



Enhanced Loan Prediction Performance Using Blending Model Approach

Nur Haryadi¹, Siti Juwariyah², Muhammad Ricky Perdana Putra^{3*}

¹PT Geo Dipa Energi (Persero), Dieng, Indonesia

²Yayasan Cendekia Harapan, Jombang, Indonesia

³Independent Researcher, Tulungagung, Indonesia

*muhammad.ricky@students.amikom.ac.id

Abstract

The loan approval process at financial institutions is carried out manually in a conventional manner by looking at the customer's track record. This process is ineffective because it takes a considerable amount of time. As a result, a machine learning (ML) model has been developed that can recognize certain patterns in datasets for automatic and rapid prediction. However, the problem is that a single ML model is still not optimal, so it needs to be improvised, one of which is with Ensemble Learning (EL), which combines more than one model. This study uses Blending EL (BEL), which is built on two layers: the first layer as a base model with KNN, DT, and NB algorithms, and the second layer as a meta layer built with XGBoost. Pre-processing uses MinMaxScaler for data normalization and SMOTE-ENN for data class balancing. Testing uses a confusion matrix covering accuracy, recall, precision, and F1-Score, as well as Area Under Curve (AUC) and execution time, then combined with K-Fold cross validation with $k=10$. The BEL model prediction results were 96.42% accuracy, 96.29% precision, 96.29% recall, 96.29% F1-Score, and 98.72% AUC. Meanwhile, the average matrix in K-Fold cross validation is accuracy 96.84%, precision 98.09%, recall 95.57%, F1-Score 96.70%, and AUC 99.32% as well as execution time 0.261 seconds.

Keywords: blending ensemble learning; loan approval; machine learning; MinMaxScaler; SMOTE-ENN

How to Cite: Nur Haryadi, Siti Juwariyah, and Muhammad Ricky Perdana Putra, "Enhanced Loan Prediction Performance Using Blending Model Approach", *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 10, no. 3, pp. 657 - 668, Jun. 2026.

Permalink/DOI: <https://doi.org/10.29207/resti.v10i3.7242>

Received: October 3, 2025

Accepted: May 23, 2026

Available Online: June 14, 2026

This is an open-access article under the [CC BY 4.0 License](#)

Published by [Ikatan Ahli Informatika Indonesia](#)

1. Introduction

Rapid technological developments are driving significant changes in various sectors, including finance [1], [2]. Technology is used in data-based decision making involving big data to produce more precise, effective and impartial decisions [3], [4]. In the banking context, one crucial issue that requires a data-driven approach is loan approval prediction [5], [6].

This prediction aims to assess whether a prospective borrower has the ability to repay their credit obligations or is potentially at risk of default [7], [8]. The level of accuracy in loan predictions affects cash flow turnover, the profitability of financial institutions, and the stability of the financial system as a whole, making research on this topic highly urgent [9], [10].

Traditionally, financial institutions rely on manual assessments by credit analysts, taking into account personal data such as employment status, the prospective borrower's financial history, and even demographic factors [11], [12]. However, this method

has limitations, such as a high degree of subjectivity, time consumption, and an inability to process large amounts of complex data efficiently [13], [14].

As a solution to these limitations, Machine Learning (ML) methods were designed to handle large and complex datasets efficiently, identify unseen patterns, and improve prediction performance. [15], [16]. Thus, ML offers a superior alternative to conventional methods, which are often limited in dealing with the complexity of modern data [17].

Previous research has shown that ML and Deep Learning (DL) are effective in improving loan eligibility prediction systems. Emphasis on security and transparency by integrating Differential Privacy (DP) and Explainable AI (XAI) [18]. The study tested two DP mechanisms, namely Laplacian and Gaussian, combined with a data balancing technique called Synthetic Minority Oversampling Technique (SMOTE).

It was then tested with several algorithms, including Random Forest (RF), Extreme Gradient Boosting (XGBoost), Adaptive Boost (AdaBoost), Logistic Regression (LR), and Category Boost (CatBoost). This research found that CatBoost with a Gaussian mechanism produced DP at a privacy budget of 1.5, achieving the highest accuracy of 81.25%, which means good performance for prediction without sacrificing personal data security and privacy.

Unlike research [19] which highlights the importance of comprehensive evaluation through several metrics, encompassing accuracy, recall, precision, F1-Score, and Area Under the Receiver Operating Characteristic Curve (ROC-AUC). By incorporating widely recognized methods such as XGBoost, Gradient Boosting Model (GBM), RF, and LightGBM.

The study showed that the GBM model produced an accuracy of 88.87%, an F1-Score of 80.84%, and a recall of 80.21%. So, this model can identify borrowers who default on their payments. Meanwhile, XGBoost obtained an AUC value of 97.14%, indicating its discriminatory ability to distinguish between borrowers who will pay (true positive) and those who will default (false negative).

Another study by [20] exploring combinations of DL and ML with data imbalance handling using two techniques, SMOTE and SMOTE-Tomek. This paper compares the performance of various ML models such as LR, RF, GBM, Decision Tree (DT), K-Nearest Neighbor (KNN), Gaussian Naive Bayes (GNB) and AdaBoost with DL architectures such as Transformer, Autoencoder, Res-Net, Dense-Net, and Multi-Layer Perceptron (MLP).

Other architectures proposed include Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU). This study found that Dense-Net and Res-Net are the most accurate architectures. The combination of SMOTE and Dense-Net resulted in the highest accuracy (92%), precision (92%), recall (92%), F1-Score (92%), AUC (95%), and Matthews Correlation Coefficient (58%).

In line with that, [21] confirms the effectiveness of ML ensembles and SMOTE oversampling in improving loan prediction performance. This research shows that the XGBoost algorithm achieves accuracy, recall, precision, and F1-Score levels above 95% on balanced data. As for [22] proposing an ensemble voting that combines several of the best algorithms to produce an accuracy of up to 87.26%.

Although these studies show promising results, there are still a number of limitations that leave room for improvement. Most studies still rely on traditional ensemble methods, such as bagging, boosting, or voting [23] which, although effective, have limitations in utilizing the relative advantages of various models more flexibly.

In addition, commonly used data balancing techniques such as SMOTE or SMOTE-Tomek, both of which focus more on oversampling or a combination of oversampling and simple under sampling [24], [25]. However, it is still less than optimal in addressing data noise issues that can reduce model performance, and the potential for establishing more adaptive ensemble methods with smarter balancing techniques has not been widely researched.

The use of more modern ensembles, namely stacking and blending, can be considered for this topic. One study on Stacking Ensemble Learning (SEL) was conducted by [26] with the help of SMOTE to balance the data, LightGBM is used to select relevant feature, and base learners such as KNN, RF, and Support Vector Machine (SVM) are used, with XGBoost as the meta learner. The model is termed LGBFS-StackingXGBoost.

Tested with two different datasets, the results showed significant improvement compared to the base learner. With the P2P Online Lending dataset. This model achieved 99.98% accuracy, while the Lending Club Loan Data Analysis achieved 91.43%. This means that even though the models were generated using the same technique, the results varied due to changes in the dataset used.

On the other hand, there is research that has used Blending Ensemble Learning (BEL) conducted by [27] which built a fusion model with LR, RF, and CatBoost in the first layer and Gradient Boosting Decision Tree (GBDT) in the second layer. It utilized Adaptive Synthetic Sampling (ADASYN) to balance the data classes. This research produced a fusion model with an accuracy of 99.4%, recall and precision of 99%.

Based on previous research, this study proposes the application of BEL method as a more adaptive approach than bagging, boosting, voting, and even stacking [17], [28]. Blending enables the simultaneous utilization of the relative strengths of various algorithms through combination at the prediction level, thereby improving model generalization [29].

In addition, this research applies the SMOTE-ENN technique, which is a combination of SMOTE and Edited Nearest Neighbor (ENN) that serves to eliminate data noise and improve class distribution. With this combination of techniques, it is hoped that the loan eligibility prediction model will not only have high performance, but will also be more balanced, stable, and reliable in supporting decision making in the banking sector.

The contribution of this study is not in proposing a completely novel algorithm, but in systematically integrating data balancing (SMOTE-ENN) with BEL to address both data imbalance and model generalization simultaneously. This dual-focus approach provides a more stable predictive framework compared to studies that treat these aspects independently.

2. Methods

BEL is an ensemble technique with an approach of building a prediction model through two layers, namely the base learner and meta learner [30]. The research process begins with data collection, pre-processing,

BEL model development, and model testing, followed by drawing conclusions based on the test results to determine whether there is an improvement in performance compared to previous research. The visualization of the research flow is shown in Figure 1.

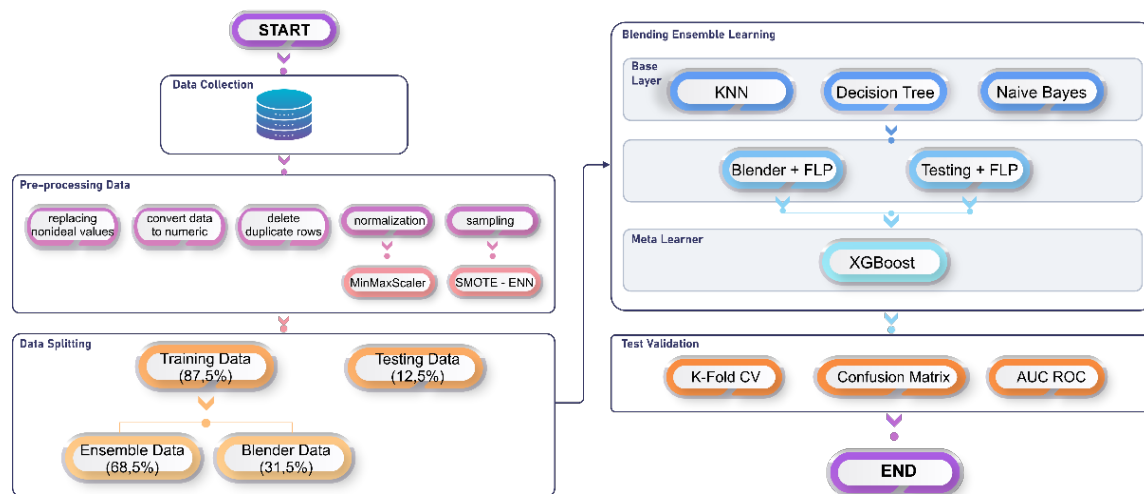


Figure 1. BEL Flowchart

2.1 Dataset

The dataset was uploaded by contributor Debdatta Chatterjee to Kaggle.com. This repository is still active, both in terms of code and discussion forums (<https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset/data>). The dataset is split into two parts: training and testing. Figure 2 shows a visualization of the distribution of training data classes.

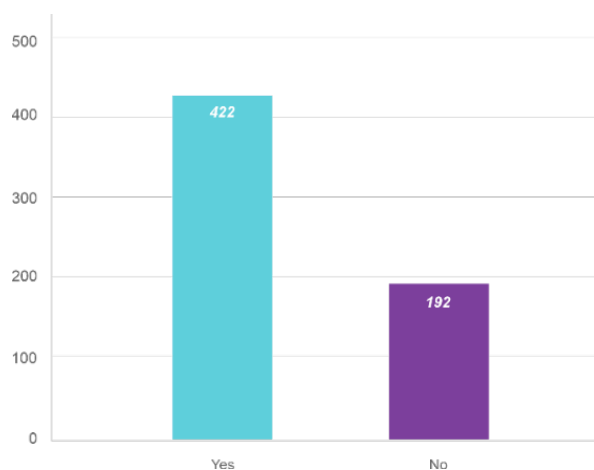


Figure 2. Training Data Class Distribution

The dataset has 13 features, Variable X as training data uses all features except loan ID and loan status. Then, variable Y as the label uses loan status, which has values of one and zero. To see the average feature values, see Table 1.

There are two reasons for choosing datasets from Kaggle: they are publicly available and free, which speeds up the research timeline. In addition, previous research has also used publicly available datasets from Kaggle and other contributors or other free dataset

repository sites. If using original datasets from banks or cooperatives, permission is required. This takes a lot of time and is less effective.

Table 1. Feature and Mean Value

Feature	Mean Value
Loan ID	0.000000
Gender	0.021173
Married	0.004886
Dependents	0.024430
Education	0.000000
Self-Employed	0.052117
Applicant Income	0.000000
CO Applicant Income	0.000000
Loan Amount	0.035831
Loan Amount Term	0.022801
Credit History	0.081433
Property Area	0.000000
Loan Status	0.000000

Although the dataset is publicly available and widely used, this study does not aim to introduce a new dataset but rather to evaluate methodological improvements in ensemble architecture under controlled and reproducible conditions. Using benchmark datasets ensures comparability with prior studies and allows the proposed method to be validated against established baselines.

Therefore, this study aims to evaluate methodological improvements in ensemble learning architectures for loan prediction tasks using publicly available benchmark datasets. Rather than focusing on deployment in a specific financial institution, the research emphasizes addressing limitations identified in previous studies, particularly in data preprocessing strategies and ensemble model design. Specifically, the proposed approach integrates data normalization and hybrid class balancing techniques with a blending

ensemble framework to enhance predictive stability and generalization performance.

2.2 Pre-processing

The pre-processing stage includes (1) replacing non-ideal values such as “3+”, (2) converting all data into numbers so that they are equivalent and can be trained into a model, for example, male gender becomes ‘1’ and female becomes “0”, (3) removing rows of data that have empty values, (4) performing normalization with MinMaxScaler, (5) balancing data classes with the SMOTE-ENN hybrid technique.

Dataset quality can be improved through pre-processing so that the model built is more optimal [31], data processes such as sorting, filtering, and applying formulas, preventing bias, and reducing data storage load [32]. Furthermore, MinMaxScaler helps to scale the attribute values in the data to the same range so that all attributes become uniform and there is no attribute dominance [33] with Equation 1 [34].

$$x' = \frac{x - \min(A)}{\max(A) - \min(A)} \quad (1)$$

SMOTE-ENN applies the K-Nearest Neighbor (KNN) concept by (1) selecting random data from the minority class, (2) finding the closest neighbors of that data, using a value of $k=5$ in this study, (3) selecting one neighbor at random, (4) creating synthetic data by taking the difference between the example and the neighbor, multiplying it by a random number between zero and one, and adding it to the original data [35].

In terms of the encoding algorithm, SMOTE-ENN begins by randomly selecting data from the minority class, then finding its nearest neighbors using KNN. Next, new synthetic data is generated based on the relationship between that data and its nearest neighbors. This process is repeated until the number of data points in the minority class reaches the desired balance ratio. Once balanced, the ENN method is used to clean the dataset by examining each data point and finding its three nearest neighbors. If a data point is classified differently by the majority of its nearest neighbors, it is considered noise and removed. The result of this process is a more balanced dataset with a lower noise level.

Previous research shows that SMOTE-ENN significantly improves prediction performance on imbalanced datasets, especially recall for minority classes while maintaining precision. Similar findings regarding the effectiveness of SMOTE-based resampling techniques in improving minority-class prediction performance have also been reported in other domains, such as disease prediction using machine learning models combined with PCA and SMOTE [35], both of which are important metrics in this research, as explained in more detail in the Testing section. Although there are cases where accuracy drops dramatically after SMOTEEN, recall can increase by

20-80% [36] because it is no longer biased towards the dominant class.

Before applying the SMOTE-ENN technique, the dataset was first divided into training and testing sets. The SMOTE-ENN resampling process was applied only to the training set, while the testing set remained unchanged. This procedure was implemented to prevent data leakage and to ensure that model evaluation reflects real generalization performance.

Furthermore, the MinMaxScaler was fitted exclusively on the training data to prevent information leakage, and the learned scaling parameters were subsequently applied to the validation and testing sets. The SMOTE-ENN resampling process was performed only after scaling and strictly on the training data. During K-Fold cross validation, scaling and resampling procedures were independently applied within each fold to ensure that no information from the validation subset was introduced into the training process.

2.3 BEL

BEL uses two layers consisting of base learners and a meta learner. The base layer employs multiple weak learners to increase model diversity and improve generalization performance through ensemble learning [37]. Blending Ensemble Learning (BEL) was selected instead of stacking due to its simpler training mechanism and lower computational cost. Unlike stacking, which requires k -fold cross validation to generate meta-features, blending uses a holdout validation set to train the meta learner. This approach reduces training complexity and helps minimize the risk of information leakage from cross-validation predictions. Considering the relatively small dataset used in this study, blending provides a more efficient and stable ensemble learning framework.

It is recommended that the number of algorithms used be odd to optimize the voting process in the meta learner so that there will definitely be a decision and avoid the possibility of a tie [38]. This research uses three classic algorithms: KNN, DT, and Naive Bayes (NB). The strength of the KNN-DT-NB combination lies in its heterogeneous inductive bias. KNN is effective for local and non-parametric patterns, DT handles non-linearity and feature interactions, and Naive Bayes excels at probabilistic data with independence assumptions, thus creating a strong ensemble for prediction [23].

When combined, these models generate uncorrelated errors, which is a critical requirement for effective ensemble learning. This diversity allows the meta-learner to learn non-linear relationships between model outputs rather than relying on redundant predictions. Previous research using this combination was conducted by [30], [39], resulting in improved performance compared to a single algorithm.

Three models are the minimum number required to build a BEL model because they satisfy the voting rule

and do not produce ties. The three must also use different algorithms for essential diversity because using the same algorithm would produce identical results and only add computational complexity without benefit [40], [41]. So, is it possible to use a single algorithm? The answer is no because that number cannot be used for voting.

KNN has a simple and easy-to-learn concept, but its weaknesses lie in (1) its inability to manage high-dimensional data, (2) time-consuming performance during testing, (3) susceptibility to data imbalance, and (4) determining the optimal k value. If k is too large, the model will be too general and less responsive to new data. Conversely, if k is too low, the model will be very susceptible to noise and outliers [42].

On the other hand, DT has the advantage of low computational costs due to its fast model training process, is easy to understand and learn because it is simple, and is able to handle missing data [43]. However, when DT becomes too complex, the risk of overfitting increases, thereby reducing generalization on new data and sensitivity to data imbalance [44]. This weakness is the basis for RF development.

Unlike DT, NB is a probabilistic classification algorithm known for its computational efficiency and stable performance, particularly on small datasets. NB assumes that all features are conditionally independent given the class label, which simplifies computation. However, this assumption often introduces high bias, causing the model to underfit complex relationships between features. In terms of performance, NB can be reliable for datasets with simpler feature structures, but it does not explicitly model relationships between features due to its independence assumption [45].

The three weak learner models were built from ensemble data and tested using blender data. The results are called data residuals, which have three columns of prediction results from the three models. The data was used to build a model in the second layer with the XGBoost algorithm. Simply put, XGBoost votes on what the base model produces, which is in line with what [38] recommends for using odd models.

To validate the role of XGBoost as a meta learner, feature importance analysis shows that it assigns different weights to each base learner prediction, indicating that it captures non-linear relationships rather than relying on simple majority voting [46]. As an extension GBDT, XGBoost iteratively minimizes model loss while incorporating regularization to reduce overfitting [47]. Additionally, its parallelization capability enhances computational efficiency, making it both robust and scalable as a meta-learning model [48].

XGBoost has a tree pruning concept, which means pruning tree branches that do not contribute significantly to model predictions in order to improve performance and avoid overfitting [49]. This can be combined with algorithms that can detect features with strong contributions, such as feature importance from both Random Forest and XGBoost itself, or using Genetic Algorithms (GA).

To complete the explanation of the models used in this research, Table 2 presents the hyperparameters used per algorithm as follows. These hyperparameters are still inherent to the algorithm itself and do not use external parameters. These parameters can be adjusted and optimized using tuning techniques such GridSearchCV or Random Search.

Table 2. Hyperparameter

Algorithm	Component	Hyperparameter	Value	Description
KNN	Default	K	5	Default value for neighbors
		Weights	Uniform	Weight based on distance
		Metric	Minkowski	Distance function
		P	2 (Euclidean)	The value of 2 in the Minkowski distance
	Imputer	Strategy	Median	Fill in missing or empty values
DT	Scaler	With Mean	False	Mean is not reduced
	Scaler	With Std	True	Set the variant to 1
	Default	Criterion	Gini	Impurity size (default Scikit-Learn library)
		Splitter	Best	Choosing the best split
		Max Depth	None	Maximum depth of tree
		Min samples split	2	Minimum sample for split
		Min samples leaf	1	Minimum sample in leaf
		Max Feature	None	Use all features
		Random State	42	Reproducibility
		Strategy	Most Frequency	Fill in missing or empty values
NB	Default	Type	Gaussian	Feature distribution assumptions
		Var Smoothing	1 ⁹	Variance stabilization
	Imputer	Strategy	Most Frequency	Fill in missing or empty values
XGBoost	Default	N Estimators	300	Number of boosting trees
		Max Dept	3	Maximum depth of tree
		Learning Rate	0.05	Learning rate (shrinkage)
		Sub Sample	0.9	Training data ratio
		Col Sample Bytree	0.9	Filter ratio per tree
		Objective	Binary Logistic	Objective function of binary classification
		Evaluation Metrics	Logloss	Internal evaluation metrics
		Random State	42	Reproducibility
		N Jobs	-1	Uses all CPU cores

2.4 Testing

In this study, testing was conducted using three techniques, namely (1) confusion matrix, (2) K-Fold cross validation, and (3) area under curve (AUC). These three techniques have different purposes, advantages, and disadvantages. The confusion matrix shows that the model produces true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions. The confusion matrix displays five matrices, namely accuracy, recall, precision (sensitivity), specificity, and F1-Score.

In this loan approval case study, the precision and recall matrices are quite crucial. High precision means preventing loan approvals to unqualified customers, thereby reducing losses due to customer defaults [50], [51]. Meanwhile, recall plays a role in selecting eligible customers, thereby increasing loan volume and financial inclusion [52]. Low recall means that the model rejects many good customers.

Next, K-Fold cross validation is used to determine data generalization by dividing the dataset into multiple subgroups according to the value of k . This research uses a value of $k=10$, this means that the dataset will be divided into ten subsets, with nine subsets as training data and the rest as test data [53]. A value of $k=10$ means that the model testing is repeated 10 times with different subsets.

Each iteration calculates the confusion matrix, which includes the four matrices mentioned above, as well as the AUC, which is a testing tool to measure the model's ability to distinguish between positive and negative classes. The AUC value ranges from zero to one. The higher the value, the better the model is at separating positive and negative classes, which means that all positive predictions are correct and all negative predictions are incorrect [54].

To facilitate reading the results, they will be visualized in the form of a line diagram that displays the confusion matrix and AUC results for each iteration of K-Fold cross validation. Then, it will be interpreted with previous research to determine whether this model has improved performance or previous research. After that, conclusions will be drawn on the process, results, and discussion of the research conducted.

3. Results and Discussions

After performing pre-processing steps one to three, the next step is normalization. In addition to converting values into a smaller range so that the data gap is not too wide, this research uses KNN, which is sensitive to data distance. Normalization can avoid the influence of outliers, thereby improving the performance of the prediction model [55]. Examples of data samples before and after normalization are presented in Tables 3 and 4.

Table 3. Before Normalization

Gender	Married	Dependents	Education	Self Employed	Applicant Income	CO Applicant Income	Loan Amount	Loan Amount Term	Credit History	Property Area
1	0	0	1	0	5849	0	999	360	1	2
1	1	1	1	0	4583	1508	128	360	1	0
1	1	0	1	1	3000	0	66	360	1	2
1	1	0	0	0	2583	2358	120	360	1	2
1	0	0	1	0	6000	0	141	360	1	2
...	...	...	...	...	...	...	...	...	...	...

Table 4. After Normalization

Gender	Married	Dependents	Education	Self Employed	Applicant Income	CO Applicant Income	Loan Amount	Loan Amount Term	Credit History	Property Area
0.5	0	0.2	1.0	0.0	0.070489	0.000000	1.000000	0.352584	1.0	1.0
0.5	0.5	0.4	1.0	0.0	0.054830	0.036192	0.120202	0.352584	1.0	0.0
0.5	0.5	0.2	1.0	0.5	0.035250	0.000000	0.057576	0.352584	1.0	1.0
0.5	0.5	0.2	0.0	0.0	0.030093	0.056592	0.112121	0.352584	1.0	1.0
0.5	0	0.2	1.0	0.0	0.072356	0.000000	0.133333	0.352584	1.0	1.0
...	...	...	...	...	...	...	...	...	...	...

As insight, further research can review the use of other scalers such as StandardScaler (Z-Score Normalizer), Robust Scaler, MaxAbs Scaler, and so on. Whether or not there is an effect of these different data scaling techniques on the performance of the resulting model. After normalization, the next step is to balance the distribution of data classes using the SMOTE-ENN technique with $k=5$.

Since SMOTE-ENN applies data reduction, the data will be reduced after this sub-stage. There are two reasons for this: to remove noise data so that the data is of higher quality, and to reduce sample overlap

resulting from the SMOTE process because this technique produces synthetic data that may have similar values [56]. Before model development, the dataset was first divided into training and testing sets using a ratio of 87.5% and 12.5%. The SMOTE-ENN balancing technique was then applied only to the training data to improve class distribution and remove noisy samples, while the testing data remained unchanged. After applying SMOTE-ENN to the training set, the number of training samples became 443 rows. This balanced training dataset was then used to build the base learners in the BEL model.

Of the 443 data rows used, there were several synthesis classes that were filtered by this technique. However, the difference in distance between data classes became more significant, making it easier for the ML algorithm to differentiate between them. This opens up another avenue for research, namely comparison using other data class balancing techniques such as SMOTE, SMOTE-Tomek, and ADASYN, which of course have their own advantages, disadvantages, and effects on model performance.

The pre-processed training data consists of 443 rows, divided into two parts, with a ratio of 87.5% for the training data consisting of 387 rows and 12.5% for the test data consisting of 56 rows. Then, the training data was further divided with a ratio of 68.5% for ensemble data consisting of 265 rows and 31.5% for blender data consisting of 122 rows. Visualization of the data before and after division is presented in Figures 3 and 4.

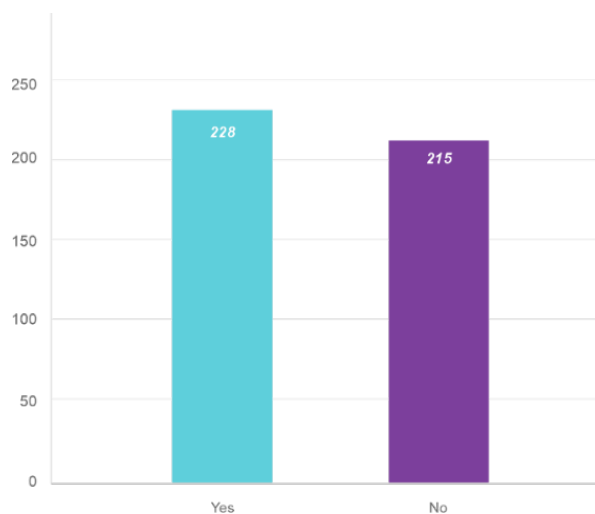


Figure 3. Data Class Distribution After Pre-processing Stage

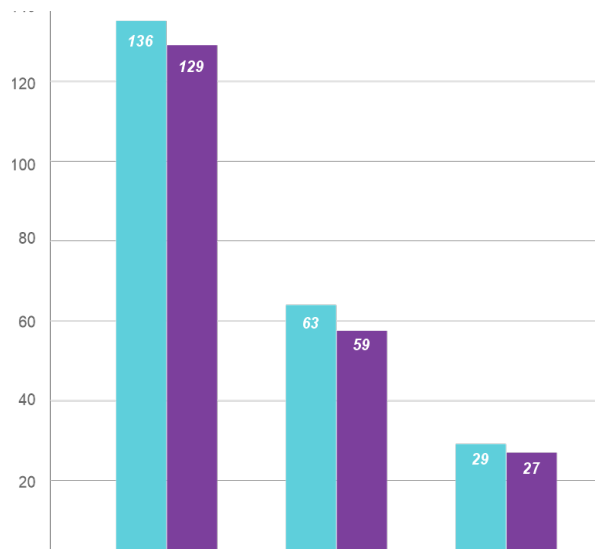


Figure 4. Distribution of Data Classes After Splitting

Model development on the base layer using three algorithms, KNN, DT, and NB, with ensemble data. Then, tested using blended data. The prediction results on the base layer are presented in Table 5. These

prediction results are new features used to train the algorithm in the meta layer, namely XGBoost. Further research allows for the addition of several other algorithms or the replacement of the algorithms currently used to compare model performance.

Table 5. Result of Base Layer Prediction

Matrix	KNN	Decision Tree	Naïve Bayes	XGBoost
Accuracy	87.50%	92.85%	82.14%	89.28%
Precision	85.71%	89.65%	81.48%	86.20%
Recall	88.88%	96.29%	81.48%	92.59%
F1-Score	87.27%	92.85%	81.48%	89.85%
AUC	93.23%	92.97%	82.22%	97.82%
Process Time	0.037	0.026	0.047	0.079

In these results, the DT model had the most optimal performance with the highest accuracy among the others, namely 92.85%. According to [43] DT has the fastest execution time, which is only 0.026 seconds. Datasets in the form of numbers, features that are not too complex, normalization, and balanced classes are suitable for DT. KNN is quite stable, although not as optimal as DT. However, the opposite results were obtained by NB.

NB, which is based on probability, does not match the dataset because it assumes that all features are independent. This is in line with what is stated by [45] One of the weaknesses of NB is its inability to recognize relational patterns or linearity in features. In addition, NB is sensitive to data imbalance. Even after SMOTE-ENN has been applied, the model still has difficulty distinguishing minority classes. [57].

Next, model development on the meta layer using XGBoost. As a result, after a single test, the BEL model achieved an accuracy of 96.42%, precision of 96.29%, recall of 96.29%, F1-Score of 96.29%, and AUC of 98.72%. These results prove that XGBoost as a meta model can optimize predictions from previous data residues. The model's ability to discriminate between positive and negative classes is quite good, as shown in Figure 5.

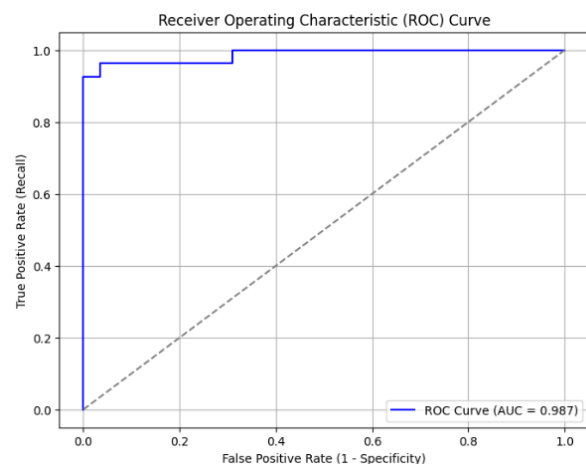


Figure 5. AUC Results

When examining the development of the BEL model from the outset, the results cannot be separated from the role of KNN and Naïve Bayes, which often produce different decisions from DT. As a result, DT has the highest performance. The three complement each other's weaknesses, so that XGBoost as a meta model can utilize the strengths of the base model to learn patterns more comprehensively rather than simply voting.

Regarding the use of XGBoost, as a meta-learner, it is possible that this model would have optimal performance if used alone. However, the results were not so. The single XGBoost model performed no better than the single Decision Tree, the best baseline model in this case study. This confirms that model blending improvements are a combination of the models and techniques used, not solely dominated by the meta layer.

These results may be considered less than optimal compared to high-performance single models such as 5C-SVM [58] and pure SVM [59]. However, single models have the disadvantage of being unable to extract significant information [60], so the ensemble approach can be an alternative because each model can address different types of data patterns and risks that cannot be captured using a single algorithm [61].

Testing the model using K-Fold cross validation with $k=10$ proves that the developed model is quite stable in the face of various data variations. With an average accuracy of 96.84%, precision of 98.09%, recall of 95.57%, F1-Score of 96.70%, AUC of 99.32%, and execution time of 0.261 seconds, which is very fast because it is less than one second. The test results are presented in Table 6.

Table 6. K-Fold Cross Validation Test Results

Matrix	Iteration										Mean
	1	2	3	4	5	6	7	8	9	10	
Accuracy	95.55	97.77	93.33	100	97.72	95.45	100	97.72	95.45	95.45	96.84
Precision	100	100	100	100	100	93.10	100	100	92.59	95.23	98.09
Recall	91.66	94.44	85	100	94.44	100	100	95	100	95.23	95.57
F1-Score	95.65	97.14	91.89	100	97.14	96.42	100	97.43	96.15	95.23	96.70
AUC	98.90	99.79	98.80	100	99.67	98.03	100	99.16	99.15	99.68	99.32
Process Time	0.114	0.817	0.877	0.072	0.130	0.115	0.092	0.094	0.112	0.185	0.261

From these results, it can be seen that the lowest accuracy occurred in the third iteration with a value of 93.33% and the highest in the fourth and seventh iterations. However, the average accuracy was 96.84%. These results are higher than the single model test, which had an accuracy of 96.42%, a difference of 0.42%. This relatively small difference indicates that the increase in accuracy is quite significant without sacrificing efficiency.

Accuracy indicates the model's ability to correctly classify all data, and the average value of 96.84% shows that the model has a high and consistent level of generalization across various data subsets. The lowest fluctuation in the third iteration (93.33%) was caused by variations in the data distribution in that fold, specifically the presence of ambiguous data between the positive and negative classes, which made it difficult for the model to distinguish patterns optimally.

This is also evident from the recall value dropping to 85%, indicating that the model failed to capture some of the positive data (resulting in an increase in false negatives), even though precision remained high (100%), meaning that the positive predictions generated were still accurate but fewer in number. The combined effect of these factors caused the F1-Score to decrease as well due to the imbalance between precision and recall.

Meanwhile, the high average AUC value of 99.32% indicates that the model has excellent discriminatory power in distinguishing between classes, and the drop to 98.03% in the sixth iteration does not indicate an error, but rather suggests that there is a higher degree of

feature overlap between classes in that subset, causing the ROC curve to decline slightly.

Finally, the longest execution time in the third iteration—0.877 seconds—was influenced by the complexity of the data in that fold, such as a more diverse feature distribution or a higher number of internal model iterations, particularly in XGBoost. Due to difficulties in finding the optimal split, making the training process more resource-intensive than in other iterations.

The improvement in performance is not solely due to the combination of models, but is primarily driven by the complementary learning behavior of the base models. KNN captures local proximity patterns, DT model hierarchical non-linear interactions, whilst NB provides probabilistic generalization under the assumption of independence. This diversity reduces both model variance and bias simultaneously, enabling the ensemble to achieve better generalization.

The precision matrix obtained a high average of 98.09%, and seven out of ten iterations obtained a perfect score of 100%. The precision value indicates that, overall, the model can accurately predict positive classes. This means that the model has high accuracy in minimizing false positives, which are errors in classifying negative samples as positive.

Meanwhile, the recall value in single testing was 96.29%, which was higher than the K-Fold Cross Validation average of 95.57%. This decrease was due to variations in the subsets in each iteration, which affected the distribution of positive classes learned by

the model. Nevertheless, both recall values were still quite high, indicating that the model was able to consistently handle positive classes well.

F1-Score is a combined metric of recall and precision that provides a comprehensive overview of model performance, especially in relation to class imbalance. Single testing showed a value of 96.29% and a K-Fold average of 96.70%. These results indicate that the model can maintain consistent performance in class imbalance between positive and negative classes.

An AUC value of 98.72% in a single test shows that the model has a good ability to distinguish between positive and negative classes. The average K-Fold AUC of 99.32% shows that the model is consistent, stable, and slightly better in its discriminatory ability across data subsets, thereby reducing errors in distinguishing between the two classes.

All iterations can be executed in a short time, namely 0.261 seconds. The fastest time was recorded in the 4th iteration, namely 0.072 seconds, and the slowest was recorded in the 3rd iteration, namely 0.877 seconds. The fourth iteration is the most optimal iteration where all matrices produce a perfect score of 100% and the fastest execution, indicating that the data subset variation in this section is easy to train and test.

One of the more consistent model development approaches is hybridization or ensemble learning, as in this study. Previously, we mentioned the advantages of ensemble learning, which is able to recognize different types of data patterns and risks that cannot be captured by a single model. However, there are trade-offs, such as high memory usage and long execution times.

Direct comparison based solely on accuracy can be misleading, as different studies utilize datasets with varying feature distributions, imbalance ratios, and preprocessing pipelines. Therefore, the superior performance observed in this study should be interpreted in the context of dataset characteristics, particularly the use of SMOTE-ENN which significantly alters class separability.

Study that conducted by [20], which explored the combination of DL and ML by utilizing data balancing techniques such as SMOTE and SMOTE-Tomek. DL architectures such as DenseNet and ResNet proved to be superior with an accuracy of 92% and an AUC of 95%. Although the results are quite high, the use of DL architecture requires greater computational resources, making it less efficient for practical implementation on a large scale.

Actually, the weakness of DL is the same as that experienced in Ensemble Learning due to the consequences of model accumulation, repeated training processes, and the need to aggregate many outputs, as well as the practical implications described earlier. As a solution, you can apply the pruning technique, which involves sorting, selecting, and then removing models that are considered less than optimal [51].

Additionally, efficiency techniques can be used by leveraging the built-in hyperparameters of the algorithms employed, such as input features and tree depth, as well as the number of parameters [62]. It is hoped that these two solutions can mitigate the issues encountered in DL and ensemble learning and serve as topics for further research.

Research [21] provides evidence that the use of ensemble learning with the SMOTE technique can significantly improve model performance. The XGBoost model in this study was able to achieve accuracy, recall, precision, and F1-Score above 95%, indicating solid performance. However, this study did not consider the aspect of execution time efficiency as an important factor in model implementation.

Meanwhile, research [22] proposed ensemble learning with voting techniques and produced an accuracy of 87.26%, but this performance still lags behind other more adaptive and modern ensemble methods such as SEL or BEL. Meanwhile, research conducted with the BEL algorithm achieved a performance of 96.42%, which means there has been a significant improvement.

When compared to previous relevant research results, this study builds and applies a BEL model with a base model of KNN, DT, and NB, as well as a meta model of XGBoost. The pre-processing stage uses MinMaxScaler and SMOTE-ENN, which not only balance the data distribution but also reduce noise.

The test results show that the BEL model is capable of achieving an accuracy of 96.42%, precision of 96.29%, recall of 96.29%, F1-Score of 96.29%, and AUC of 98.72%. In addition, testing with the K-Fold Cross Validation technique with $k=10$ yielded fairly stable performance with an average accuracy of 96.84%, precision of 98.09%, recall of 95.57%, F1-Score of 96.70%, AUC of 99.32%, and execution time of 0.261 seconds.

4. Conclusions

This study demonstrates that the application of Blending Ensemble Learning (BEL) with KNN, Decision Tree (DT), and Naive Bayes (NB) as base models and XGBoost as the meta model is capable of improving loan eligibility prediction performance. Support in the pre-processing stage using MinMaxScaler and SMOTE-ENN contributes to improving data quality through normalization, class balancing, and noise reduction.

Single testing results show an accuracy of 96.42%, precision of 96.29%, recall of 96.29%, F1-Score of 96.29%, and AUC of 98.72%. Meanwhile, K-Fold cross validation testing obtained an average accuracy of 96.84%, precision of 98.09%, recall of 95.57%, F1-Score of 96.70%, and AUC of 99.32% with an execution time of 0.261 seconds. These results indicate that the BEL model provides stable and consistent performance on the dataset used in this study.

One limitation of this study is the relatively small dataset size derived from a public Kaggle repository. Although the proposed method demonstrates strong performance on this dataset, further validation using larger, more diverse, and real-world banking datasets is necessary to confirm the model's robustness and generalizability in practical financial environments. In general, this research contributes improvements in the pre-processing stage, prediction performance, and evaluation techniques. Future research may explore the use of other data balancing techniques such as SMOTE-Tomek and ADASYN, the application of metaheuristic optimization algorithms such as Genetic Algorithm (GA), and the integration of alternative base learners or meta learners to enhance ensemble performance.

Further experimentation on real banking datasets or larger and more heterogeneous datasets is recommended to improve model generalization for industrial-scale implementation. Future developments are expected not only to improve prediction performance but also to address computational efficiency, memory usage, scalability, and model reliability so that the system can be integrated into financial decision-support environments without compromising customer data privacy.

From a practical perspective, the improved stability of the blending model has direct implications for risk management. Higher precision reduces the likelihood of approving high-risk applicants (reducing default rates), while stable recall ensures that eligible customers are not unnecessarily rejected, thereby supporting financial inclusion. This balance is critical in real-world banking systems where both risk minimization and market expansion must be achieved simultaneously.

Another area that can be explored is research that emphasizes data security and privacy using Differential Privacy (DP) and Explainable AI (XAI) [18], which is claimed to be quite effective in maintaining user data confidentiality. Combined with a BEL model that performs well in terms of prediction, the hope is that there will be improvements in the trade-off between privacy and model performance.

References

- [1] I. Mavlutova, A. Spilbergs, A. Verdenhofs, A. Natrins, I. Arefjevs, and T. Volkova, "Digital Transformation as a Driver of the Financial Sector Sustainable Development: An Impact on Financial Inclusion and Operational Efficiency," *Sustainability*, vol. 15, no. 1, 2023, doi: [10.3390/su15010207](https://doi.org/10.3390/su15010207).
- [2] X. Haoran, M. Wenlong, and Z. Siyu, "Digital technology development and systemic financial risks: Evidence from 22 countries," *Borsa Istanbul Review*, vol. 24, pp. 1–9, 2024, doi: <https://doi.org/10.1016/j.bir.2024.08.002>.
- [3] Y. Wang, J. Wang, H. Zhang, and J. Song, "Bridging prediction and decision: Advances and challenges in data-driven optimization," *Ynexus*, vol. 2, no. 1, p. 100057, 2025, doi: <https://doi.org/10.1016/j.ynexus.2025.100057>.
- [4] A. Szukits and P. Möriz, "Towards data-driven decision making: the role of analytical culture and centralization efforts," *Review of Managerial Science*, vol. 18, no. 10, pp. 2849–2887, 2024, doi: [10.1007/s11846-023-00694-1](https://doi.org/10.1007/s11846-023-00694-1).
- [5] X. Zhang et al., "Data-Driven Loan Default Prediction: A Machine Learning Approach for Enhancing Business Process Management," *Systems*, vol. 13, no. 7, 2025, doi: [10.3390/systems13070581](https://doi.org/10.3390/systems13070581).
- [6] N. Uddin, Md. K. Uddin Ahamed, M. A. Uddin, Md. M. Islam, Md. A. Talukder, and S. Aryal, "An ensemble machine learning based bank loan approval predictions system with a smart application," *International Journal of Cognitive Computing in Engineering*, vol. 4, pp. 327–339, 2023, doi: <https://doi.org/10.1016/j.ijcee.2023.09.001>.
- [7] X. Zhu, Q. Chu, X. Song, P. Hu, and L. Peng, "Explainable prediction of loan default based on machine learning models," *Data Science and Management*, vol. 6, no. 3, pp. 123–133, 2023, doi: <https://doi.org/10.1016/j.dsm.2023.04.003>.
- [8] L. Monje, R. A. Carrasco, and M. Sánchez-Montañés, "Machine Learning XAI for Early Loan Default Prediction," *Comput. Econ.*, 2025, doi: [10.1007/s10614-025-10962-9](https://doi.org/10.1007/s10614-025-10962-9).
- [9] A. Citterio, "Bank failure prediction models: Review and outlook," *Socioecon. Plann. Sci.*, vol. 92, p. 101818, 2024, doi: <https://doi.org/10.1016/j.seps.2024.101818>.
- [10] V. Chang, S. Sivakulasingam, H. Wang, S. T. Wong, M. A. Ganatra, and J. Luo, "Credit Risk Prediction Using Machine Learning and Deep Learning: A Study on Credit Card Customers," *Risks*, vol. 12, no. 11, 2024, doi: [10.3390/risks12110174](https://doi.org/10.3390/risks12110174).
- [11] A. Amato, J. R. Osterrieder, and M. R. Machado, "How can artificial intelligence help customer intelligence for credit portfolio management? A systematic literature review," *International Journal of Information Management Data Insights*, vol. 4, no. 2, p. 100234, 2024, doi: <https://doi.org/10.1016/j.ijime.2024.100234>.
- [12] A. Alamsyah, A. A. Hafidh, and A. D. Mulya, "Innovative Credit Risk Assessment: Leveraging Social Media Data for Inclusive Credit Scoring in Indonesia's Fintech Sector," *Journal of Risk and Financial Management*, vol. 18, no. 2, 2025, doi: [10.3390/jrfm18020074](https://doi.org/10.3390/jrfm18020074).
- [13] U. Sivarajah, M. M. Kamal, Z. Irani, and V. Weerakkody, "Critical analysis of Big Data challenges and analytical methods," *J. Bus. Res.*, vol. 70, pp. 263–286, 2017, doi: <https://doi.org/10.1016/j.jbusres.2016.08.001>.
- [14] P. Singh, "Systematic review of data-centric approaches in artificial intelligence and machine learning," *Data Science and Management*, vol. 6, no. 3, pp. 144–157, 2023, doi: <https://doi.org/10.1016/j.dsm.2023.06.001>.
- [15] F. M. Surur et al., "Unlocking the power of machine learning in big data: a scoping survey," *Data Science and Management*, 2025, doi: <https://doi.org/10.1016/j.dsm.2025.02.004>.
- [16] E. Dritsas and M. Trigka, "Exploring the Intersection of Machine Learning and Big Data: A Survey," *Mach. Learn. Knowl. Extr.*, vol. 7, no. 1, 2025, doi: [10.3390/make7010013](https://doi.org/10.3390/make7010013).
- [17] Y. Wu, B. Sicard, and S. A. Gadsden, "Physics-informed machine learning: A comprehensive review on applications in anomaly detection and condition monitoring," *Expert Syst. Appl.*, vol. 255, p. 124678, 2024, doi: <https://doi.org/10.1016/j.eswa.2024.124678>.
- [18] M. M. Hossain, M. Mamun, A. Munir, M. M. Rahman, and S. H. Chowdhury, "A Secure Bank Loan Prediction System by Bridging Differential Privacy and Explainable Machine Learning," *Electronics (Switzerland)*, vol. 14, no. 8, Apr. 2025, doi: [10.3390/electronics14081691](https://doi.org/10.3390/electronics14081691).
- [19] X. Zhang et al., "Data-Driven Loan Default Prediction: A Machine Learning Approach for Enhancing Business Process Management," *Systems*, vol. 13, no. 7, Jul. 2025, doi: [10.3390/systems13070581](https://doi.org/10.3390/systems13070581).
- [20] E. Hussein Sayed, A. Alabrah, K. Hussein Rahouma, M. Zohaib, and R. M. Badry, "Machine Learning and Deep Learning for Loan Prediction in Banking: Exploring Ensemble Methods and Data Balancing," *IEEE Access*, vol. 12, pp. 193997–194019, 2024, doi: [10.1109/ACCESS.2024.3509774](https://doi.org/10.1109/ACCESS.2024.3509774).
- [21] F. Muhammad, J. C. Halim, H. Lucky, and D. Suhartono, "Loan Eligibility Prediction Using Ensemble Machine Learning Techniques and SMOTE," in *2024 14th International Conference on System Engineering and Technology, ICSET 2024 - Proceeding*, Institute of Electrical and Electronics Engineers Inc., 2024, pp. 102–107, doi: [10.1109/ICSET63729.2024.10775270](https://doi.org/10.1109/ICSET63729.2024.10775270).
- [22] N. Uddin, M. K. Uddin Ahamed, M. A. Uddin, M. M. Islam, M. A. Talukder, and S. Aryal, "An ensemble machine learning

- imbalanced health data,” *Expert Syst.*, vol. 41, no. 8, Aug. 2024, doi: [10.1111/exsy.13596](https://doi.org/10.1111/exsy.13596).
- [57] S. DEMİR and E. K. ŞAHİN, “Evaluation of Oversampling Methods (OVER, SMOTE, and ROSE) in Classifying Soil Liquefaction Dataset based on SVM, RF, and Naïve Bayes,” *European Journal of Science and Technology*, Feb. 2022, doi: [10.31590/ejosat.1077867](https://doi.org/10.31590/ejosat.1077867).
- [58] J. Amalia, A. J. M. Manalu, J. N. P. Ambarita, and D. Sihombing, “Creditworthiness Classification Utilizing AHP-SVM Based on 5C Criteria,” *sinkron*, vol. 9, no. 3, pp. 1788–1795, Aug. 2025, doi: [10.33395/sinkron.v9i3.15049](https://doi.org/10.33395/sinkron.v9i3.15049).
- [59] T. Berhane, T. Melese, and A. M. Seid, “Performance Evaluation of Hybrid Machine Learning Algorithms for Online Lending Credit Risk Prediction,” *Applied Artificial Intelligence*, vol. 38, no. 1, 2024, doi: [10.1080/08839514.2024.2358661](https://doi.org/10.1080/08839514.2024.2358661).
- [60] Y. Salaheldin, S. Abdelhady, R. Abdallah, A. M. Fawzy, and M. A. Mohamed, “A Data Driven Model for predicting Loan Approval Using Machine Learning Approaches,” *ERU Research Journal*, vol. 4, no. 1, pp. 2271–2289, Jan. 2025, doi: [10.21608/erurj.2025.291674.1149](https://doi.org/10.21608/erurj.2025.291674.1149).
- [61] L. Wei, J. Dong, and H. Yu, “NERHF: a hybrid machine learning-driven efficient credit risk control framework,” *Sci. Rep.*, Dec. 2025, doi: [10.1038/s41598-025-30905-6](https://doi.org/10.1038/s41598-025-30905-6).
- [62] E. A. Firdaus, J. Manurung, H. Saragih, and M. A. Prabukusumo, “Optimization of XGBoost hyperparameters using grid search and random search for credit card default prediction,” *Jurnal Mandiri IT*, vol. 14, pp. 269–280, Oct. 2025. doi: <https://doi.org/10.35335/mandiri.v14i2.468>.