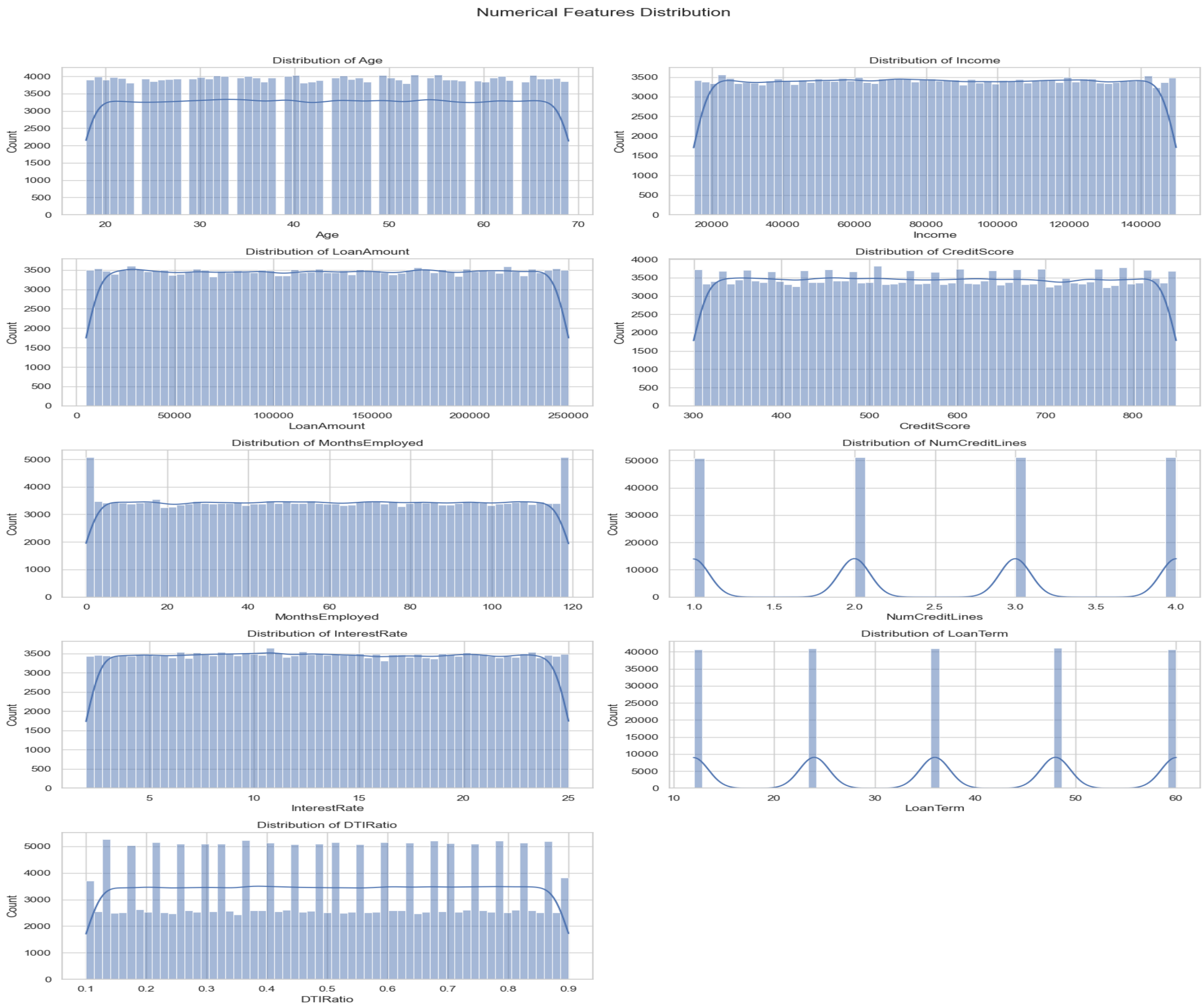

Figure 4: Distribution of numerical features

- **Age:**
  - The age distribution appears roughly uniform, with values ranging from around 20 to 70.
  - This uniform distribution suggests that age is well-spread across the dataset without any particular age group dominating, making it unnecessary to apply any standardization or transformation.
- **Income:**
  - Income shows a nearly uniform distribution between 20,000 and 140,000, with a slightly higher concentration around the mean.

  - This balanced spread indicates a diverse range of incomes among the dataset, suggesting a representative sample across different income levels.

- **LoanAmount:**
  - The distribution of loan amounts is also approximately uniform, spanning from 0 up to 250,000.
  - The absence of major peaks suggests that loan amounts are distributed evenly, representing different loan categories without any heavy bias.

- **CreditScore:**
  - Credit scores are nearly uniformly distributed from around 300 to 800.
  - This indicates that individuals with a broad range of creditworthiness are included, providing diversity in credit score levels.

- **MonthsEmployed:**
  - The months employed feature shows a relatively uniform distribution, with values up to 120.
  - This uniformity suggests that individuals with varying lengths of employment history are included, supporting varied employment backgrounds.

- **NumCreditLines:**
  - The distribution of the number of credit lines shows distinct peaks at whole number values, with values ranging from 1 to 4.
  - This pattern likely arises because the number of credit lines is discrete (integer-based), and most individuals have between 1 and 4 credit lines.

- **InterestRate:**
  - Interest rates are approximately uniformly distributed, with values ranging from around 5 to 25.
  - This distribution suggests a variety of loan products with differing interest rates, providing a broad view of borrowing costs.

- **LoanTerm:**
  - Loan terms have clear peaks at regular intervals, corresponding to common loan term durations.
  - These peaks are likely due to typical loan products with standard term lengths (e.g., 15, 30, or 45 years), reflecting realistic loan market conditions.

- **DTIRatio (Debt-to-Income Ratio):**
  - The debt-to-income ratio shows a nearly uniform distribution, with values spread between 0.1 and 0.9.
  - This spread indicates a range of debt burden levels relative to income, offering diverse risk profiles among the individuals.

Conclusion: Since most of these features are either uniformly distributed or have natural discrete values, standardization was deemed unnecessary. The dataset captures a broad range of values for each feature, supporting diverse customer profiles for analysis.

Outlier Detection

Box plots were used to identify potential outliers in numerical columns. The interquartile range (IQR) method was applied, flagging values outside the range of Q1 - 1.5 * IQR to Q3 + 1.5 * IQR as potential outliers.

- Analysis of numerical columns confirmed that the dataset did not contain extreme values, suggesting no significant outliers were present.

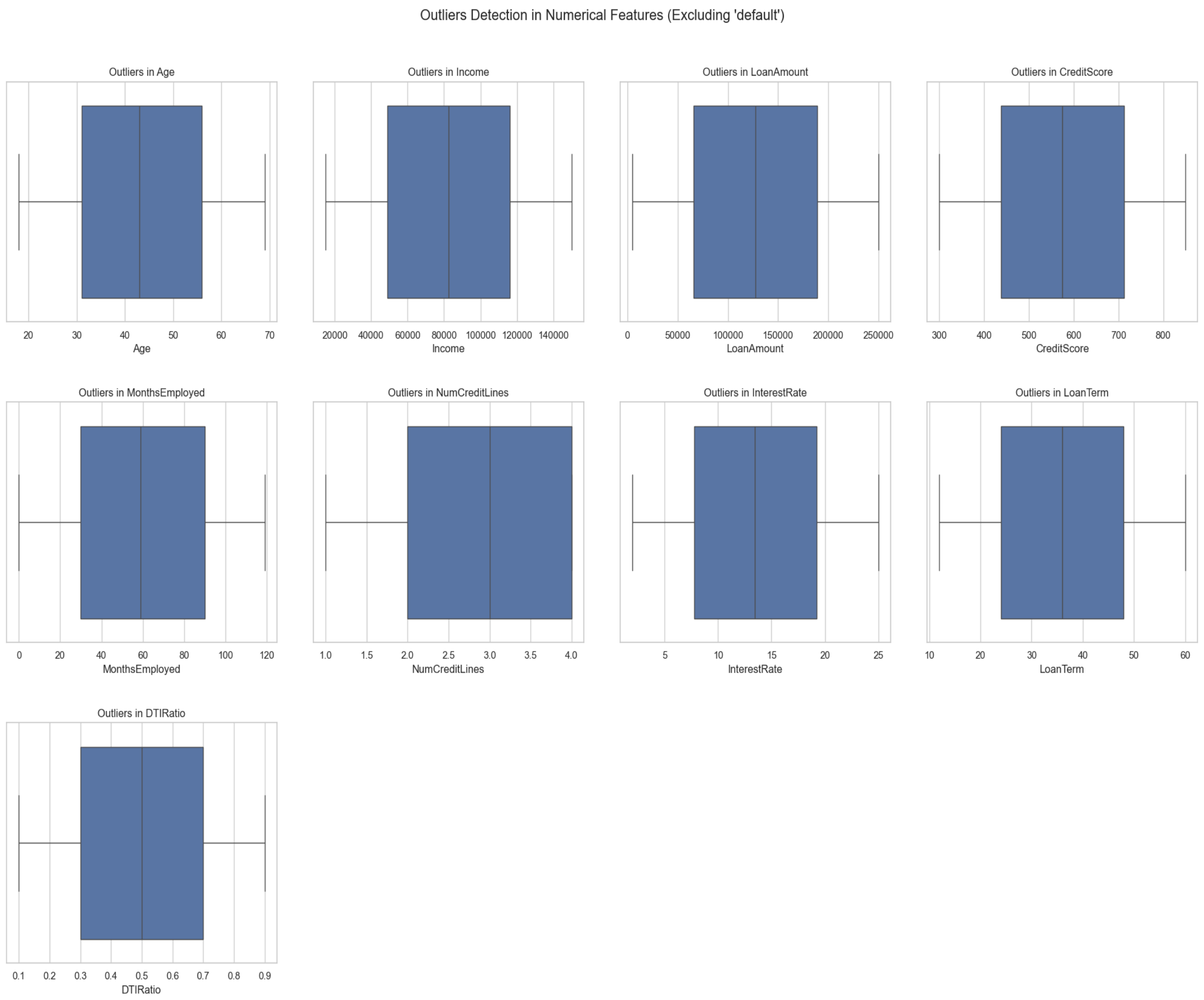

Figure 5: Outlier detection using boxplot

Correlation Analysis

A correlation heatmap was generated to visualize relationships among numerical features, helping to identify multicollinearity.

- Upon studying this heatmap, no significant correlation was found between any 2 features

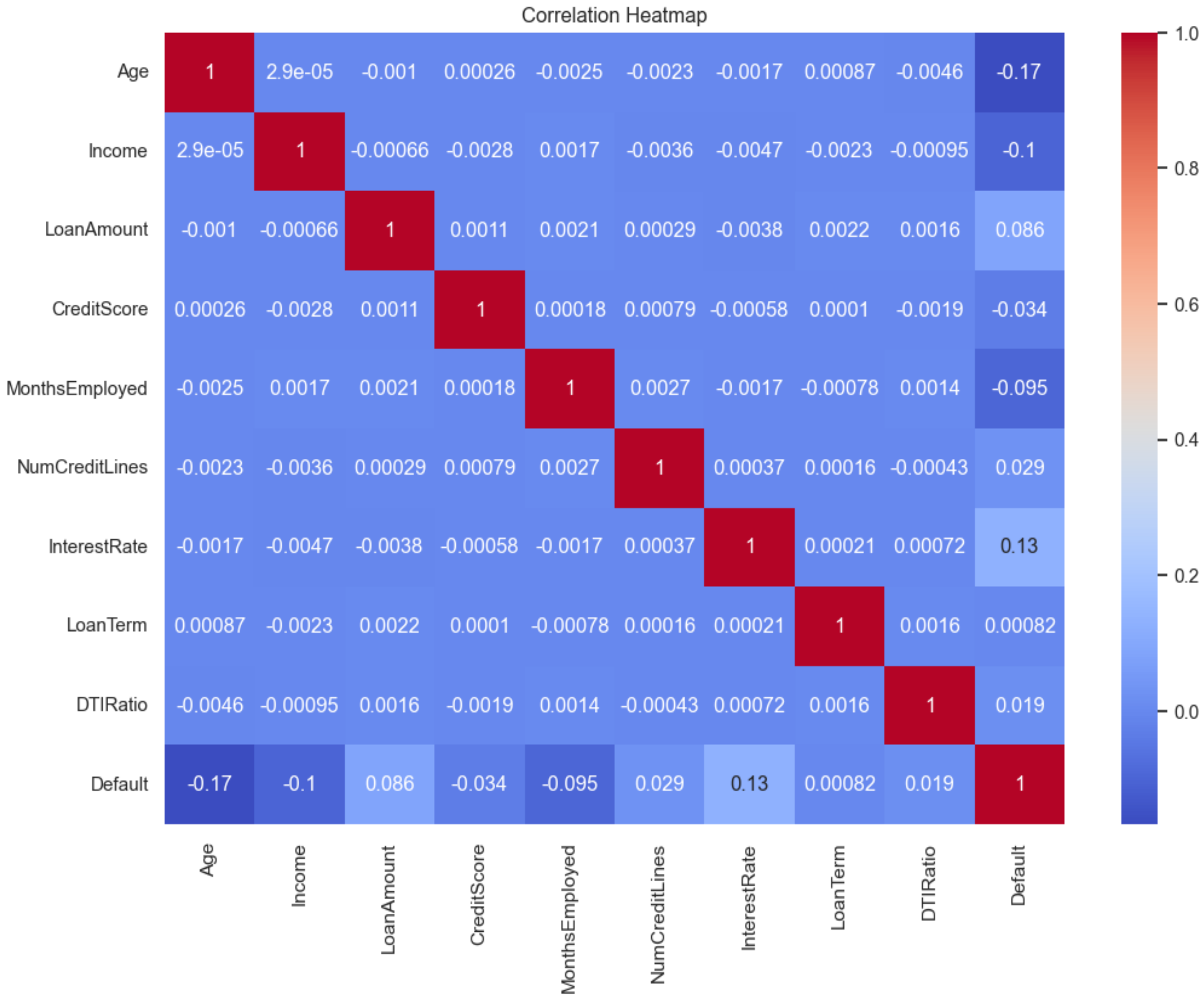

Figure 6: Correlation Heatmap

Target Variable Analysis

The distribution of the `Default` target variable was analyzed to assess class balance. Since `Default` is a binary variable, classification models were chosen for this problem.

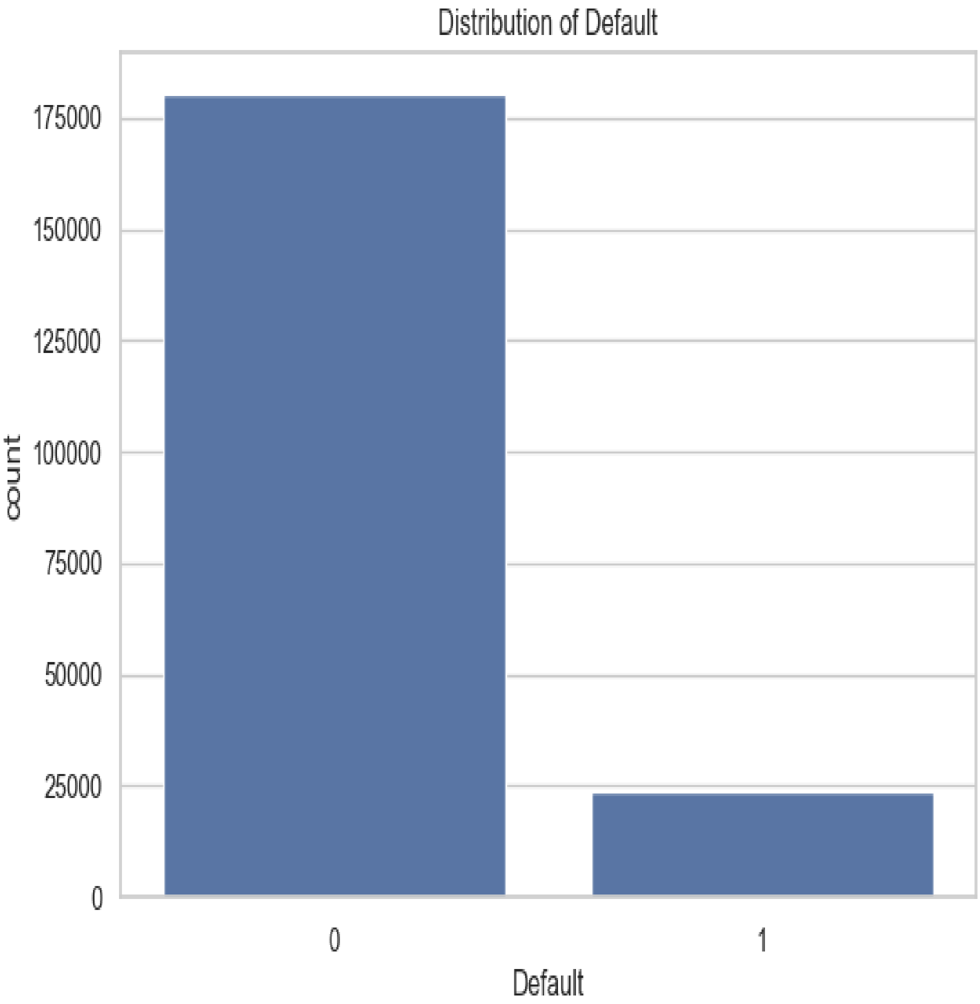

Figure 7: Distribution of target variable

Numerical Columns Analysis Against Target

Box plots were generated for each numerical feature against the target variable to observe patterns in distributions across the target classes.

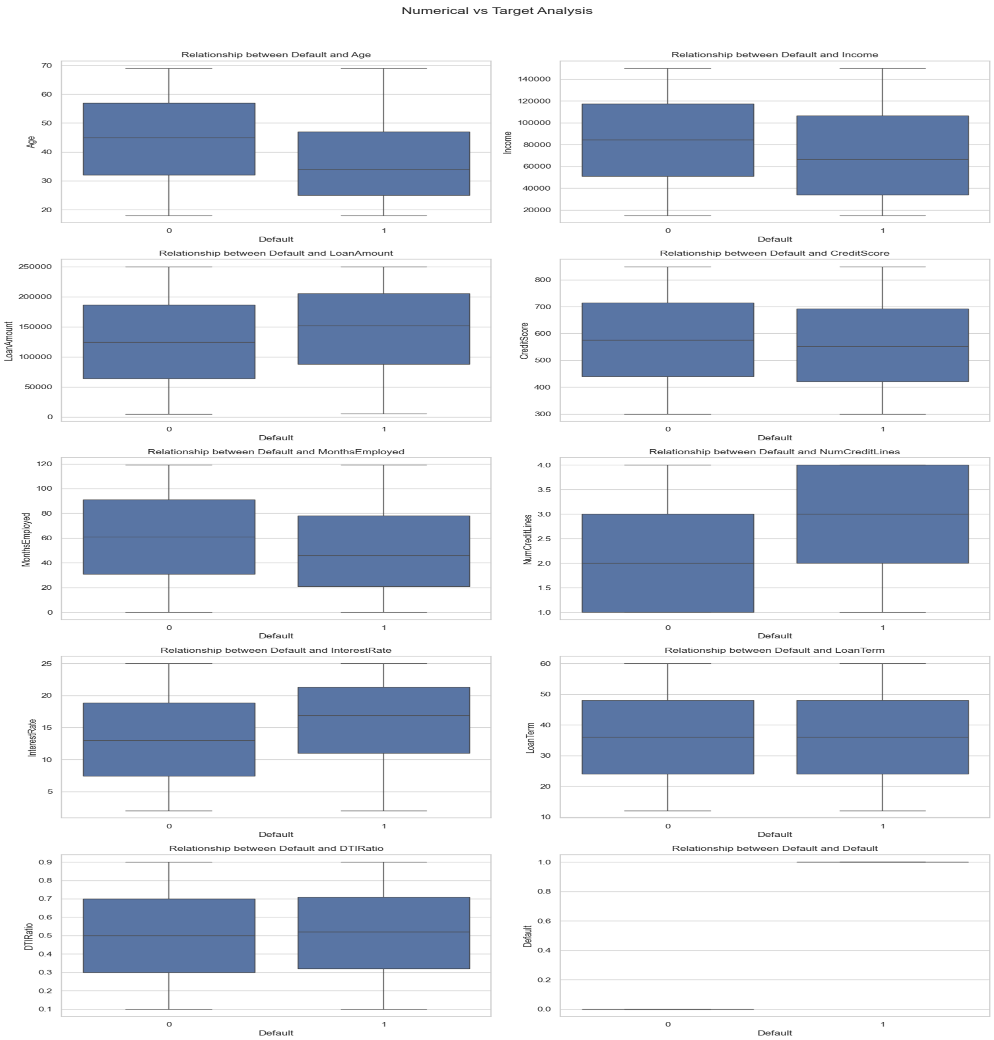

Figure 8: Relationship between target and features

- **Age vs. Default:**
  - Non-default cases (0) tend to have a slightly older median age than default cases (1).
  - The age ranges are broad for both classes, but there's a slight indication that younger individuals might have a higher likelihood of default.
- **Income vs. Default:**
  - The income distribution for non-default cases is somewhat higher than for default cases.
  - Non-default cases (0) have a broader range of incomes, while default cases (1) seem to concentrate at lower income levels. This suggests income may have a relationship with the likelihood of default.
- **LoanAmount vs. Default:**
  - Both default and non-default cases have a similar distribution of loan amounts.
  - There is a slight overlap in the interquartile range, indicating that loan amount alone may not be a strong predictor of default.

- **CreditScore vs. Default:**
  - Non-default cases (0) generally have higher credit scores than default cases (1).
  - This aligns with the expectation that higher credit scores are associated with a lower likelihood of default.
- **MonthsEmployed vs. Default:**
  - Non-default cases (0) show a higher median and broader distribution of months employed, while default cases (1) have a lower median employment duration.
  - This suggests that individuals with longer employment histories may be less likely to default.
- **NumCreditLines vs. Default:**
  - Non-default cases tend to have a higher median number of credit lines than default cases.
  - This may indicate that individuals with more established credit histories (more credit lines) are less likely to default.
- **InterestRate vs. Default:**
  - Both classes have overlapping interest rate distributions, but the median interest rate is slightly higher for default cases.
  - Higher interest rates might be slightly associated with default risk, though the relationship is not very strong.
- **LoanTerm vs. Default:**
  - Loan terms show similar distributions across default and non-default cases, with little distinction.
  - This suggests that loan term duration may not significantly impact the likelihood of default.
- **DTIRatio (Debt-to-Income Ratio) vs. Default:**
  - Default cases (1) tend to have a higher median debt-to-income ratio than non-default cases.
  - This supports the idea that individuals with higher debt burdens relative to their income may be more likely to default.
- **Default vs. Default:**
  - This plot is redundant, showing the distribution of the target variable against itself.

Categorical Columns Analysis Against Target

Histograms were generated for each categorical feature against the target variable to observe patterns in distributions across the target classes.

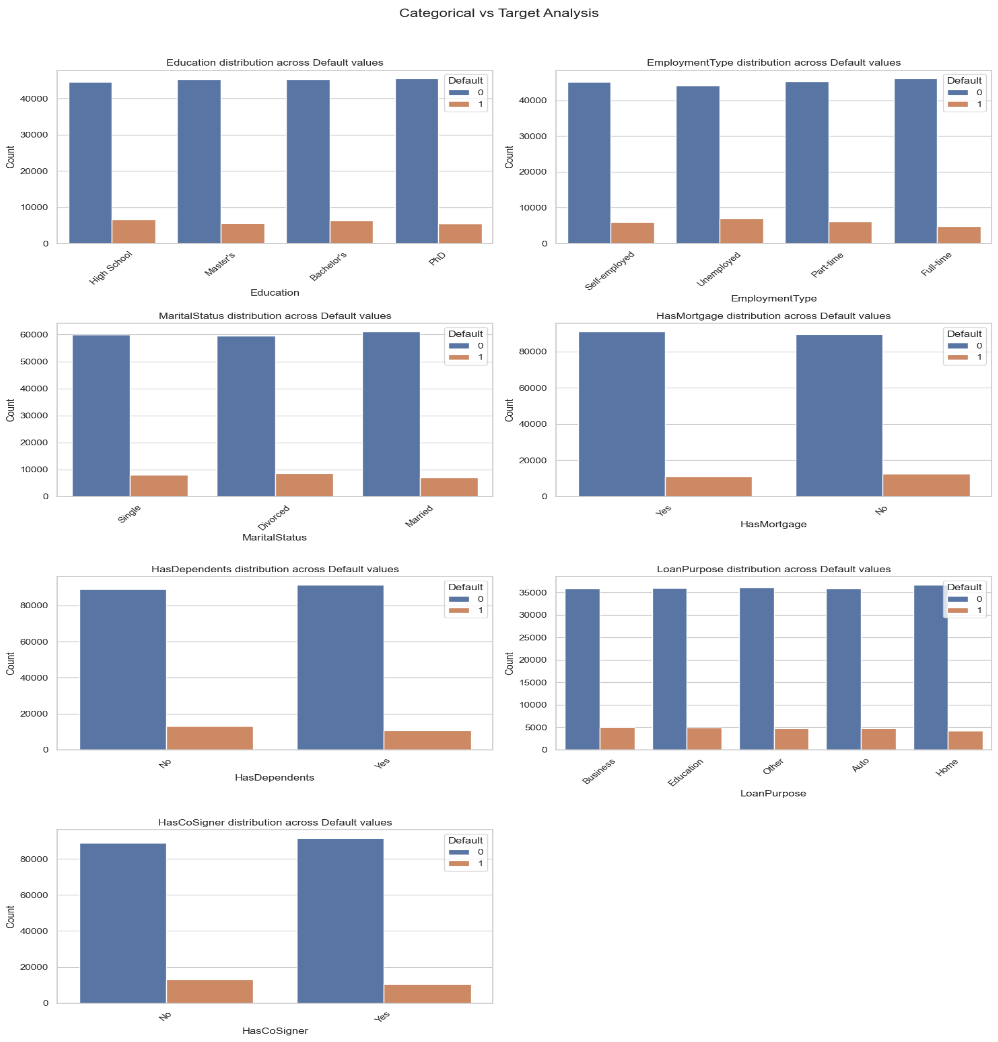

Figure 9: Relationship between target and categorial features

- **Education**:
  - Across all education levels, the number of non-defaulters (`Default = 0`) is significantly higher than defaulters (`Default = 1`).
  - There does not appear to be a strong relationship between education level and default rate, as the proportions remain fairly consistent across categories.
- **EmploymentType**:
  - Similar to education, there are more non-defaulters in each category, with the proportions of defaulters being relatively similar across employment types.
  - This suggests that employment type may not be a strong predictor of loan default on its own.
- **MaritalStatus**:
  - Each marital status category has a larger number of non-defaulters compared to defaulters.

  - The proportions are consistent across marital statuses, which may indicate a limited relationship between marital status and loan default.

- **HasMortgage**:
  - This plot compares the default rates for individuals with or without a mortgage.
  - Although there are more non-defaulters in each group, there appears to be a slight increase in the proportion of defaulters among those without a mortgage, which might indicate a minor association.

- **HasDependents**:
  - This plot compares individuals with and without dependents.
  - Both categories have significantly more non-defaulters, and the proportion of defaulters does not vary much between the groups, suggesting that having dependents might not strongly influence default risk.

- **LoanPurpose**:
  - This plot shows the default distribution across different loan purposes, such as Business, Education, Other, Auto, and Home.
  - The distribution of non-defaulters is higher in each category, with minor variation among categories, indicating that loan purpose may not significantly impact default likelihood.

- **HasCoSigner**:
  - This plot indicates whether having a co-signer affects default rates.
  - Similar to other features, both categories (having a co-signer or not) show a larger proportion of non-defaulters, with no notable difference in the default rate between the groups.

Overall Interpretation: The categorical features in these plots do not show significant variations in the proportions of defaulters and non-defaulters across different categories. This consistency suggests that these categorical features, individually, may have limited predictive power in determining loan default risk. However, they may still be useful in combination with other features or through more advanced feature engineering and model interactions.

3. Models and Experiments

Models Tested

- **Random Forest Classifier:** Random Forest provided robustness and interpretability due to its ensemble nature and the averaging of multiple decision trees.
  - **Strengths:**
    - Handles overfitting with proper parameter tuning.
    - Works well with moderately sized datasets.
  - **Weaknesses:**
    - Struggles with high cardinality features or imbalanced data.

    - Prone to overfitting without adequate regularization.

  While it performed adequately, it required significant tuning to balance bias and variance effectively.

- **XGBoost Classifier:** XGBoost's ability to capture complex patterns made it a strong candidate.

  - **Strengths:**
    - High predictive accuracy.
    - Efficient with well-tuned parameters.
  - **Weaknesses:**
    - Computationally intensive for hyperparameter tuning.
    - Requires preprocessing for categorical features.

  Despite its potential, it did not outperform CatBoost due to the dataset's characteristics, including categorical features.

- **CatBoost Classifier (Best Performing Model):** CatBoost excelled due to its native support for categorical data and efficient gradient boosting.

  - **Strengths:**
    - Requires minimal preprocessing for categorical features.
    - Robust to overfitting with well-tuned parameters.
    - Efficient and accurate on complex datasets.
  - **Weaknesses:**
    - Computationally intensive compared to simpler models.

  With optimal hyperparameter tuning, CatBoost demonstrated superior performance and robustness, making it the best choice.

- **Logistic Regression (Baseline Model):** Logistic Regression served as a baseline, providing insights into the dataset's linear separability.

  - **Strengths:**
    - Computationally efficient and easy to interpret.
    - Performs well on linearly separable data.
  - **Weaknesses:**
    - Struggles to capture non-linear relationships.
    - Limited predictive power for complex datasets.

  While useful as a baseline, it was unable to compete with more sophisticated models on this dataset.

- **Support Vector Machine (SVM):** Support Vector Machines were evaluated for their ability to handle high-dimensional data and provide robust decision boundaries. PCA was applied for dimensionality reduction due to challenges encountered during training with the original feature set.

  - **Strengths:**
    - Effective in high-dimensional spaces when combined with PCA.
    - Provides clear decision boundaries for binary classification.
  - **Weaknesses:**
    - **Without PCA, the model failed to converge, even after 8 hours of training.**
    - Computationally expensive, particularly with kernels like RBF, for large datasets.
  - **Kernels Tested:**
    - **Linear Kernel:** Achieved the same accuracy as other kernels but required PCA for tractable computation.
    - **RBF Kernel:** Exhibited comparable performance but was computationally more intensive.
    - **Polynomial Kernel:** Produced similar accuracy to other kernels but was slower and less interpretable.

  Despite computational challenges and kernel variations, SVM with PCA yielded consistent accuracy across all tested kernels, highlighting its robustness in the reduced feature space.

- **Artificial Neural Network (ANN):** ANNs were implemented using PyTorch to explore the capabilities of deep learning for binary classification. A robust architecture with dropout layers and an adaptive learning rate scheduler was employed to enhance generalization and stability.

  - **Strengths:**
    - Handles large and complex datasets effectively.
    - Incorporates dropout and L2 regularization to mitigate overfitting.
    - Adaptive learning rate scheduling improved convergence.
  - **Weaknesses:**
    - Computationally expensive and slower to train compared to traditional models.
    - Requires substantial hyperparameter tuning for optimal performance.

  The ANN model, with its ability to learn intricate patterns in the data, proved to be a valuable addition to our classification pipeline, particularly for capturing non-linear relationships.

- **Recurrent Neural Network (RNN):** A bidirectional Long Short-Term Memory (LSTM) model was implemented to leverage the sequential nature of the data for binary classification. Key architectural and training strategies include:

  - **Strengths:**
    - Utilizes bidirectional LSTMs to capture both forward and backward temporal dependencies.
    - Dropout regularization effectively reduces overfitting.
    - Adaptive learning rate scheduling ensures efficient convergence.
  - **Weaknesses:**
    - Computational complexity increases with sequence length and model depth.

        - Sensitive to hyperparameter initialization, requiring meticulous tuning.

  The RNN model exhibited its capability to learn temporal patterns, enhancing the classification pipeline's robustness for dynamic and time-sequential features.

- **Fully Connected Neural Network (FCNN):** A feedforward neural network was designed for binary classification with a focus on both numerical and categorical feature integration. The key elements of this approach include:
    - **Strengths:**
        - Efficient handling of mixed data types using preprocessing pipelines for standardization and passthrough strategies.
        - Implementation of dropout layers to mitigate overfitting during training.
        - Use of rectified linear units (ReLU) for activation, ensuring non-linearity and stability.
    - **Weaknesses:**
        - Limited capability to model temporal dependencies in sequential datasets.
        - Relies on careful preprocessing and scaling of input data for optimal performance.

  The FCNN architecture provides a robust baseline for structured data while balancing complexity and interpretability.

Performance Analysis

- **XGBoost:** Hyperparameter tuning significantly improved XGBoost's accuracy.
    - **Training Accuracy:** 0.8868
    - **Classification Report:**

```
              precision    recall  f1-score   support

           0       0.89      1.00      0.94    180524
           1       0.65      0.06      0.11     23753

    accuracy                           0.89    204277
   macro avg       0.77      0.53      0.52    204277
weighted avg       0.86      0.89      0.84    204277
```

    - **Confusion Matrix:**

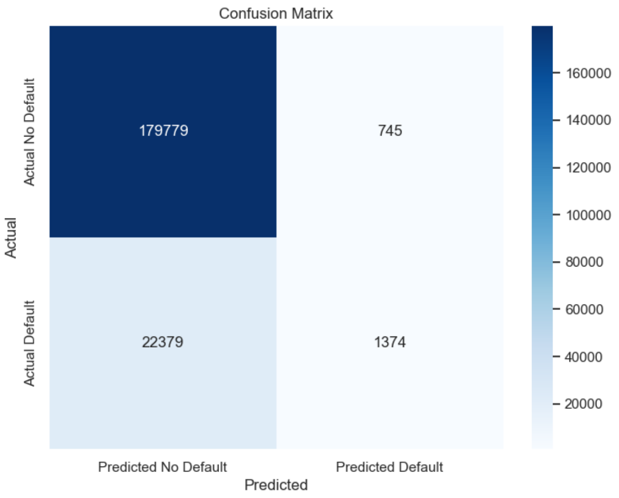

Figure 10: Confusion Matrix for XGBoost

- **Random Forest:** Although Random Forest achieved strong accuracy, it was outperformed by XGBoost due to XGBoost's boosting mechanism. Random Forest was still valuable for its interpretability and ensemble approach.
  - **Training Accuracy:** 0.8887
  - **Classification Report:**

```
              precision    recall  f1-score   support

           0       0.89      1.00      0.94    135381
           1       0.95      0.05      0.09     17826

    accuracy                           0.89    153207
   macro avg       0.92      0.52      0.51    153207
weighted avg       0.90      0.89      0.84    153207
```

  - **Confusion Matrix:**

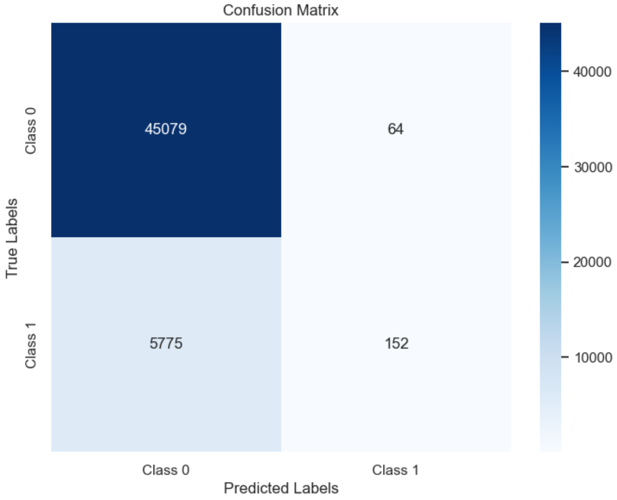

Figure 11: Confusion Matrix for Random Forest

- **CatBoost:** The CatBoost model demonstrated competitive performance, particularly with categorical data. With `iterations=500`, `depth=4`, and `learning_rate=0.05`, CatBoost showed strong accuracy, making it suitable for scenarios where categorical data plays a critical role.
  - **Training Accuracy:** 0.8869
  - **Classification Report:**

```
              precision    recall  f1-score   support

           0       0.89      1.00      0.94    180524
           1       0.71      0.05      0.09     23753

    accuracy                           0.89    204277
   macro avg       0.80      0.52      0.51    204277
weighted avg       0.87      0.89      0.84    204277
```

  - **Confusion Matrix:**

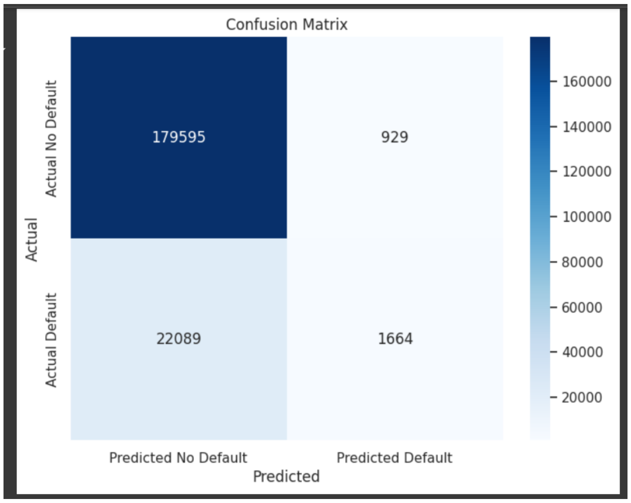

Figure 12: Confusion Matrix for CatBoost

- **Logistic Regression:** Logistic Regression performed reasonably well but struggled with classifying the minority class (1). The model showed a high recall for class 0, but low recall for class 1.
  - **Training Accuracy:** 0.8838
  - **Classification Report:**

```
              precision    recall  f1-score   support

           0       0.88      1.00      0.94    135381
           1       0.70      0.00      0.01     17826

    accuracy                           0.88    153207
   macro avg       0.79      0.50      0.47    153207
weighted avg       0.86      0.88      0.83    153207
```

  - **Confusion Matrix:**

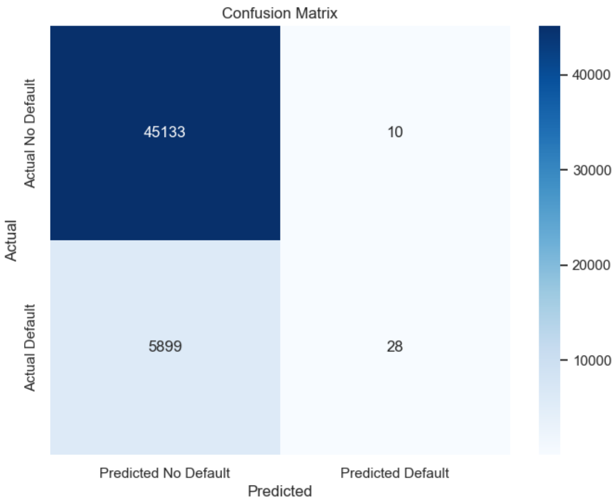

Figure 13: Confusion Matrix for Logistic Regression

- **Fully Connected Neural Network (FCNN):** The FCNN achieved an overall accuracy of 0.8845. However, similar to Logistic Regression, it struggled with the minority class (Default), showing very low recall.
  - **Accuracy:** 0.8845
  - **Classification Report:**

```
              precision    recall  f1-score   support

 Not Default       0.89      1.00      0.94     36073
     Default       0.62      0.04      0.07      4783

    accuracy                           0.88     40856
   macro avg       0.75      0.52      0.50     40856
weighted avg       0.86      0.88      0.84     40856
```

  - **Confusion Matrix:**

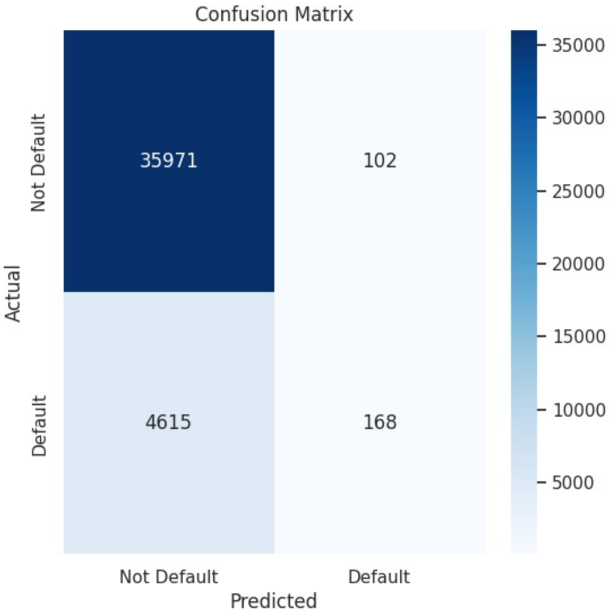

Figure 14: Confusion Matrix for Fully Connected Neural Network

- **Artificial Neural Network (ANN):** The ANN model achieved an accuracy of 0.8836. It showed slightly better recall for the minority class compared to the FCNN but still struggled with classification balance.
  - **Accuracy:** 0.8839
  - **Classification Report:**

```
               precision    recall  f1-score   support

    Not Default       0.88      1.00      0.94     36073
        Default       0.64      0.03      0.06      4783

    accuracy                           0.88     40856
   macro avg       0.76      0.52      0.50     40856
weighted avg       0.86      0.88      0.84     40856
```

  - **Confusion Matrix:**

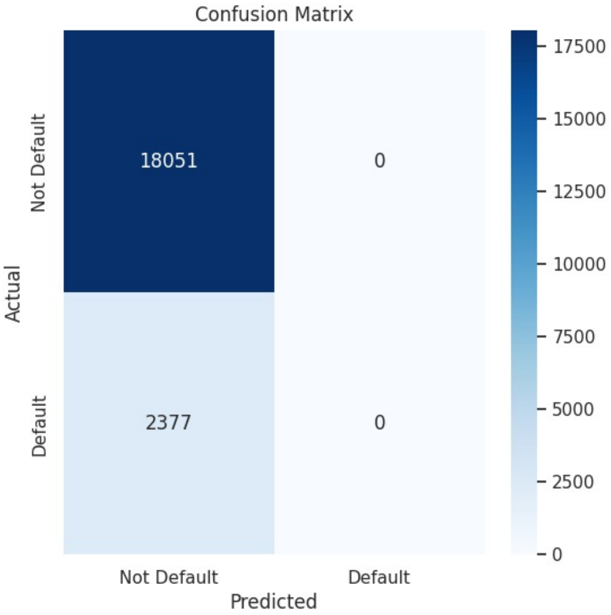

Figure 15: Confusion Matrix for Artificial Neural Network

- **Recurrent Neural Network (RNN):** The RNN achieved an accuracy of 0.8836. Despite being capable of capturing sequential data patterns, it underperformed in classifying the minority class.
  - **Accuracy:** 0.8812
  - **Classification Report:**

```
               precision    recall  f1-score   support

    Not Default       0.88      1.00      0.94     36073
        Default       0.59      0.02      0.03      4783

    accuracy                           0.88     40856
   macro avg       0.73      0.51      0.48     40856
weighted avg       0.86      0.88      0.84     40856
```

  - **Confusion Matrix:**

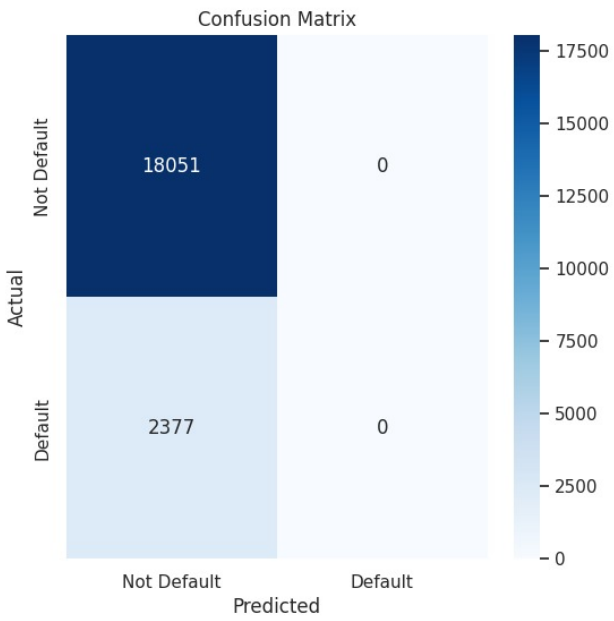

Figure 16: Confusion Matrix for Recurrent Neural Network

We used PCA for faster training

- **Support Vector Machine (SVM) - Linear Kernel:** The SVM with a linear kernel achieved an accuracy of 0.8829. It performed similarly to the ANN, with balanced precision but poor recall for the minority class.
  - **Accuracy:** 0.8829
  - **Classification Report:**

```
               precision    recall  f1-score   support

    Not Default       0.88      1.00      0.94     36073
        Default       0.61      0.02      0.04      4783

    accuracy                           0.88     40856
   macro avg       0.74      0.51      0.49     40856
weighted avg       0.86      0.88      0.84     40856
```

- **Support Vector Machine (SVM) - RBF Kernel:** The SVM with an RBF kernel achieved an accuracy of 0.8829. It showed slightly better results compared to the linear kernel but still struggled with recall for the minority class.
  - **Accuracy:** 0.8829
  - **Classification Report:**

```
               precision    recall  f1-score   support

    Not Default       0.88      1.00      0.94     36073
```

```
        Default       0.60      0.02      0.04      4783

    accuracy                           0.88     40856
   macro avg       0.74      0.51      0.49     40856
weighted avg       0.86      0.88      0.84     40856
```

- **Support Vector Machine (SVM) - Polynomial Kernel:** The SVM with a polynomial kernel achieved an accuracy of 0.8829. While it performed well for the majority class, it struggled the most with the minority class recall.
  - **Accuracy:** 0.8829
  - **Classification Report:**

```
               precision    recall  f1-score   support

    Not Default       0.88      1.00      0.94     36073
        Default       0.58      0.01      0.02      4783

    accuracy                           0.88     40856
   macro avg       0.73      0.51      0.48     40856
weighted avg       0.86      0.88      0.84     40856
```

4. Results and Predictions

Each model generated predictions on the test dataset, with the **CatBoost model** providing the most reliable output. The predictions were saved in CSV files, containing default probability predictions for each loan ID. The **CatBoost model** achieved an accuracy of 0.88811 on the Kaggle submission.

In part 2 , the models used include:

- **SVM** (linear, rbf, and poly)
- **RNN**
- **ANN**
- **FCNN**

The best accuracy score now comes from the **FCNN model**, with a score of 0.88678.