Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th February 2026. Vol. 5. No. 2; e-ISSN: 2808-4519

Comparison of Logistic Regression and XGBoost Model Performance in Predicting Credit Scores

Stacyana Jesika Surianto[1]*, Chairunisah[2]

[1,2]Universitas Negeri Medan
stacyanajs42@gmail.com[1], denisaziyad0105@gmail.com[2]*

Abstract

Credit Scoring is a mathematical approach used to assess the creditworthiness of individuals or companies by classifying debtors into certain categories based on their risk profiles. This study aims to compare the performance of the Logistic Regression and XGBoost machine learning algorithms in predicting credit scores (credit scoring) to reduce the risk of Non-Performing Loan (NPL) risk at PT Graha Mazindo Mandiri. The secondary dataset used contains 1,533 car loan debtor data with 17 variables, including 1dependent variable and 16 independent variables. The research process includes data preprocessing (cleaning, handling outliers, encoding, normalization, and class balancing with SMOTE), modeling, and evaluation using the Accuracy, Precision, Recall, F1-score, and ROC-AUC metrics. The results show that XGBoost excels with 96% accuracy and ROC-AUC of 0.99 compared to Logistic Regression with an accuracy of 88% and ROC-AUC0.94, due to XGBoost ability to capture non-linear patterns and handle data imbalance. This study provides insights into credit risk factors and supports more accurate credit decision-making, with recommendations for hyperparameter optimization and model integration into operational systems.

Keywords: *Credit Scoring, Logistic Regression, XGBoost, Machine Learning, Non-Performing Loan, SMOTE*

1. Introduction

In the financial sector, lending is an important aspect that enables individuals and businesses to obtain funds for various purposes, ranging from personal needs to investments [1]. However, every loan carries the risk of default, so financial institutions need to implement an appropriate credit assessment system to reduce this risk [2]. One method used in credit risk assessment is credit scoring. Credit scoring is a mathematical approach used to assess the creditworthiness of individuals or companies by classifying them into specific categories based on their risk profiles [3]. This scoring model is generally expressed in the form of a probability function that links certain variables to the likelihood of default.

Previous studies have explored various approaches to improve credit scoring accuracy. One of them, conducted by Ainul Yaqin, compared the performance of XGBoost and Logistic Regression using the Australian Credit Approval dataset from the UCI Machine Learning Repository. Using stratified k-fold cross-validation, XGBoost achieved an average accuracy of 85.51%, while Logistic Regression reached 85.94%. Logistic Regression outperformed XGBoost in accuracy, recall, and F1-score, while XGBoost showed better precision but tended to overfit more. Based on these findings, Logistic Regression was identified as the best algorithm for credit scoring in this study. However, both models were able to classify creditworthiness effectively and can assist credit analysts in decision-making [4].

Research by Delima proposed the use of Extreme Gradient Boosting (XGBoost) and Adaptive Boosting (AdaBoost) to identify loan risks and prevent payment defaults. Using the Kaggle dataset *Peer-to-Peer (P2P) Lending*, the study applied 34 borrower-related features in classification models. Key influential features included interest rate (*int_rate*), monthly installment (*installment*), and loan amount (*loan_amnt*). Based on ROC curve evaluation, XGBoost achieved superior performance with an AUC of 0.92 compared to AdaBoost's 0.89, indicating better classification capability [5].

Although many studies have applied algorithms in credit scoring, several research gaps remain. Key challenges include optimizing feature selection to improve prediction accuracy and addressing class imbalance, which often hinders traditional classification models. This study aims to explore a more effective credit scoring approach by comparing Logistic Regression and XGBoost, evaluating their ability to handle imbalanced data, and identifying the main factors influencing creditworthiness.

Mathematically, the approach that is often used is Logistic Regression, which represents the relationship between independent variables (borrower's financial features) and dependent variables (creditworthiness) [6]. On the other hand, XGBoost is a decision tree-based algorithm that utilizes gradient boosting, which works by building models iteratively and optimizing results to improve prediction accuracy by adjusting the error weight at each iteration [7]. The use of these two algorithms in credit scoring is crucial to minimize the risk of default and improve the efficiency of the financial system. Logistic Regression and XGBoost are closely related in the credit score prediction process because both are used to estimate the probability of default, but through different approaches. This relationship is evident in their shared primary objective, which is to measure credit risk, where Logistic Regression serves as an easily interpretable baseline, while XGBoost is a more sophisticated method for improving prediction accuracy and performance.

This study provides significant benefits for companies. First, it helps reduce losses due to an increase in Non-Performing Loans (NPL) and serves as a reference for financial institutions in assessing borrower eligibility, thereby making the credit evaluation process more efficient [4]. Second, this study enables the model to identify the most influential features that affect creditworthiness, supporting more optimal decision-making [8]. Finally, analyzing borrower patterns such as age trends or higher default rates in certain occupations provides data driven insights that improve service quality and help companies anticipate NPL risks.

2. Research Method

This research is quantitative in nature and falls under the category of applied research because it aims to solve practical problems, namely reducing non-performing loans (NPL) at credit financing companies. The data used is secondary data from credit financing companies, which is aggregated, meaning that it has been grouped or summarized to maintain customer confidentiality and privacy. This data source includes historical data on car loan customers, which will later be analyzed statistically and computationally.

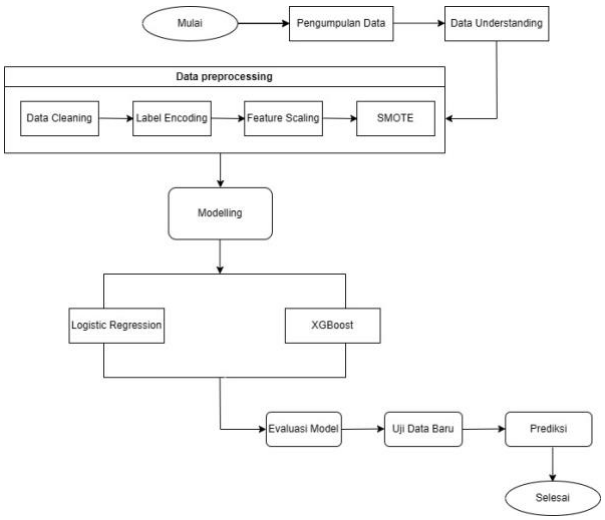

Fig. 1: Flowchart Research

The following is a discussion of the research stages from the image above
1. Data Collection: The data selected is secondary data from credit financing companies.
2. Data Understanding: This process is an exploration to understand the characteristics of the data.
3. Data pre-processing, which is the process of cleaning the data from noise or inconsistent values so that the data is ready to be used in the model.
4. Model creation, which is building a model using the Logistic Regression and XGBoost algorithms.
5. Model evaluation, which is measuring model performance using metrics such as Accuracy, Precision, Recall, F1 Score, and AUC-ROC.

3. Result and Discussion

3.1. Data Source

In this study, the data used is secondary data, namely data on car loan debtors at a financing company branch in Medan, North Sumatra. The data obtained consists of 1533 debtors with 17 variables (16 independent variables and 1 dependent variable). Some of the variables include: *Sk_Id_Curr* (code or ID of the creditor), *Name_Car* (type of car being financed), *Price_Car*, *Percent_Dp* (down payment percentage), *Rate_Interest* (interest rate borne by the debtor), *Amt_Credit* (amount of credit borrowed), *Amt_Annuity* (amount of monthly installments paid), *Income_Total* (total income of the debtor), *Cnt_Payment* (number of installment months), *Rasio_Dti* (debt-to-income ratio), *Data_Blacklist* (debtor's credit history list), *debtor* (type of debtor, whether old or new), *Age, Occupation_Type* (debtor's type of employment), *Cnt_Family* (number of dependents in the family), *Housing_Type* (debtor's home ownership status). and last the dependent variable is *Target* (whether the credit application is accepted or rejected).

3.2. Data Understanding

Overall, the analyzed dataset contained 1,533 debtors. The first stage involved initial data exploration to understand the basic characteristics of these variables. This exploration revealed the diversity of debtor profiles, ranging from occupation type and total income to car ownership.

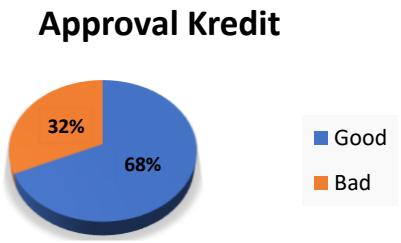

Fig. 2: Target distribution percentage

Figure 2 shows that the class distribution in the target variable indicates data imbalance, with a ratio of positive classes to negative classes of approximately 2:1. In this case, there are 1,047 debtors in the positive class and 486 debtors in the negative class. The next step is to perform matrix correlation, which is used to measure and compare the relationships between variables in the dataset. A positive value indicates a positive correlation (an increase in one variable is followed by an increase in another variable), while a negative value indicates a negative correlation (an increase in one variable is followed by a decrease in another variable). A coefficient of 0 indicates no correlation.

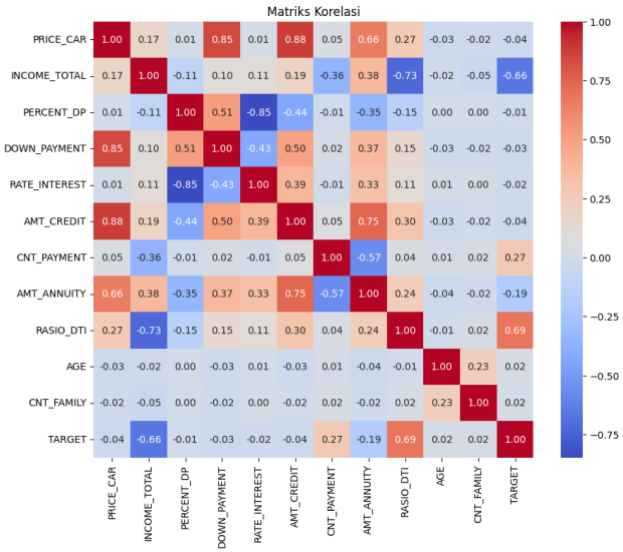

Fig. 3: Correlation Matrix

Figure 4.2 above shows a correlation matrix from which we can conclude the relationships between variables in the dataset, including:
1. The *amt_annuity* variable has a strong positive correlation with *Price_Car* (0.66) and *Amt_Credit* (0.75), indicating that the higher the car price and the larger the amount of credit taken, the higher the installment amount that must be paid.
2. The *Rate_Interest* variable shows a strong negative correlation with *Percent_Dp* (-0.85), indicating that the higher the down payment, the lower the interest rate that the debtor is likely to receive.
3. The *target* variable has a strong positive correlation with *rasio_dti* (0.69), where the debt-to-income ratio has a strong correlation with credit risk (target). When *Ratio_Dti* is high (meaning that debt is greater than income), the possibility of credit default (target = 1) also increases. Meanwhile, *Income_Total* (-0.66) has a strong negative correlation, which means that the greater the customer's total income, the lower the possibility of credit default risk (target = 1).
4. The relationship between *Income_Total* and *Rasio_Dti* has a strong negative correlation of (-0.73), which means that the higher a person's total income, the lower their DTI ratio, reflecting the customer's good financial health.

3.3. Preprocessing Data

3.3.1. Data Cleaning

Data cleaning is performed to ensure the quality of the dataset. To prevent missing values, data imputation is performed for numeric and categorical columns, using the median for numeric columns and the mode for categorical columns. During data cleaning, duplicate data is also removed.

3.3.2. Handling Outliers

The outlier handling process was carried out using IQR (Interquartile Range), resulting in 3% of data being considered outliers originating from the *amt_annuity* feature. *amt_credit*, and *rate_interest* data are considered outliers because the *amt_credit* value exceeds 685 million, the number of installments exceeds 18 million, and the interest rate exceeds 4%, which is considered to be outside the normal range. A total of 51 rows of data are included in the outliers. After deleting the outlier data, the amount of data before and after handling the outliers can be seen in Table 1.

Table 1: Comparation before and after removal outliers

Amount of data before outlier removal	1533
Amount of data after outlier removal	1482

3.3.3. Leakage Feature Removal

Leakage features are information that should not be available when the model makes predictions, but is accidentally included in the training features. This can cause the model to obtain information that should not be there, making the evaluation results appear very good, but causing poor performance when applied to new data. Some of the leakage features that were discarded were *rasio_dti* and *income_total*. These variables have a fairly strong correlation, where the relationship between *rasio_dti* (0.69) and *income_total* (-0.66) with the *Target* variable is such that including these features could cause information to leak into the model.

3.3.4. Data Transformation (Encoding)

The encoding process is the process of converting categorical data into numerical data. Categorical features such as employment status or car type are converted into numerical data using label encoding, a process that involves replacing categorical values with numbers. In this process, there are five categorical features that are encoded, namely *name_car, occupation_type, data_blacklist, debitur*, and *housing_type*, resulting in numerical values. The results displayed before and after data transformation (encoding) can be seen in Tables 2 and 3 below:

Table 2: Data before encoding

Name_Car	Occupation_Type	Data_Blacklist	Debitur	Housing_Type
Mazda 2	Group A	There Is	New	Own Property
Mazda 6	Group B	There isn't any	Old	Parents' Property

Table 3: Data after encoding

Name_Car	Occupation_Type	Data_Blacklist	Debitur	Housing_Type
1	0	0	0	1
3	1	1	1	0

3.3.5. Data Transformation (Encoding)

The next step is data splitting, which is the process of dividing the dataset into different subsets, such as dividing the data into training data and testing data. The dataset that has undergone preprocessing, with a total of 1482 entries after applying SMOTE, was divided using the random split methide with a ratio of 80:20. This approach was chosen to test the robustness and consistency of the Logistic Regression and XGBoost models performance against variation in the composition of the training and testing data. The stratify=y parameter was applied to maintain a balanced classs proportion, given the initial imbalance of dataset (2:1) ratio before SMOTE). The data split resulted in 1185 entries for the train set and 297 entries for the test set. Training and testing data splits can be seen table 4.

Table 4: Splitting data

Dataset	Total Data	Percentage
Data Training	1185	80%
Data Testing	297	20%
Total	1482	100%

3.3.6. Handling Data Imbalance

The dataset shows an imbalance with a ratio of positive (smooth) to negative (congested) classes of 2:1. The SMOTE (Synthetic Minority Oversampling Technique) technique is only applied to the train set, producing new synthetic data. This technique works by selecting one

minority sample, then finding k-nearest neighbors (k = 5) from the same class, and then creating a new sample on the connecting line between that sample and one of its closest neighbors. The formula used is

$$x_{new} = x_i + (x_{nn} - x_i) \times \delta \quad (1)$$

Where x_i is the original minority data, x_{nn} is the closest neighbor, and δ is a random number between 0-1. This process is repeated until the amount of minority data increases as needed, so that the total entries increase from 1185 to 1604. This approach was chosen to improve the representation of the minority class without removing the original data. A comparison of the amount of data before and after applying the SMOTE technique can be see in Figure 4.

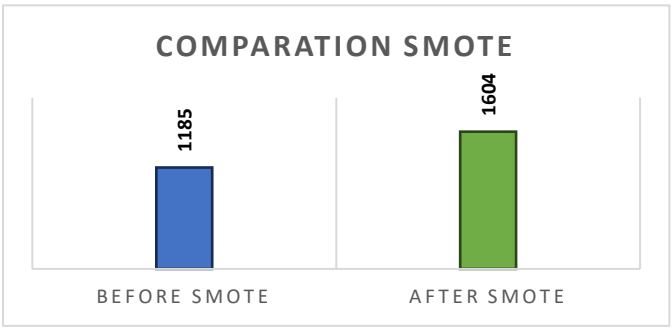

Fig. 4: Comparation Before and after SMOTE

3.3.7. Data Normalization

Data normalization is a process used to convert the value range of a dataset to a more standard range using *StandardScaler*. Normalization is very important when data has different scales, because many machine learning algorithms are sensitive to feature scales. Features such as PRICE_CAR (range of Rp200 million - Rp583 million) and CNT_PAYMENT (0 - 60) have different scales, so standardization is performed using *StandardScaler* to convert the data to a standard distribution (mean = 0, variance = 1). This process is applied before modeling to ensure that the contribution of each feature is balanced. A comparison of the results before and after normalization can be seen in Tables 5 and 6 below:

Table 5: Data before normalization

Price_Car	Cnt_Payment	Rate_Interest
371100000	60	0.033
403300000	36	0.037
583800000	48	0.031

Table 6: Data after normalization

Price_Car	Cnt_Payment	Rate_Interest
- 1.510132	0.775508	1.03454
-1.364824	-1.459335	1.87126
-0.550291	-0.341914	0.61619

3.4. Logistic Regression Modeling

Logistic Regression was chosen for its simplicity and effectiveness in binary classification, particularly with standardized numerical features. The model was implemented using the Scikit-learn library with default parameters as a baseline and trained on a balanced training set generated through SMOTE. Regularization parameters are used to help prevent overfitting in models. With C = 0.001, the model will be very regular, which may cause the model to be simpler and more generalized. By setting class_weight to 'balanced', the algorithm automatically adjusts the weight for each class based on its frequency of occurrence in the data.

Modeling was conducted using random split to test the consistency of Logistic Regression performance across different training and testing compositions. Additionally, 5-fold cross-validation was applied to the training set, dividing it into five folds where four were used for training and one for validation in each iteration. This technique is crucial for handling imbalanced class distributions, ensuring each fold maintains the original class composition and providing more representative and unbiased evaluation. The evaluation metrics included Accuracy, ROC-AUC, and Recall, with Recall emphasized to reduce false negatives and ensure positive cases are properly detected.

The cross-validation results were averaged to obtain the mean and standard deviation for each metric. The mean reflects the model's overall performance, while the standard deviation indicates consistency across folds. Model stability was further assessed using the coefficient of variation (CV), defined as the ratio of standard deviation to mean. A small CV suggests consistent performance and good stability, whereas a large CV indicates instability and varying results across folds. Based on the 5-Fold Stratified Cross-Validation, the model performance was obtained as follows:

Table 7: Evaluation report 5 fold cross validation

Metric	Mean	Std Dev	CV
Accuracy	0.9040	0.0226	2.50%
ROC-AUC	0.9652	0.0132	1.37%
Recall	0.8641	0.0238	2.75%

This can be explained from Table 7. Overall, all three metrics have low CV (<5%), indicating that the model is not sensitive to variations in the training data composition. AUC is the most stable metric, followed by accuracy, while recall has slightly higher variability but remains good. Finally, the evaluation was conducted on the test set (297 entries) using the metrics of accuracy, F1-score, precision, recall, and AUC-ROC to measure the model's ability to capture minority classes (traffic jam risk). The evaluation report results are shown in Table 8 as follows:

Table 8: Classification Report Model Logistic Regression

	Precision	***Recall***	***F1-Score***	***Support***
0 (Good)	0.93	0.89	0.91	201
1 (Bad)	0.79	0.86	0.83	96
Accuracy			0.88	297
Macro avg	0.86	0.88	0.87	297
Weighted avg	0.89	0.88	0.88	297
ROC-AUC Score	*0.94*			

3.5. Extreme Gradient Boosting (XGBoost) Modeling

XGBoost was selected for its ability to handle imbalanced data, deliver high accuracy, support regularization to prevent overfitting, and provide adjustable parameters to reduce bias toward the majority class. To enhance model robustness, 5-fold cross-validation was applied to the training set. Evaluation metrics included Accuracy, ROC-AUC, and Recall, with Recall emphasized to minimize false negatives and ensure positive cases are properly detected. The XGBoost model employed several key parameters. *n_estimators* determines the number of trees, with more trees generally capturing data patterns better. *Learning_rate* (0.1) slows learning but improves performance by allowing each tree to correct previous errors. *Max_depth* sets the maximum depth of decision trees, with a depth of 4 ensuring simplicity and generalization. *Subsample* specifies the proportion of training data used for each tree, helping prevent overfitting by introducing variation. *Colsample_bytree* defines the proportion of features randomly selected for each tree. *Eval_metric* was set to "logloss" to measure how well predicted probabilities fit the true labels.

Cross-validation results produce mean and standard deviation values for each metric, where the mean indicates the average performance of the model and the standard deviation indicates the consistency of its performance across each fold. Model stability is assessed through the coefficient of variation (CV), which is the ratio of standard deviation to mean. A small CV indicates consistency and stability of the model, while a large CV indicates instability and variability in performance across folds.

Table 9: Evaluation Report 5 fold cross validation

Metric	Mean	Std Dev	CV
Accuracy	0.9776	0.0067	0.68%
ROC-AUC	0.9971	0.0021	0.21%
Recall	0.9938	0.0040	0.40%

The results of the 5-fold evaluation show that all three metrics have a CV below 2%, which means that the model has very high stability and consistent performance across various data subsets. AUC is the most stable metric (CV 0.21%). Accuracy and Recall are also very stable, with near-perfect performance. Overall, these results show that the model is not only accurate, but also very robust to data variations in the cross-validation process. The evaluation was conducted on the test set (297 entries) using the Accuracy, F1-score, precision, recall, and AUC-ROC metrics to measure the model's ability to capture minority classes (traffic jam risk). The evaluation report results are shown in Table 10 as follows:

Table 10 : Clasification Report Model XGBoost

	Precision	***Recall***	***F1-Score***	***Support***
0 (Good)	0.98	0.96	0.97	201
1 (Bad)	0.91	0.97	0.94	96
Accuracy			0.96	297
Macro avg	0.95	0.96	0.95	297
Weighted avg	0.96	0.96	0.96	297
ROC-AUC Score	0.99			

The next step is to analyze feature importance in the XGBoost algorithm, where feature importance is a metric used to measure the extent to which each feature (variable) contributes to the model's prediction. In the context of this study, feature importance is calculated based on the number of feature splits in the XGBoost decision tree (gain-based importance), which shows how much the feature improves accuracy. By reducing uncertainty (impurity) at each node of the tree. The use of feature importance is important because it provides insight into the variables that are most influential in predicting credit scoring. This can be seen in Figure 5 below:

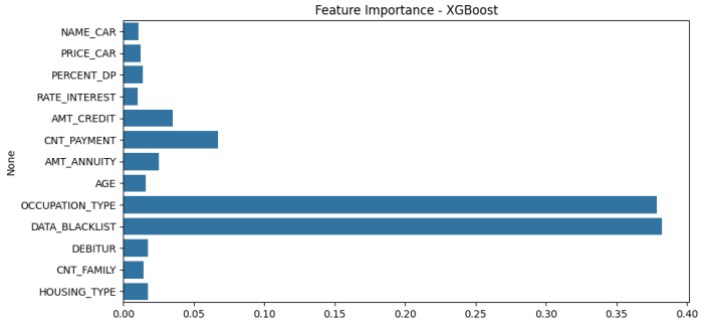

Fig. 5: Feature Importance Model XGBoost

Figure 5 present a bar chart illustrating feature importance in the XGBoost model. Each bar represent a feature, with its length indicating the constribution to prediction. The five most influential feature are:

1. *Data_Blacklist* : the strongest predictor, showing whether borrower is have prior financial blacklist records, a clear red flag for default risk.
2. *Occupation_Type* : borrowers job type, reflecting income stability and repayment ability, making it a key credit risk indicator.
3. *Cnt_Payment* : the number of payment or loan tenor, where longer terms increase the risk of financial changes during repayment.
4. *Amt_Credit* : the loan amount requested, with larger credit burdens raising default risk.
5. *Amt_Anuuity* : the borrower is monthly installment, where high anuuity relative to income signals heavier financial pressure and higher default probability.

3.6. Comparation of Model Evaluation Results

After the training and evaluation process on the car loan dataset was completed, a comparison stage was carried out to evaluate the performance of two machine learning models, namely Logistic Regression and XGBoost, in predicting debtor credit risk. This study utilized a dataset that had undergone preprocessing, including data cleaning, outlier handling, encoding, standardization, and class balancing through SMOTE on the train data, resulting in 1185 entries. This comparison aims to identify the most optimal model based on evaluation metrics, namely Accuracy, Recall, and ROC-AUC Score, which reflect overall prediction ability and the ability to discriminate between positive (default risk) and negative (non-default) classes. The analysis was performed on a test set consisting of 405 entries, taking into account the context of the initial data imbalance and the optimized hyperparameter tuning process.

Accuracy metrics are measured as the proportion of correct predictions to total predictions, providing an overview of the model's overall performance. However, given the initial class imbalance (2:1 ratio), this metric is supplemented with the ROC-AUC Score, which measures the area under the Receiver Operating Characteristic (ROC) curve. ROC-AUC evaluates the model's ability to distinguish between positive and negative classes by varying the threshold, making it more robust against data imbalance. The Recall metric is used to measure the model's effectiveness in comprehensively identifying the minority class (traffic jam risk), ensuring that as many actual positive cases as possible are detected, where Recall plays a crucial role in minimizing false negatives. The selection of this combination of metrics is based on the need to optimize the detection of traffic jam risk, which is a minority class.

Evaluation on the test set shows that XGBoost achieves an accuracy of 96%, recall of 96%, and a ROC-AUC score of 0.99, while Logistic Regression achieves an accuracy of 88%, recall of 88%, and a ROC-AUC of 0.94. These differences reflect the performance of the models on the data summarized in Table 11.

Table 11: Comparation Evaluation Report

Model	Accuracy	ROC-AUC Score	Recall
Logistic Regression	88%	0,94	88%
XGBoost	96%	0,99	96%

The superior performance of XGBoost (Recall 96%, ROC-AUC 0.99) compared to Logistic Regression (Recall 88%, ROC-AUC 0.94) can be explained by XGBoost's ability to capture non-linear patterns and complex interactions between features. Logistic Regression is limited by its assumption of linearity, as evidenced by its lower ROC-AUC, although it remains competitive as a baseline. These results indicate that XGBoost is more suitable for credit risk prediction applications on this dataset, especially in the context of feature imbalance and complexity. The implementation of XGBoost can support decision-making in financial institutions that provide credit services to identify high-risk debtors more accurately. However, Logistic Regression remains useful for preliminary analysis or scenarios where interpretability is required.

4. Conclusion

Based on the research conducted, the conclusions from the comparison of the performance of the Logistic Regression and XGBoost models in predicting credit scores are as follows

1. Both Logistic Regression and XGBoost were successfully applied to predict credit scores using secondary data from car loan customer data. The process involved debtor data collection, preprocessing with Label Encoding, normalization using StandardScaler, and class balancing with SMOTE (ratio 2:1). Logistic Regression was implemented with regularization (C=0.001) and balanced class weights, while XGBoost used parameters such as n_estimators=100, learning_rate=0.1, and max_depth=4. The models were trained on 1,617 entries and tested on 405 entries, producing predictions that support credit decision-making.
2. The analysis showed that XGBoost achieved 96% accuracy, 96% recall, and a ROC-AUC score of 0.99, outperforming Logistic Regression with 88% accuracy, 88% recall, and a ROC-AUC of 0.94. Key features influencing creditworthiness included *Data_Blacklist, Occupation_Type, Cnt_Payment, Amt_Credit*, and *Amt_Annuity*. SMOTE improved minority class representation, making the models more sensitive to high-risk borrowers. XGBoost proved effective in detecting credit risk, helping reduce Non-Performing Loans (NPL).
3. Method selection was based on recall and ROC-AUC metrics. ROC-AUC is robust for imbalanced datasets, while recall ensures detection of potential defaults. XGBoost demonstrated superior performance (ROC-AUC 0.99, recall 0.96) compared to Logistic Regression (ROC-AUC 0.94, recall 0.88). Its strength lies in capturing non-linear patterns and complex feature interactions, as well as handling imbalanced data after SMOTE. Although Logistic Regression is simple and interpretable, its linear assumptions limit performance. Therefore, XGBoost is recommended for credit scoring applications to support more accurate and efficient credit decisions.

References

[1] A. Waluyo, A. Mukid, and T. Wuryandari, “Perbandingan Analisis Klasifikasi Nasabah Kredit Menggunakan Regresi Logistik Biner Dan Cart (Classification And Regression Trees),” Vol. 4, Pp. 215–225, 2020.

[2] Fatimah, A. Mukid, And A. Rusgiyono, “Analisis Credit Scoring Menggunakan Metode Bagging K-Nearest Neighbor,” vol. 6, no. 1996, pp. 161–170, 2020.

[3] A. R. Hakim, M. A. Mukid, H. Yasin, and S. Sugito, “Analisis Klasifikasi Credit Scoring Menggunakan Weighted Probabilistic Neural Network (Wpnn),” *J. Stat. Univ. Muhammadiyah Semarang*, vol. 7, no. 1, 2020, [Online]. Available: https://jurnal.unimus.ac.id/index.php/statistik/article/view/4807

[4] A. Yaqin, “Penilaian Kredit Menggunakan Algoritma XGBoost dan Logistic Regression,” *J. Inform. J. Pengemb. IT*, vol. 8, no. 1, pp. 4–10, 2022, doi: 10.30591/jpit.v8i1.4337.

[5] R. Delima, M. Hosianna, D. Pebrianty, and J. Amalia, “Credit Risk Analysis dengan Algoritma Extreme Gradient Boosting dan Adaptive Boosting,” *J. Inf. Syst. Graph. Hosp. Technol.*, vol. 05, pp. 1–7, 2023.

[6] R. M. Daffaa, D. Santika, and F. Mahardika, “Perbandingan Xgboost dan Logistic Regression dalam Memprediksi Credit Card Customer Churn,” *Publ. Ilmu Keteknikan Ind. Tek. Elektro dan Inform.*, vol. 3, 2025.

[7] S. E. Herni Yulianti, S. Oni, and S. Yuana, “Penerapan Metode Extreme Gradient Boosting (XGBOOST) pada Klasifikasi Nasabah Kartu Kredit,” *J. Math. Theory Appl.*, vol. 4, no. 1, pp. 21–26, 2022, doi: 10.31605/jomta.v4i1.1792.

[8] J. Xiao, Y. Wang, J. Chen, L. Xie, and J. Huang, “Impact of resampling methods and classification models on the imbalanced credit scoring problems,” *Inf. Sci. (Ny).*, vol. 569, pp. 508–526, 2021, doi: 10.1016/j.ins.2021.05.029.