Available online at http://iorajournal.org/indx.php/orics /index

Operations Research: International Conference Series

e-ISSN: 2722-0974
p-ISSN: 2723-1739

Vol. 6, No. 2, pp. 73-89, 2025

Multi-Algorithm Machine Learning Approach for Loan Risk Prediction: Comparative Analysis of Performance of Different Models

Hans Steven .V. Sinaga[1*]

[1]*Department of Mathematics, Universitas Padjadjaran, Sumedang, Indonesia*

**Corresponding author email: hans21001@mail.unpad.ac.id*

Abstract

This study focuses on predicting the probability of customer default in consumer loan products using historical customer behavior data. The dataset includes information such as income, age, work experience, marital status, home ownership, car ownership, and others. Several techniques such as Principal Component Analysis (PCA) are used to reduce the dimensions of correlated features. Various machine learning algorithms are applied, such as Logistic Regression, K-Nearest Neighbors (KNN), Random Forest, XGBoost, and others. The results show that the XGBoost model provides the best performance with the highest AUC Score even though Random Forest provides the best accuracy performance.

Keywords: Finance, machine learning, loan risk, prediction models

1. Introduction

In the financial industry, risk management is a critical aspect that determines business continuity. One of the major risks faced by financial institutions is credit risk, which refers to the possibility of customers failing to meet their payment obligations. Failure to repay a loan as agreed, or credit default, creates major challenges for both borrowers and banks. The negative impacts include the risk of loss of collateral, legal issues, decreased credit scores, and reduced access to future credit facilities for borrowers (Domeher & Abdulai, 2012). On the other hand, banks face losses in revenue and capital, increased costs for supervision and recovery, and decreased profitability and sustainability due to delinquent loans.

As technology advances, traditional methods of assessing risk, such as manual assessment based on past financial records and customer behavior, are becoming inefficient. The large volume of data and the complexity of variables that influence credit decisions require a more sophisticated approach. In recent years, machine learning has emerged as an effective approach to addressing credit risk prediction. By leveraging machine learning algorithms, financial institutions can automate the risk evaluation process, identify patterns from historical data, and make faster and more accurate decisions.

This study aims to help financial organizations predict the likelihood of customer default based on historical data on their behavior. The dataset used includes various demographic and financial attributes of customers, such as income, age, work experience, marital status, home ownership, car ownership, and geographic location. In addition, additional information such as the number of years working in the current job and the length of time living in the house are also included in the variables analyzed.

The main challenge in this research is class imbalance, where most customers do not default (class 0) while only a small portion defaults (class 1). This imbalance can affect the performance of the predictive model because the model tends to be more accurate in predicting the majority class. Therefore, imbalanced data handling techniques, such as Synthetic Minority Over-sampling Technique (SMOTE), are used to improve the class distribution and improve model accuracy.

In addition, proper data processing is essential to obtain optimal prediction results. Data pre-processing steps include checking for missing data, data visualization to understand the distribution and relationships between variables, and outlier handling. Principal Component Analysis (PCA) is also applied to reduce multicollinearity between strongly correlated features, such as work experience and number of years working in the current job. This dimensionality reduction aims to improve the efficiency and accuracy of the model by eliminating irrelevant or redundant features.

This study uses several machine learning algorithms to predict whether a customer will default on a loan or not. Some of the algorithms tested include Logistic Regression, K-Nearest Neighbors (KNN), Decision Tree, Random Forest, and XGBoost. Each algorithm is selected based on its ability to handle complex data, overcome class imbalance, and efficiency in making predictions. Model evaluation is carried out based on several metrics, including accuracy, precision, recall, and area under the curve (AUC) of the Receiver Operating Characteristic (ROC), which provide a complete picture of the performance of each algorithm.

Through this research, it is expected to find the most effective prediction model to help financial institutions identify high-risk customers more accurately. In addition, the results of this study can provide guidance for organizations in choosing the right machine learning algorithm according to the characteristics of the dataset they have. That way, credit risk management can be carried out more efficiently, reducing the possibility of financial losses, and increasing profitability and long-term business stability.

2. Literature Review

2.1 Machine Learning

Machine learning has revolutionized the financial world with its ability to automate and improve decision-making based on historical data and current trends. In the digital era, financial institutions are faced with a huge increase in the volume and variety of data, including transaction data, user behavior, and external data such as macroeconomic conditions. In this situation, traditional statistical-based models such as linear and logistic regression often struggle to handle the complexity of modern data. ML enables deeper analysis through the use of algorithms that are able to identify hidden patterns and non-linear relationships. This is especially important in the financial sector, especially in loan risk assessment, as the accuracy and efficiency of predictive models can have a direct impact on a company's profitability and stability. According to Bengio et al. (2017), the implementation of ML-based models also enables financial companies to increase data processing speed and enhance automation in risk evaluation, which ultimately reduces loan processing time and increases customer satisfaction.

Unsupervised learning methods are also used in the financial sector, especially to detect anomalies or suspicious activity in transaction patterns, which are related to credit risk and fraud. Techniques such as clustering and dimensionality reduction can help in understanding borrower segments or detecting different behavior patterns between high and low risk borrowers.

2.2 Credit Risk

Credit risk is one of the greatest financial risks faced by financial institutions. This risk relates to the possibility that borrowers will fail to meet their financial obligations. Historically, the traditional approach to assessing loan risk involved analyzing quantitative variables such as credit history, income, debt-to-income ratio, and qualitative variables such as borrower employment and stability.

However, in the modern world, there are more and more variables to consider, including unstructured data such as user behavior on social media, digital transaction behavior, to data from third-party platforms such as e-commerce. This approach drives research to find the best way to combine these various data sources to enrich loan risk analysis. According to Moro et al (2015), this non-traditional data, if processed properly, can increase accuracy in predicting credit risk, because this data better reflects the borrower's actual behavior in managing daily finances.

Other risk factors include macroeconomic conditions, such as changes in interest rates, inflation, and labor market conditions. Fluctuations in these factors can have a significant impact on a borrower's ability to repay a loan. For example, economic recessions often lead to an increase in loan default rates, which is a systemic risk for financial institutions (Beck et al., 2013).

2.3 Loan Prediction Model

ML-based prediction models have the ability to analyze more complex and diverse data compared to traditional models. Some algorithms used in credit risk prediction include Logistic Regression, Decision Tree, Random Forest, KNN, and XGBoost. Each of these algorithms has its own advantages and disadvantages.

2.3.1 Logistic Regression

Logistic regression is a basic model used in many loan risk prediction applications. It works by predicting the probability of a binary outcome (e.g., whether a loan will default or not). The main advantages of logistic regression are its simplicity and high interpretability. It provides odds ratios that are easy to understand and is often the choice when a transparent and easy-to-explain model is needed (Hosmer & Lemeshow, 2000). However, logistic regression has limitations in handling very complex and nonlinear data.

2.3.2 Decision Tree

Decision treeis a model that divides data based on key features that influence the outcome, with an iterative splitting approach until a final decision is reached. The advantage of this model is its interpretability, where each decision can be traced through the decision tree path. However, decision tree models are often prone to overfitting if they are too deep (Breiman, 2001).

2.3.3 Random Forest

*Random forest*developed by combining multiple decision trees to improve the stability and accuracy of predictions. Each tree in a random forest is trained on a different subset of the data, resulting in a model that is more robust to data variability. Research by Lessmann et al (2015) shows that random forest provides very good results in credit risk prediction with a high level of accuracy and is good against overfitting.

2.3.4 KNN

K-Nearest Neighbors (KNN) is a supervised learning algorithm used for classification and regression tasks. This algorithm works based on the principle of similarity, where classification or prediction is done by looking at a number of nearest neighbors of the data point to be predicted. The data point is then classified according to the majority of labels from its nearest neighbors.

2.3.5 XGBoost

Gradient boosting is an ensemble method that combines multiple simple models, usually decision trees, to improve prediction accuracy. With this approach, the models are built incrementally, with each new model trying to improve the prediction errors of the previous model. This process continues until the model reaches a desired level of accuracy, making gradient boosting very effective in handling complex prediction tasks. The XGBoost algorithm, an optimized version of GBM, has become popular due to its excellent ability to handle large and diverse data and better computational speed (Chen & Guestrin, 2016).

3. Materials and Methods

3.1. Materials

Data: The dataset used in this study was obtained from Kaggle.com. This dataset includes various features relevant to predicting loan risk, including attributes such as location (CITY), customer information, and other variables that determine the risk profile.

Tools and Libraries Used: This research was conducted using the Python programming language with several main libraries as follows:

a) **Numpy**: For numerical computations and array-based operations.
b) **Panda**: For data manipulation and analysis in DataFrame format.
c) **Matplotlib and Seaborn**: For data visualization that includes graphs such as histograms, boxplots, and heatmaps.
d) **Scikit-learn**: Used for machine learning algorithm implementation, preprocessing, and model evaluation. This library includes various techniques such as StandardScaler, MinMaxScaler, PCA, and model evaluation metrics (accuracy, precision, recall, F1-score, etc.).
e) **Imbalanced learning (SMOTE)**: Used to handle data imbalance with an oversampling technique called SMOTE (Synthetic Minority Over-sampling Technique).
f) **Scipy**: For statistical tests, such as the chi-squared test.
g) **Google Colab (files and gdown)**: Used to download data from Google Drive and integrate it into the notebook environment.

3.2. Methods

a) **Data collection**:

The first stage in this method is data collection. The data used in this study were taken from public sources that provide datasets relevant to credit customer behavior, for example from platforms such as Kaggle. This dataset includes various features relevant to predicting the risk of loan default, including demographic attributes such as age, income, marital status, home ownership, car ownership, and work experience. In addition, some additional information such as geographic location, number of years of work, and length of residence at home were also used in the study.

The data collection process is done by utilizing Python libraries such as gdown, which is used to download data from Google Drive or other online sources. The downloaded data is then saved in CSV format and loaded into a

Pandas DataFrame for further manipulation. The use of Pandas is very important in this case because it allows for easier and more efficient data analysis and manipulation. In addition, Pandas supports various functions such as statistical description, data cleaning, and DataFrame-based operations that are very suitable for big data analysis tasks.

Once the data is downloaded, the next step is to perform an initial check on the dataset. This involves exploring the structure of the dataset such as the number of columns, data types, and missing values or outliers. By using the df.info() and df.describe() functions, researchers can understand the distribution of variables in the dataset. This information is important for determining the next steps of data processing, including data cleaning and normalization steps.

In addition, at this stage, researchers also check the quality of the data. Missing data or null values must be identified and treated appropriately, for example by imputation methods such as filling in the average or median value. In more extreme cases, rows with missing values can be removed if the percentage of missing data is too high. Once the data is ready, the next stage is to pre-process it to ensure that the data is ready to be used in the machine learning model.

b) Data Exploration and Preprocessing:

Once the data is collected, the next step is to explore the data. This stage is important to gain initial insight into how the variables in the dataset relate to each other, as well as the distribution of each variable. Data visualization techniques are used to provide a clearer picture of the dataset. For example, histograms are used to see the distribution of numeric features such as income and age, while bar plots can be used to understand the proportion of categorical data such as marital status or car ownership.

In addition to visualization, researchers also need to examine the correlation between variables in the dataset. Correlation between variables can be seen using a heatmap or correlation matrix, which shows the extent to which two variables have a linear relationship with each other. This is important in deciding whether to perform dimensionality reduction using techniques such as Principal Component Analysis (PCA) to reduce multicollinearity among highly correlated features. This can help improve model efficiency by removing less relevant features.

In this section, handling of missing data should also be considered. Missing data or outliers can significantly affect model performance. To overcome this, techniques such as filling missing values with the mean or median are used, depending on the type and distribution of the data. If the amount of missing data is too large, it can be imputed using more complex algorithms such as K-Nearest Neighbors Imputation, or even discarding the data if it does not affect the overall results too much.

In addition, numeric features often require normalization or standardization so that machine learning models can process the data optimally. Features such as income or age, which have very different ranges of values, need to be transformed to the same scale using techniques such as StandardScaler or MinMaxScaler. This will help algorithms, especially those sensitive to data scale such as KNN or Logistic Regression, to produce more accurate predictions.

c) Data Sharing:

The next step is to divide the dataset into two main parts: training data and testing data. The purpose of this division is to test the performance of the model on data that has never been seen before, thus providing an indication of the model's performance when applied to new data. In this study, the dataset was divided into 70% for training and 30% for testing, although this ratio can vary depending on the size of the dataset and the purpose of the study.

Data splitting is done using the train_test_split function from the Scikit-learn library, which randomly splits the dataset into two parts. The training data is used to train the model, while the test data is used to evaluate the model. This process is critical to ensure that the model does not overfit to the training data, meaning the model learns too much from the training data and cannot generalize to new data.

Before dividing the data, researchers should also pay attention to the distribution of target classes in the dataset. Class imbalance is a common problem in credit risk prediction, where the majority of customers do not default. To overcome this problem, methods such as oversampling with SMOTE (Synthetic Minority Over-sampling Technique) or undersampling in the majority class can be used. This technique aims to improve the class distribution so that the model is not biased towards the majority class.

Cross-validation is also applied to ensure that the results of data partitioning are independent of how the data is divided. This technique divides the dataset into several subsets and trains the model multiple times with different combinations of training and testing data. Thus, cross-validation helps reduce the risk of overfitting and provides a more stable picture of the model's performance.

d) Machine Learning Algorithms:

Several machine learning algorithms are used in this study to build prediction models. Each algorithm has different characteristics and advantages in handling complex data and class imbalance.

a. **Logistic Regression**: This model is used to predict the probability of a binary outcome, such as whether a customer will default or not. Logistic Regression is often chosen because of its simplicity and high interpretability. However, this model has limitations in handling non-linear relationships between variables.
b. **K-Nearest Neighbors (KNN)**: This algorithm works on the principle of similarity between data, where new data is classified based on data that has the closest distance. KNN is often chosen because of its ease of implementation and its effectiveness for datasets that are not too large. However, KNN is prone to scalability issues and slow performance on large datasets.
c. **Decision Tree**: This algorithm builds a decision tree based on the main features that affect the prediction results. Decision Tree is very intuitive and easy to interpret, but is prone to overfitting if not pruned properly. To overcome this, more sophisticated versions such as Random Forest are often used.
d. **Random Forest**: This is an ensemble method that combines multiple Decision Trees to improve model accuracy and stability. Random Forest is known for its ability to handle complex data and tends to be more resistant to overfitting than a single Decision Tree.
e. **XGBoost**: This algorithm is a more efficient version of gradient boosting, which aims to minimize prediction errors by building a model incrementally. XGBoost is known for its excellent performance on large and complex data, as well as its ability to handle class imbalance.

e) **Model Evaluation: Model performance is assessed using several key metrics, including:**

1) **Accuracy**: Calculates the percentage of correct predictions from all data.
2) **Precision**: Shows how accurate the model is in predicting the positive class.
3) **Recall**: Assesses the ability of the model to detect all examples from the positive class.
4) **F1-Score**: Is the average between precision and recall, providing a balance between the two.
5) **ROC Curve**illustrates the relationship between True Positive Rate (TPR) or recall and False Positive Rate (FPR) at various thresholds. A good ROC Curve will be closer to the upper left corner, indicating that the model is able to maximize positive detection while minimizing false positive errors.
6) **AUC (Area Under the Curve)**is the area under the ROC curve. The AUC value ranges between 0 and 1. The higher the AUC value, the better the model's performance in distinguishing between the classes. A value of 0.5 indicates that the model is no better than random guessing, while a value of 1 indicates that the model is perfect in distinguishing between the two classes.

Once the models are trained using the training data, the performance of each model is evaluated using various metrics. These metrics provide a complete picture of how well the model is at predicting the risk of loan default.

f) **Comparative Analysis**: The results of various algorithms are compared to find the most effective model in predicting loan risk based on the data used. The performance of each model is analyzed and presented in graphs and tables for easy interpretation.
g) **Interpretation of Results**: Shows which model is best at predicting risk.

3.2.1. Formula / Equation

a. Logistic Regression

$$ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \cdots + \beta_n X_n \quad (1)$$

$\boldsymbol{ln\left(\frac{p}{1-p}\right)}$	: logit of the dependent variable (i.e. the probability of success divided by the probability of failure).
$\boldsymbol{\beta_0}$	: constant (intercept) of the model
$\boldsymbol{\beta_1, \beta_2, \beta_3, \dots, \beta_n}$	**:**regression coefficient of each independent variable
$\boldsymbol{X_1, X_2, X_3, \dots, X_n}$	**:**independent variables used in the model

Implementation in Python:

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter = 500000)
model.fit(X_res, y_res)
y_pred = model.predict(X_test)
accuracy = model.score(X_test, y_test)
accuracy
```

b. Decision Tree

$$Entropy(S) = -\sum_{i=1}^{n} p_i \times log_2\, p_i \tag{2}$$

$$Gain(S, A) = Entropy(S) - \sum_{i=1}^{n} p(x) \times Entropy(Si) \tag{3}$$

Implementation in Python:

```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(criterion="entropy",random_state=420)
model.fit(X_res, y_res)
y_pred = model.predict(X_test)
accuracy = model.score(X_test, y_test)
accuracy
```

c. Random Forest

A combination of several Decision Trees.
Implementation in Python:

```
from sklearn.ensemble import RandomForestClassifier

gr_rf_model = RandomForestClassifier(class_weight="balanced", random_state=101, max_depth=12,
n_estimators= 200, min_samples_split = 3)
gr_rf_model.fit(X_train,y_train)

rf_feature_imp = pd.DataFrame(index = X.columns, data = gr_rf_model.feature_importances_,
                        columns = ["Feature Importance"]).sort_values("Feature Importance", ascending =
False)
rf_feature_imp

new_df = df2[["Income", "CITY", "Age", "Profession", "STATE", "PC2", "PC1",
"CURRENT_HOUSE_YRS", "Risk_Flag"]]
X_new = new_df.drop('Risk_Flag',axis=1)
y_new = new_df['Risk_Flag']
X_train_new, X_test_new, y_train_new, y_test_new = train_test_split(X_new, y_new, test_size = 0.3,
stratify=y, random_state = 101)

rf_model = RandomForestClassifier(class_weight="balanced", random_state=101)
rf_model.fit(X_train_new,y_train_new)
y_pred_new = rf_model.predict(X_test_new)
accuracy_rf = rf_model.score(X_test_new, y_test_new)
accuracy_rf
```

d. K-Nearest Neighbors

KNN works by classifying a data point based on the majority of its neighboring classes. The distance between points is calculated using the Euclidean distance:

$$d = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2\ )} \tag{4}$$

Where d is the distance between two points in n-dimensional space.
Implementation in Python:

```
from sklearn.neighbors import KNeighborsClassifier
# Initialize the KNN model
model = KNeighborsClassifier(n_neighbors=5)
# Training the model
model.fit(X_res, y_res)
# Prediction with test data
y_pred = model.predict(X_test)
# Calculating model accuracy
accuracy = model.score(X_test, y_test)
accuracy
```

e. XGBoost

XGBoost uses gradient boosting technique to combine multiple decision trees by correcting the errors of previous models. The general formula is as follows:

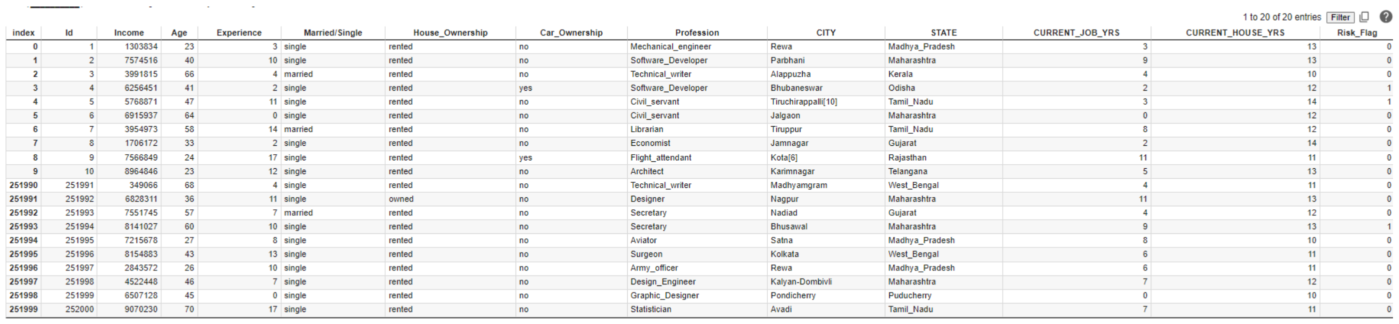
$$y_i = \sum_{k=1}^{K} f_k(x_i) \tag{5}$$

Where is the weak learner function (usually a decision tree) and is the number of trees. $f k\ K$
Implementation in Python:

```
from xgboost import XGBClassifier

model =XGBClassifier(learning_rate=0.1,n_estimators=1000,use_label_encoder=False,random_state=420)
model.fit(X_res, y_res)
y_pred = model.predict(X_test)
accuracy = model.score(X_test, y_test)
accuracy
```

Table 1: Dataset



1 to 20 of 20 entries Filter

index	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	Profession	CITY	STATE	CURRENT_JOB_YRS	CURRENT_HOUSE_YRS	Risk_Flag
0	1	1303834	23	3	single	rented	no	Mechanical_engineer	Rewa	Madhya_Pradesh	3	13	0
1	2	7574516	40	10	single	rented	no	Software_Developer	Parbhani	Maharashtra	9	13	0
2	3	3991815	66	4	married	rented	no	Technical_writer	Alappuzha	Kerala	4	10	0
3	4	6256451	41	2	single	rented	yes	Software_Developer	Bhubaneswar	Odisha	2	12	1
4	5	5768871	47	11	single	rented	no	Civil_servant	Tiruchirappalli[10]	Tamil_Nadu	3	14	1
5	6	6915937	64	0	single	rented	no	Civil_servant	Jalgaon	Maharashtra	0	12	0
6	7	3954973	58	14	married	rented	no	Librarian	Tiruppur	Tamil_Nadu	8	12	0
7	8	1706172	33	2	single	rented	no	Economist	Jamnagar	Gujarat	2	14	0
8	9	7566849	24	17	single	rented	yes	Flight_attendant	Kota[6]	Rajasthan	11	11	0
9	10	8964846	23	12	single	rented	no	Architect	Karimnagar	Telangana	5	13	0
251990	251991	349066	68	4	single	rented	no	Technical_writer	Madhyamgram	West_Bengal	4	11	0
251991	251992	6828311	36	11	single	owned	no	Designer	Nagpur	Maharashtra	11	13	0
251992	251993	7551745	57	7	married	rented	no	Secretary	Nadiad	Gujarat	4	12	0
251993	251994	8141027	60	10	single	rented	no	Secretary	Bhusawal	Maharashtra	9	13	1
251994	251995	7215678	27	8	single	rented	no	Aviator	Satna	Madhya_Pradesh	8	10	0
251995	251996	8154883	43	13	single	rented	no	Surgeon	Kolkata	West_Bengal	6	11	0
251996	251997	2843572	26	10	single	rented	no	Army_officer	Rewa	Madhya_Pradesh	6	11	0
251997	251998	4522448	46	7	single	rented	no	Design_Engineer	Kalyan-Dombivli	Maharashtra	7	12	0
251998	251999	6507128	45	0	single	rented	no	Graphic_Designer	Pondicherry	Puducherry	0	10	0
251999	252000	9070230	70	17	single	rented	no	Statistician	Avadi	Tamil_Nadu	7	11	0

Table 2: Prediction Performance Measurement

Evaluation Metrics	Formula

Accuracy Score	$\frac{TP+TN}{TP+TN+FP+FN}$
Precision Score	$\frac{TP}{TP+FP}$
Recall Score (Sensitivity)	$\frac{TP}{TP+FN}$
F1 Score	$\frac{2\times Precision \times Recall}{Precision + Recall}$
Support	Number of instances
Macro Avg	Mean of Precision, Recall, F1 across classes
Weighted Avg	Weighted mean of Precision, Recall, F1 across classes
ROC-AUC Score	$\frac{\sum(Rank\ of\ Positive\ Samples) - \frac{n_{positive}\times(n_{positive}+1)}{2}}{n_{positive}\times n_{negative}}$

Note: TP—true positive, TN—true negative, FP—false positive, FN—false negative

Table 3: Model Prediction Accuracy Level

Model	Accuracy	Percentage
Logistic regression	0.5328835978835978	53%
KNN	0.8600529100529101	86%
Random Forest	0.8960185185185185	90%
Decision Tree	0.8671957671957672	87%
XGBoost	0.8908862433862433	89%

4. Results and Discussion

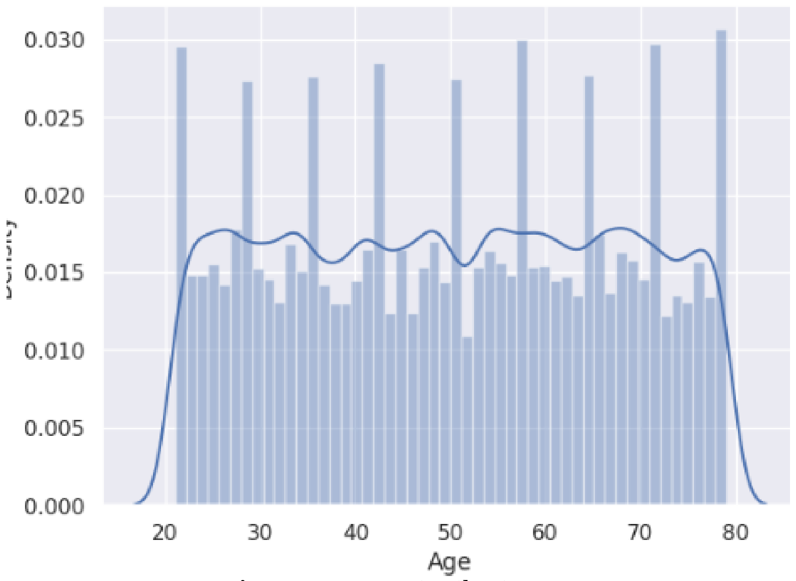
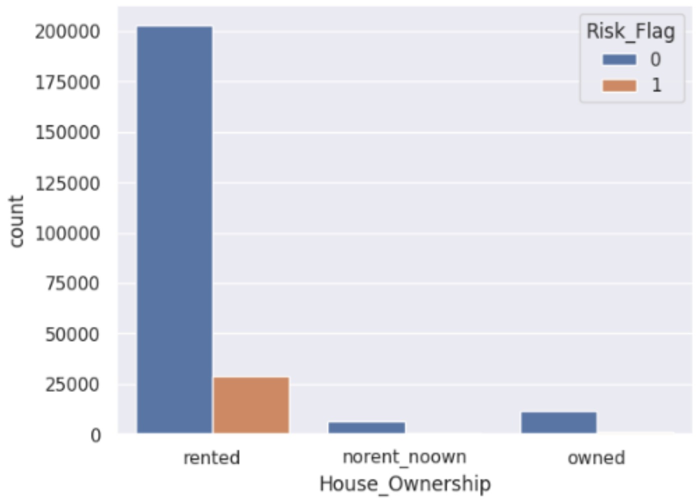


Figure 1: Age Distribution



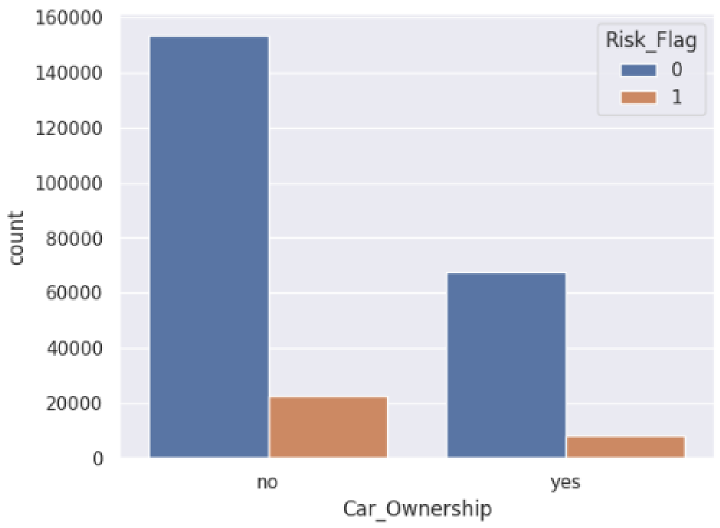
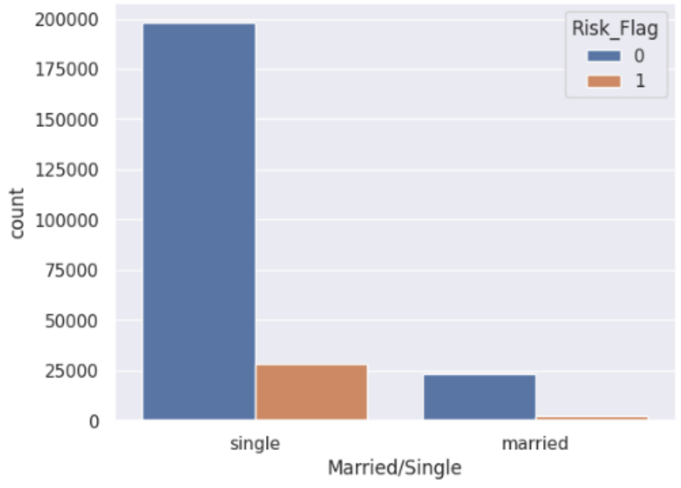


Figure 2: Effect of home ownership with risk flag

Figure 3: Effect of car ownership with risk flag

Figure 4: Effect of marital status on risk flags



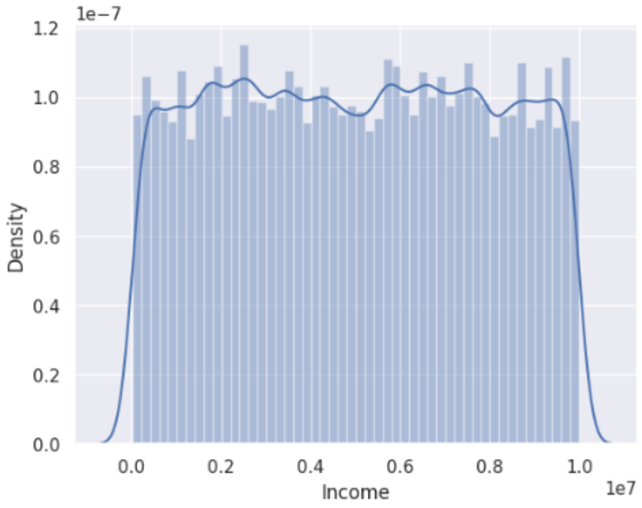


Figure 5: Income distribution

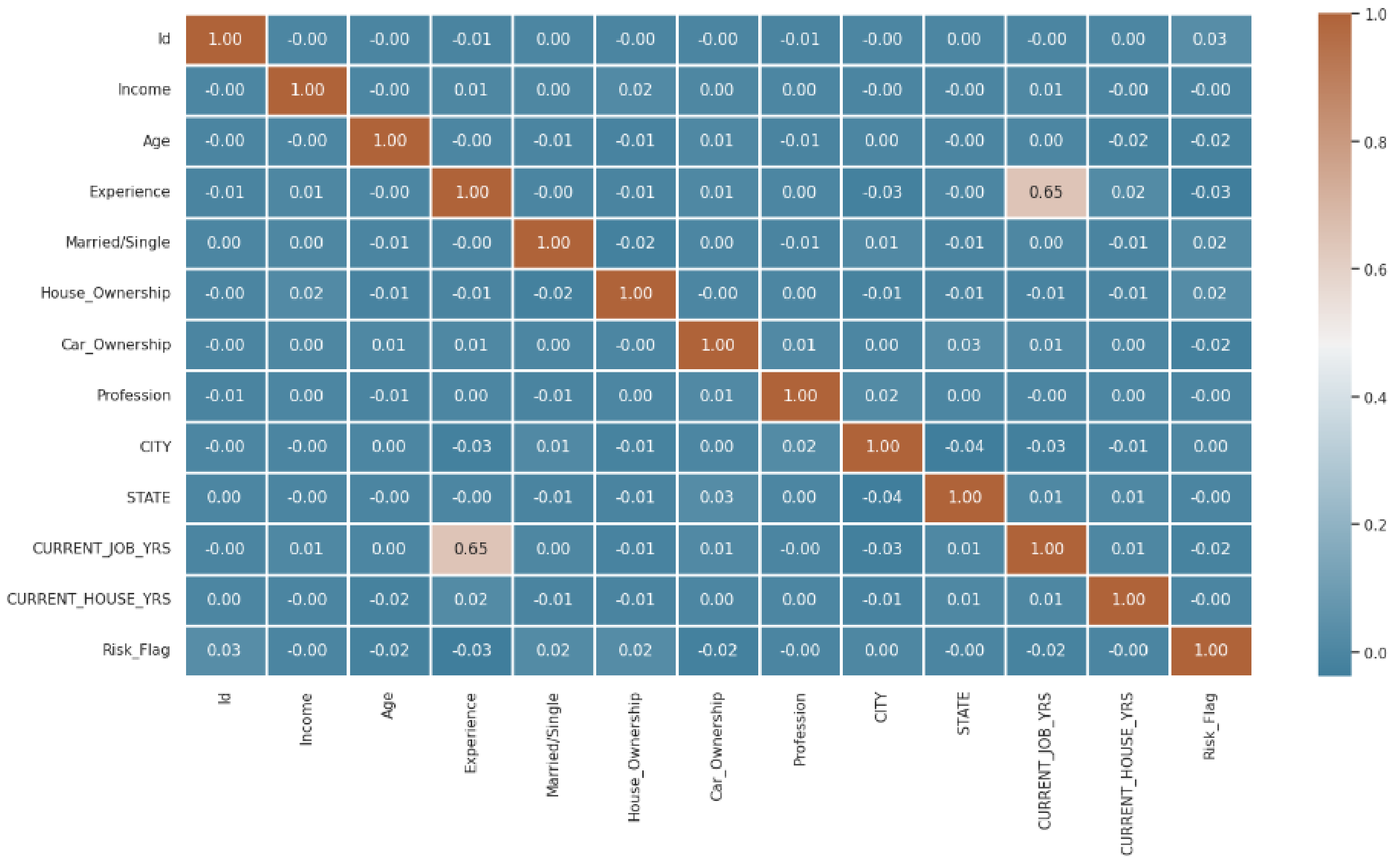


Figure 6: Correlation Matrix

The correlation matrix shows the linear relationship between two variables using the Pearson correlation coefficient, which ranges from -1 (perfect negative correlation) to +1 (perfect positive correlation). In the correlation heatmap, blue indicates a negative or zero correlation, while brown indicates a positive correlation. The darker the color, the stronger the relationship between the two variables. The main diagonal shows a correlation of 1.00, which represents a perfect correlation between the variables and themselves. For example, there is a strong positive correlation (0.65) between years of service in the current job (CURRENT_JOB_YRS) and work experience (Experience), indicating that the longer a person works, the more experience they gain. Meanwhile, the income variable (Income) does not show a significant relationship with the other variables, with a correlation value close to zero. In addition, the target variable (Risk_Flag) also does not show a strong correlation with the other variables, indicating that there may not be a significant linear relationship between these variables and risk prediction. This correlation value approaching zero illustrates the absence of a clear linear relationship between several variables, while a positive relationship between CURRENT_JOB_YRS and Experience is logical, considering that both are related to the duration of work experience.

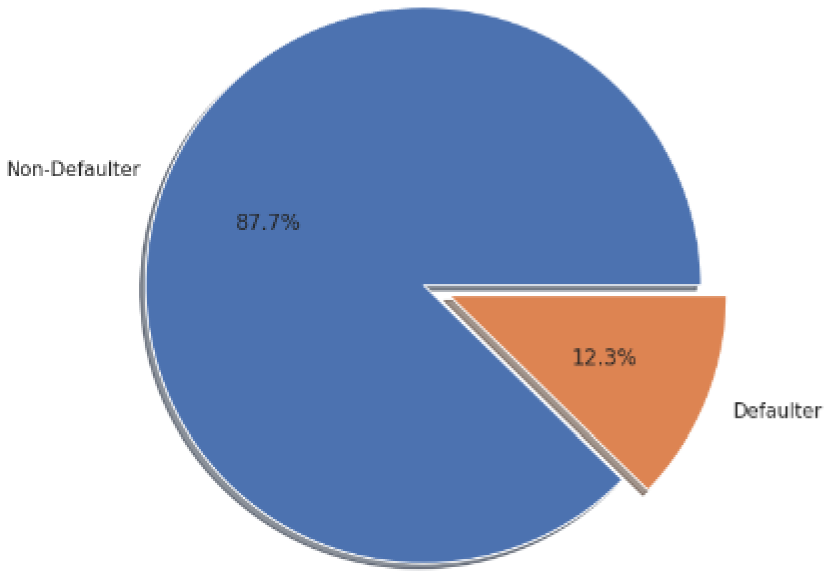


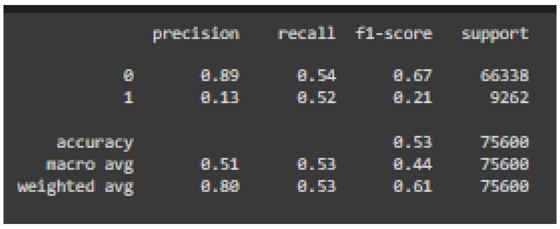
Figure 7 : Outliers

The dataset used shows significant imbalance between classes, where class 0 covers 87.7% of the total data, while class 1 only 12.3%. This imbalance needs to be addressed in order for the model to function properly. In addition, initial analysis shows no outliers in the dataset, which means the data is clean enough for further analysis. However, we still need to measure age and income to understand their impact on credit risk.

The strong correlation between experience and years in the current job (CURRENT_JOB_YRS) suggests information redundancy. Therefore, we may consider removing one of these columns during the feature selection process. Alternatively, we can use Principal Component Analysis (PCA) to reduce the dimensionality of the data without losing important information.

For categorical variables such as marital status (Married/Single), home ownership, and car ownership, we can use binarization or one-hot encoding techniques to make them easier to analyze. To understand the relationship between the target variable and the categorical variable, we can conduct a Chi-square test. With these steps, we can better understand the factors that influence the risk of credit default.

4.1. Logistic Regression



```
              precision    recall  f1-score   support

           0       0.89      0.54      0.67     66338
           1       0.13      0.52      0.21      9262

    accuracy                           0.53     75600
   macro avg       0.51      0.53      0.44     75600
weighted avg       0.80      0.53      0.61     75600
```

Figure 8: Logistic Regression test results

The results of the Logistic Regression model evaluation show that there is a significant difference in performance between the different classes. For class 0, which represents individuals who do not default, the precision reaches 0.89. This means that 89% of all predictions for this class are correct, indicating that the model is quite good at identifying people who are not at risk. However, the precision for class 1, which represents individuals who are at risk of default, is very low, only 0.13. This indicates that the model has difficulty in detecting individuals who may default, so many of these predictions turn out to be wrong.

In terms of recall, the model was able to identify 54% of individuals who were not in default (class 0), which is relatively low. Meanwhile, the recall for class 1 was 52%, which means the model only managed to capture half of the individuals who were actually at risk of default. Although the recall for class 1 is higher, there are still many individuals who should have been detected that were not identified by the model.

The F1-score shows mixed results; for class 0, the F1-score is 0.67, indicating a fairly good balance between precision and recall. However, for class 1, the F1-score is very low at 0.21, indicating that the model is less effective in identifying defaulters. With an overall accuracy of 53%, the model is only able to correctly predict 53% of the total data, indicating a need for improvement, especially in identifying defaulters. The macro average and weighted average also show that while the model is effective in identifying the majority class, its performance in the minority class still needs improvement.

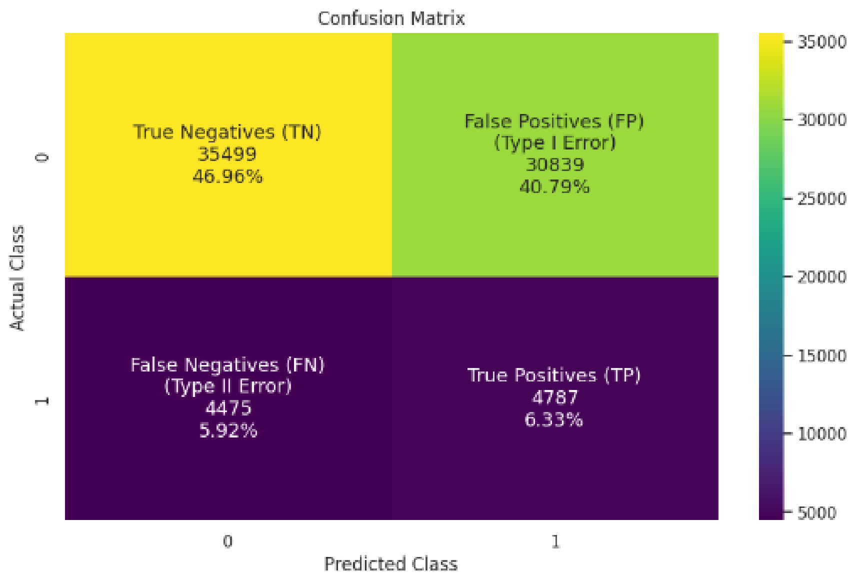
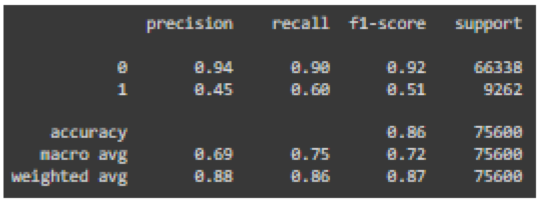


Figure 9: Confusion Matrix Model

4.2. KNN



```
              precision    recall  f1-score   support

           0       0.94      0.90      0.92     66338
           1       0.45      0.60      0.51      9262

    accuracy                           0.86     75600
   macro avg       0.69      0.75      0.72     75600
weighted avg       0.88      0.86      0.87     75600
```

Figure 10: KNN test results

The evaluation results of the K-Nearest Neighbors (KNN) model show quite good performance, especially for class 0, which represents individuals who do not default. The precision for class 0 reaches 0.94, which means that 94% of all predictions for this class are correct. This shows that the model is very effective in identifying individuals who are not at risk. The recall for class 0 is also quite high, which is 0.90, indicating that the model can identify 90% of individuals who are actually not in default. Overall, the F1-score for class 0 is 0.92, reflecting a good balance between precision and recall.

However, for class 1, which indicates individuals at risk of default, the model performance is not as good as class 0. The precision for this class is only 0.45, meaning that only 45% of the predictions for this class are correct. The recall for class 1 is better, reaching 0.60, indicating that the model can identify 60% of individuals who are truly at risk of default. The F1-score for class 1 is 0.51, indicating that although the model is better at detecting at-risk individuals, there is still a lot of room for improvement.

With an overall accuracy of 86%, the KNN model demonstrates a strong ability in predicting credit status. The macro averages for precision, recall, and F1-score are 0.69, 0.75, and 0.72, respectively, indicating that despite good performance in the majority class, the model still faces challenges in recognizing individuals in the minority class. Meanwhile, the weighted averages show higher values: precision 0.88, recall 0.86, and F1-score 0.87. This indicates that the model is quite effective in identifying default risks despite class imbalance.

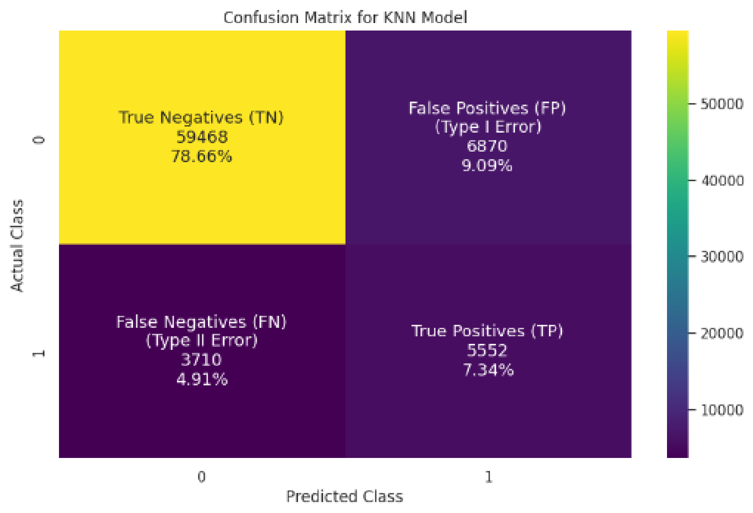
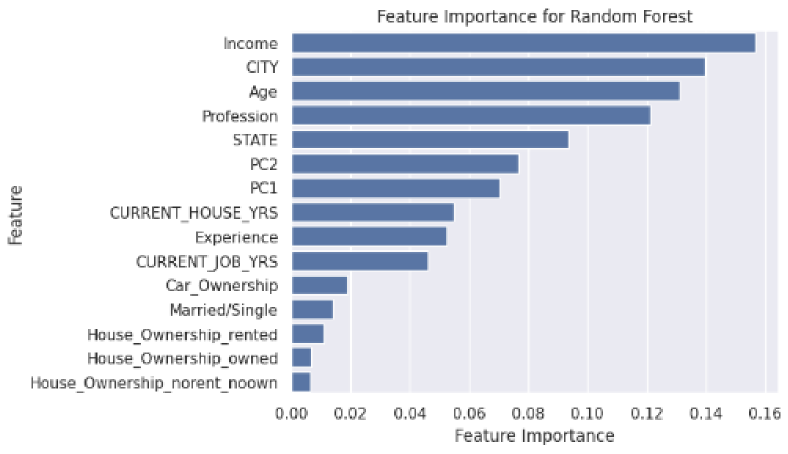


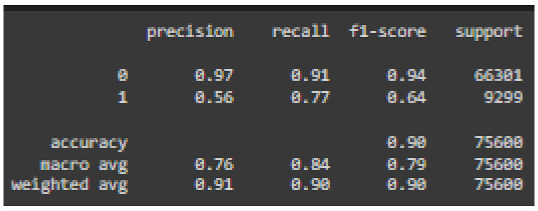
Figure 11: Confusion matrix model



4.3. Random Forest

Figure 12 : Feature Importance

Feature Importance is a measure that shows how much influence each feature (variable) has in predicting the outcome or target in a machine learning model. In the Random Forest model, feature importance is calculated based on how much each feature contributes to reducing uncertainty (impurity) in the model's decision making.



```
              precision    recall  f1-score   support

           0       0.97      0.91      0.94     66301
           1       0.56      0.77      0.64      9299

    accuracy                           0.90     75600
   macro avg       0.76      0.84      0.79     75600
weighted avg       0.91      0.90      0.90     75600
```

Figure 13: Random Forest test results

The evaluation results of the Random Forest model show very good performance in predicting credit status, especially in class 0, which represents individuals who do not default. The precision for this class reaches 0.97, which means that 97% of the predictions for class 0 are correct. With a recall of 0.91, the model successfully detects 91% of individuals who are truly not defaulting. The resulting F1-score for class 0 is 0.94, indicating that the model has a very good balance between precision and recall, making this model very effective in classifying individuals who are not at risk.

For class 1, which are individuals at risk of default, the model also performs quite well, although not as strong as in class 0. The precision for this class is 0.56, indicating that 56% of the predictions for individuals at risk of default are correct. However, the recall for this class is higher, at 0.77, meaning that the model is able to detect 77% of the individuals who are actually at risk of default. The F1-score of 0.64 indicates an improvement in the balance between precision and recall compared to the results of the previous model, but it is still important to note that this model is weaker in detecting defaulting individuals compared to non-defaulting individuals.

Overall, the model accuracy is 90%, indicating that the Random Forest model can predict credit status with fairly high accuracy. The macro averages for precision, recall, and F1-score are 0.76, 0.84, and 0.79, respectively, reflecting strong model performance, but still some gaps in handling minority classes. The weighted average, which takes into account the number of instances in each class, gives a precision of 0.91, recall of 0.90, and F1-score of 0.90. This confirms that the model performs well overall, despite the class imbalance.

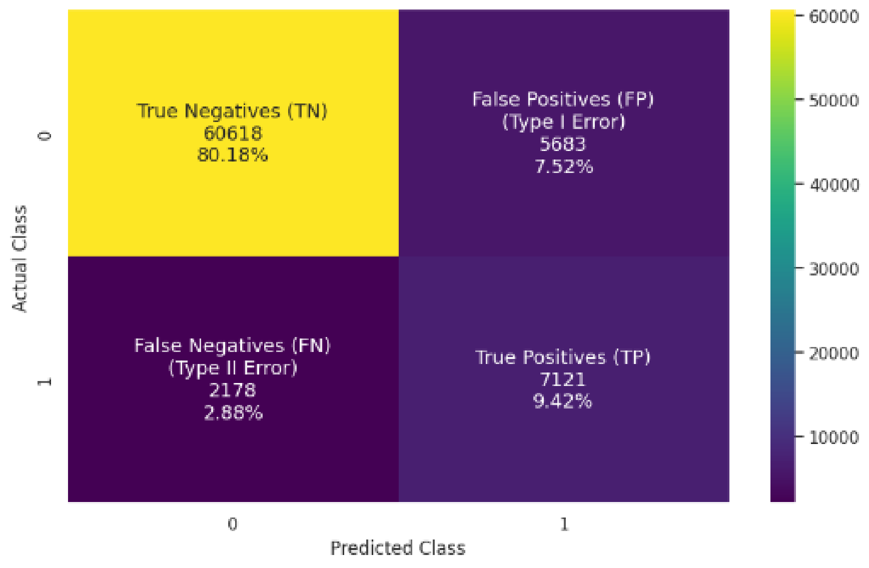
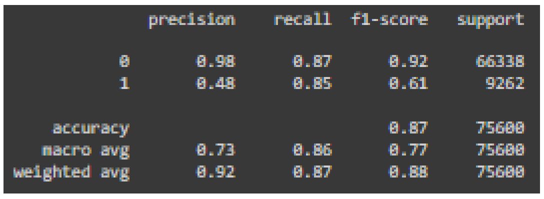


Figure 14: Confusion matrix model



```
              precision    recall  f1-score   support

           0       0.98      0.87      0.92     66338
           1       0.48      0.85      0.61      9262

    accuracy                           0.87     75600
   macro avg       0.73      0.86      0.77     75600
weighted avg       0.92      0.87      0.88     75600
```

4.4. Decision Tree

Figure 15: Decision Tree test results

The evaluation results of the Decision Tree model show very good performance for class 0, namely individuals who do not default. Precision for class 0 reaches 0.98, which means that 98% of the predictions generated for this class are correct. Recall of 0.87 indicates that the model successfully detected 87% of individuals who did not actually default. The F1-score for this class is 0.92, reflecting a good balance between precision and recall, as well as the effectiveness of the model in classifying individuals who are not at risk.

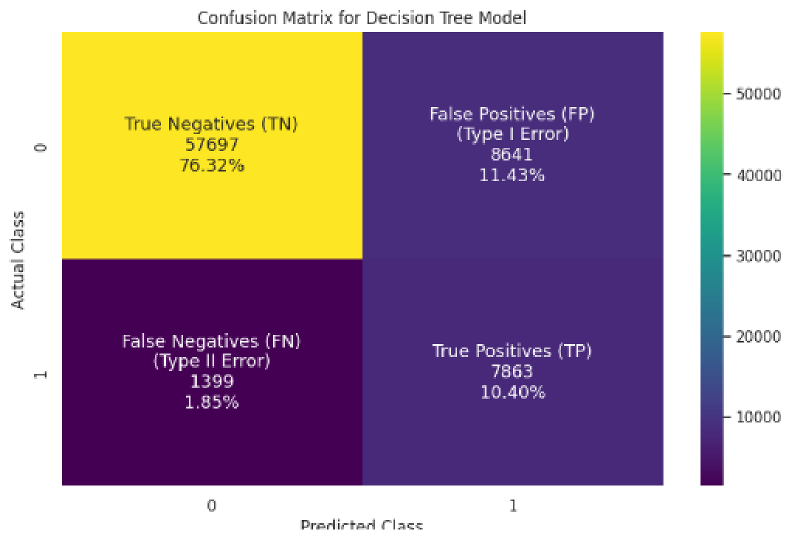
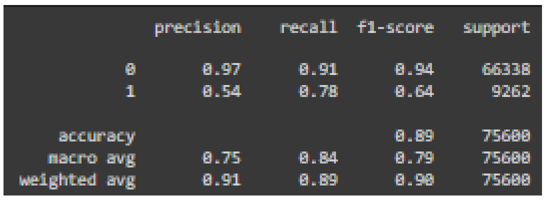


Figure 16: Confusion matrix model

However, the model performance on class 1, i.e. individuals at risk of default, despite the improvement in recall, still has low precision. Precision for this class is 0.48, meaning that only 48% of the predictions for at-risk individuals are correct. However, recall for class 1 is quite high at 0.85, indicating that the model is able to identify 85% of individuals who are truly at risk of default. The F1-score of 0.61 indicates that, despite the high recall, low precision limits the balance in detecting at-risk individuals.

Overall, the accuracy of the Decision Tree model is 87%, which means that the model is able to predict credit status quite well. The macro averages for precision, recall, and F1-score are 0.73, 0.86, and 0.77, respectively, indicating that the model has a fairly good performance, but the gap between the majority and minority classes still needs attention. The weighted average, which takes into account the class distribution, shows a precision of 0.92, recall of 0.87, and F1-score of 0.88. This confirms that the Decision Tree model performs well for the majority class, but still has limitations in predicting individuals at risk of default.


```
              precision    recall  f1-score   support

           0       0.97      0.91      0.94     66338
           1       0.54      0.78      0.64      9262

    accuracy                           0.89     75600
   macro avg       0.75      0.84      0.79     75600
weighted avg       0.91      0.89      0.90     75600
```

4.5. XGBoost

Figure 17 : XGBoost test results

The evaluation results of the XGBoost model show very good performance, especially in classifying class 0, namely individuals who do not default. The precision for this class reaches 0.97, which means that 97% of the predictions generated for individuals who do not default are correct. With a recall of 0.91, the model is able to detect 91% of all individuals who are truly not defaulted. The F1-score for this class reaches 0.94, which reflects a very good balance between precision and recall, indicating that this model is effective in predicting individuals who are not at risk.

In class 1, which is individuals at risk of default, the model precision is lower at 0.54. This means that only 54% of the predictions for at-risk individuals are correct. However, the recall for this class is quite high at 0.78, indicating that the model is able to detect 78% of all individuals who are truly at risk of default. The F1-score for this class is 0.64, indicating that despite the high recall, the lower precision slightly reduces the model's balance in classifying the at-risk class.

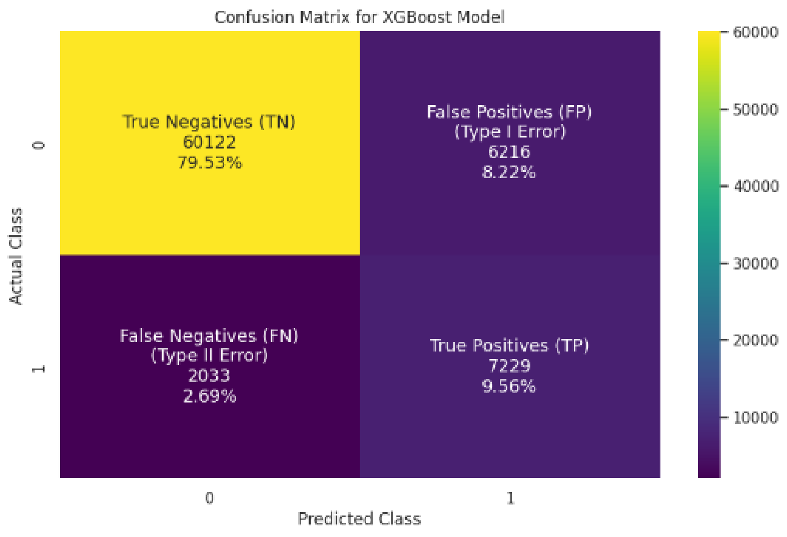
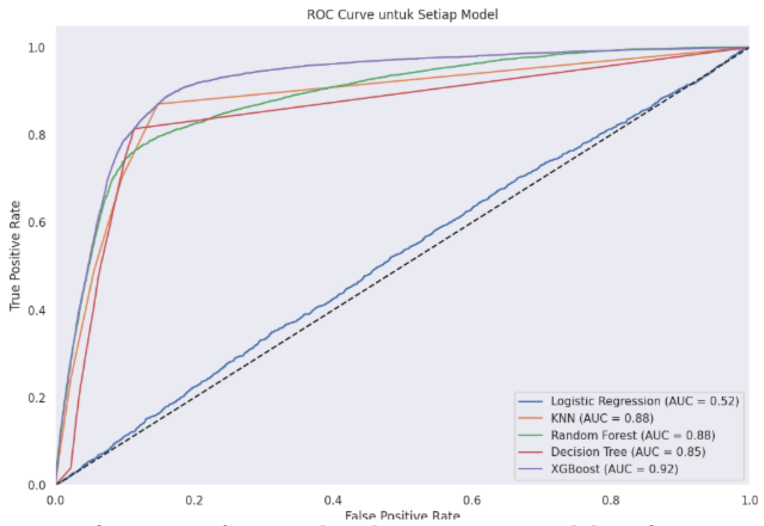


Figure 18: Confusion Matrix XGBoost

Overall, the XGBoost model has an accuracy of 89%, indicating a good performance in predicting credit status. The macro average values for precision, recall, and F1-score are 0.75, 0.84, and 0.79, respectively, indicating that the model is quite reliable in handling both classes, although there is a difference in performance between the majority and minority classes. The weighted average, which takes into account the class distribution, shows a precision of 0.91,



a recall of 0.89, and an F1-score of 0.90, confirming that the XGBoost model performs very well overall, especially in predicting non-defaulting individuals.

Figure 19: ROC curve of the model

4.6. ROC-AUC Score

In this study, ROC AUC is used to assess model performance because the data used is very imbalanced, where class 0 covers 87.7% of the dataset, while class 1 is only 12.3%. This imbalance makes accuracy alone insufficient to describe the overall quality of the model. A model may have high accuracy by only predicting the majority class, but fails to detect the minority class, which is the main focus in credit risk analysis.

ROC AUC was chosen because it measures the model's ability to distinguish between the majority and minority classes, regardless of the imbalance in the amount of data. This metric provides a more comprehensive view of how well the model can identify individuals at risk of default, even when the class is much smaller in number than the non-risk class.

From the results obtained, Logistic Regression only has an ROC AUC of 0.52, almost equivalent to random prediction, indicating that this model is less able to handle data imbalance. The KNN and Random Forest models each recorded an ROC AUC of 0.88, which shows quite good performance in separating the two classes. Decision Tree is slightly behind with a score of 0.85, but still shows decent ability. XGBoost recorded the highest ROC AUC with 0.92, indicating that this model is the most reliable in dealing with imbalance and predicting risk accurately.

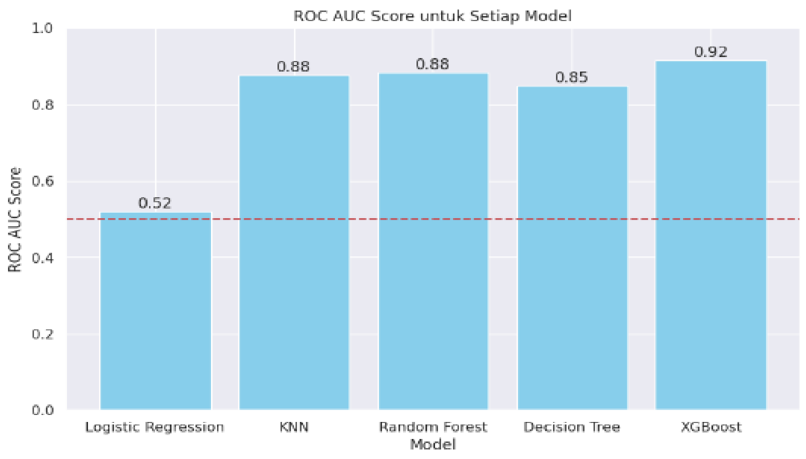


Figure 20: ROC-AUC score

Table 4. Comparison of accuracy with AUC score

Model	Accuracy	Percentage	AUC Score
Logistic regression	0.5328835978835978	53%	0.52
KNN	0.8600529100529101	86%	0.88
Random Forest	0.8960185185185185	90%	0.88
Decision Tree	0.8671957671957672	87%	0.85
XGBoost	0.8908862433862433	89%	0.92

Comparison of accuracy with AUC score

5. Conclusion

The conclusion of this study shows that model selection in credit risk analysis is crucial, especially when faced with data imbalance. Although Random Forest has the highest accuracy among all tested models with an accuracy value of 0.90 and an AUC of 0.88, XGBoost is considered the best model with an AUC of 0.92. This is due to its ability to handle data imbalance more effectively. Data imbalance is a challenge in credit risk modeling, where the defaulting borrower class only accounts for 12.3% of the total dataset. In this case, the Area Under the Curve (AUC) score becomes a more relevant metric than accuracy. XGBoost shows a better ability to distinguish between positive and negative classes, resulting in more accurate predictions related to credit risk with a recall of 0.78.

XGBoost is also better at preventing overfitting than other models. With better regularization techniques, XGBoost improves its performance on imbalanced datasets, providing more reliable results. In this case, the F1-score for XGBoost is 0.64, indicating a fairly good balance between precision and recall.

Overall, this study emphasizes the importance of considering data characteristics in model selection. XGBoost is proven to be superior in dealing with the challenge of data imbalance, making it a good choice in credit risk analysis. With a higher AUC, XGBoost shows its more consistent performance in predicting loan default risk.

Acknowledgements

Many thanks to Kang Panji who has acted as a supervisor and has taught this course during the process of making this paper. His guidance and support are very helpful in completing this research.

Gratitude is also expressed to Prof. Sukono as the lecturer in charge of the course who has provided direction and support during the lecture. His teaching greatly contributed to the writing of this paper.

References

Barongo, R.I., & Mbelwa, J.T. (2024). Using machine learning for detecting liquidity risk in banks. *Machine Learning with Applications, 15*, 100511.

Chen, Y., Calabrese, R., & Martin-Barragan, B. (2024). Interpretable machine learning for imbalanced credit scoring datasets. *European Journal of Operational Research, 312*(1), 357-372.

Kozina, A., Kuźmiński, Ł., Nadolny, M., Miałkowska, K., Tutak, P., Janus, J., Płotnicki, F., Walaszczyk, E., Rot, A., Dziembek, D., & Krol, R. (2023). The default of leasing contracts prediction using machine learning. *Procedia Computer Science, 225*, 424–433.

Li, X., Ergu, D., Zhang, D., Qiu, D., Cai, Y., & Ma, B. (2022). Prediction of loan default based on multi-model fusion. *Procedia Computer Science, 199*, 757–764.

Rodgers, W., Hudson, R., & Economou, F. (2023). Modeling credit and investment decisions based on AI algorithmic behavioral pathways. *Technological Forecasting and Social Change, 191*, 122471.

Setiawan, N., Suharjito, & Diana. (2019). A comparison of prediction methods for credit default on peer to peer lending using machine learning. *Procedia Computer Science, 157*, 38–45.

Wang, O., Zhang, Y., Lu, Y., & Yu, X. (2020). A comparative assessment of credit risk model based on machine learning: A case study of bank loan data. *Procedia Computer Science, 174*, 141–149.

Xianyu, Q., & Hai, M. (2023). Research on default prediction model of corporate credit risk based on big data analysis algorithm. *Procedia Computer Science*, 221, 300–307.

Zedda, S. (2024). Credit scoring: Does XGboost outperform logistic regression? A test on Italian SMEs. *Research in International Business and Finance, 70*(Part B), 102397.

Zhu, L., Qiu, D., Ergu, D., Ying, C., & Liu, K. (2019). A study on predicting loan default based on the random forest algorithm. *Procedia Computer Science, 162*, 503–513.