Indonesian Journal of Electronics, Electromedical Engineering, and Medical Informatics e-ISSN: 2656-8624
Homepage: https://ijeeemi.org/; Vol. 7, No. 2, pp. 203-220, May 2025

RESEARCH PAPER

OPEN ACCES

Evaluation of the Impact of SMOTEENN on Monkeypox Case Classification Performance Using Boosting Algorithms

Laifansan Siena[1], Triando Hamonangan Saragih[1], Radityo Adi Nugroho[1], Dwi Kartini[1], Muliadi[1], and Wahyu Caesarendra[2]

[1] Department of Computer Science, Lambung Mangkurat University, Banjarbaru, Indonesia
[2] Department of Mechanical and Mechatronics Engineering, Faculty of Engineering and Science, Curtin University Malaysia, Sarawak, Malaysia

ABSTRACT

Monkeypox is a zoonotic disease with increasing global prevalence, posing a significant challenge in healthcare. Its widespread transmission necessitates more accurate detection systems to assist medical professionals in diagnosing and managing cases effectively. One of the main challenges in developing monkeypox prediction models is class imbalance in datasets, which can cause models to favor the majority class and reduce predictive accuracy for rarer cases. To address this issue, this study evaluates the effectiveness of the SMOTEENN resampling technique in improving the classification performance of monkeypox cases. Three boosting algorithms Gradient Boosting, XGBoost, and LightGBM were applied to a monkeypox dataset consisting of 25,000 samples. The data preprocessing steps included handling missing values, feature encoding, and feature scaling. The dataset was then balanced using SMOTEENN, a hybrid technique combining the Synthetic Minority Over-sampling Technique (SMOTE) and Edited Nearest Neighbors (ENN). Additionally, hyperparameter tuning with GridSearchCV was performed to optimize model performance by systematically selecting the best parameter combinations. The results indicate that applying SMOTEENN significantly improved classification accuracy, achieving a maximum of 69%, with an F1-score of 67%. Compared to previous studies, the proposed approach demonstrated superior performance in handling class imbalance and enhancing classification robustness. These findings highlight the potential of SMOTEENN and boosting algorithms in medical data classification, particularly for infectious diseases with imbalanced datasets. This study contributes to the development of more reliable machine learning techniques for improving disease detection, classification accuracy, and overall model generalization. Future research should explore additional resampling techniques, deep learning architectures, and feature selection methods to further improve predictive performance in medical diagnostics.

PAPER HISTORY

Received Feb. 20, 2025
Accepted April 20, 2025
Published April 23, 2025

KEYWORDS

Monkeypox;
SMOTEENN;
Boosting;
Classification;
Class Imbalance

AUTHOR EMAIL

laifansansiena89@gmail.com
triando.saragih@ulm.ac.id
radityo.adi@ulm.ac.id
dwikartini@ulm.ac.id
muliadi@ulm.ac.id

1. INTRODUCTION

The Monkeypox is a viral infection caused by the monkeypox virus, a member of the Orthopoxvirus family. Since its first identification in 1958, monkeypox has been recognized as a zoonotic disease that can be transmitted from animals to humans. The first human case was recorded in the Democratic Republic of the Congo in 1970. Over time, its spread has increased significantly across various countries, with a total of 88,122 cases reported worldwide, including more than 30,000 cases in the United States [1]. The high number of cases indicates that monkeypox remains a global health threat, necessitating more accurate detection and classification strategies.

However, one of the main challenges in classifying monkeypox cases is class imbalance in the dataset. Class imbalance occurs when the number of samples in one class is significantly larger than in the other. This can lead to a classification model that favors the majority class while ignoring the minority class, ultimately reducing the predictive accuracy for rare cases [2]. Previous studies have attempted to address this issue using various approaches, but they still have limitations in improving the accuracy of classification models [3].

Previous studies have explored various approaches to classifying monkeypox cases. Anugrah et al. (2024) [4] conducted research using a dataset of 5,000 monkeypox cases and applied the Support Vector Machine (SVM) method, achieving an accuracy of 65%. He suggested

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

increasing the dataset size and conducting further research with a more balanced dataset. Subsequently, Cindy et al. (2024) [3] employed an even larger dataset with 25,000 samples and applied resampling techniques, namely oversampling and undersampling (ROS and RUS), to address class imbalance. The results showed that accuracy on the imbalanced dataset (67.1%) was higher than on the balanced dataset (36.5%). However, the AUC value for the imbalanced dataset (0.6) was only slightly higher than that of the balanced dataset (0.4), indicating that while the model on the imbalanced data was more accurate in recognizing the majority class, its overall ability to distinguish between both classes remained limited.

Based on the limitations of previous studies, this research proposes the use of the SMOTEENN resampling technique for data balancing [5]. SMOTEENN is a combination of the Synthetic Minority Oversampling Technique (SMOTE) and Edited Nearest Neighbor (ENN) [6]. SMOTE is one of the most popular resampling techniques for handling data imbalance [7]. This technique generates synthetic data based on the k-Nearest Neighbor (k-NN) of each minority class instance [8]. However, despite its effectiveness in addressing overfitting caused by oversampling, SMOTE has a drawback: it can introduce noise and cause synthetic minority class data to be misclassified as part of the majority class [8]. To overcome this weakness, this study integrates SMOTE with ENN [7]. ENN functions to remove samples that may be noise or misclassified, thereby refining decision boundaries and potentially reducing overfitting [9]. By combining the advantages of both oversampling and undersampling in a single process, SMOTEENN enhances overall data quality [10]. The study conducted by Gao et al. (2020) [11] demonstrated that a model utilizing the SMOTEENN resampling technique achieved an accuracy of 95%, outperforming other resampling methods.

After balancing the data using SMOTEENN, the boosting method was employed as the classification approach in this study, as it is known to enhance classification performance [12]. Boosting is one of the most popular categories of ensemble learning, building a strong classification model by sequentially combining multiple weak models [7]. This study utilizes three boosting algorithms: Gradient Boosting (GB), XGBoost (Extreme Gradient Boosting), and LightGBM (Light Gradient Boosting Machine). GB constructs a decision tree-based model by iteratively correcting prediction errors at each stage [13]. XGBoost is an optimized version of GB with additional features to control overfitting and handle missing values more efficiently [14]. Meanwhile, LightGBM is designed to improve efficiency and scalability, especially for large datasets [15]. In addition to selecting the appropriate classification method, hyperparameter tuning is also a crucial aspect in improving model performance [16]. In this study, the Grid Search method was used to find the best hyperparameter combination for the applied boosting models. This method performs an exhaustive search over various parameter combinations to optimize the training algorithm [17].

This study aims to evaluate the effectiveness of the SMOTEENN technique in improving the performance of an imbalanced monkeypox classification model. The contributions of this research are as follows: 1) developing a classification model that supports decision-making systems for monkeypox diagnosis and management by addressing data imbalance with SMOTEENN, 2) contributing to the advancement of SMOTEENN as a solution for handling class imbalance in medical datasets, 3) enhancing the accuracy of monkeypox detection and classification, which can accelerate response and preventive measures, and 4) providing empirical insights into the impact of SMOTEENN on improving classification performance, serving as a reference for future studies in medical data resampling techniques.

This paper is structured as follows: Section II presents the dataset, the proposed methodology, and the training and testing schemes. Section III discusses the data distribution before and after resampling, the results of hyperparameter tuning, and the evaluation of the classification model. Section IV provides an interpretation of the results, a comparison with previous studies, and the study's limitations. Finally, Section V concludes the study by restating the objectives, summarizing key findings, and outlining directions for future research.

2. MATERIALS AND METHOD

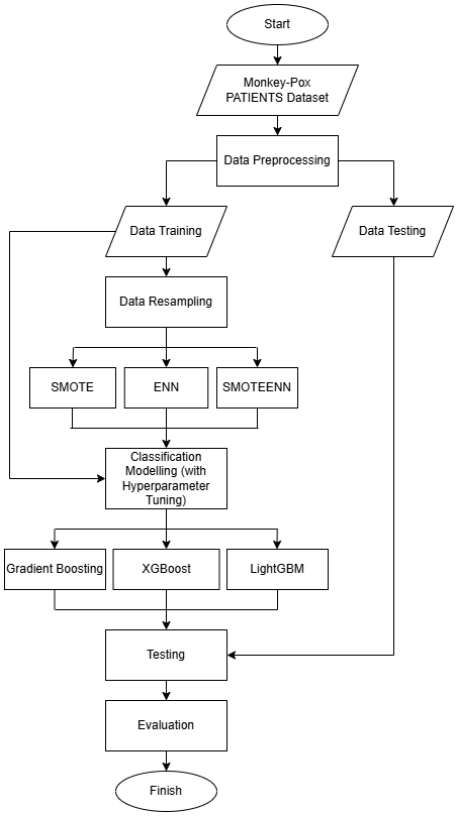

Fig. 1. Research flowchart of monkeypox classification using SMOTE, ENN, and SMOTEENN with boosting algorithms.

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

This study involves the implementation of three data resampling methods: Synthetic Minority Over-sampling Technique (SMOTE), Edited Nearest Neighbors (ENN), and the combination of SMOTE-ENN to address class imbalance in the Monkeypox dataset. Additionally, hyperparameter tuning using GridSearchCV is applied to optimize the performance of three boosting-based machine learning algorithms: Gradient Boosting, XGBoost, and LightGBM.

This study consists of seven main stages: data collection using the Monkeypox dataset, data preprocessing, data splitting, data resampling, hyperparameter tuning, classification modeling, and model evaluation. Fig. 1 illustrates the complete research workflow, outlining each process sequentially.

A. Data Collection

The dataset used in this study is the Monkey-Pox PATIENTS Dataset, which was downloaded from the Kaggle platform via the link: https://www.kaggle.com/datasets/muhammad4hmed/monkeypox-patients-dataset. This dataset is synthetic data generated based on a study published by The BMJ titled Clinical features and novel presentations of human monkeypox in a central London centre during the 2022 outbreak: descriptive case series. The study discusses the clinical characteristics and novel presentations of monkeypox in patients treated at a healthcare center in London during the 2022 outbreak [18].

This dataset contains a total of 25,000 patient records, each with 11 attributes that capture key clinical symptoms and conditions associated with monkeypox. The target variable, labeled MonkeyPox, is a binary variable indicating whether a patient has been diagnosed with monkeypox (1 for positive cases, 0 for negative cases). Meanwhile, the remaining 10 attributes serve as independent variables, which act as predictors for classification modeling. For the purpose of this study, the Patient_ID attribute was excluded from the analysis, as it does not contribute to the classification process and is irrelevant to model development. A detailed description of each feature included in the dataset is provided in Table 1.

Table 1. Description of dataset features, including feature names, data types, and corresponding descriptions.

Feature	Data Type	Description
Patient_ID	Categorical	Unique ID for each patient (P0, P1, P2, ... , P24999)
Systemic Illness	Categorical	Systemic illness experienced by the patient
Rectal Pain	Boolean	Indicates presence of rectal pain (True/False)
Sore Throat	Boolean	Indicates presence of sore throat (True/False)
Penile Oedema	Boolean	Indicates penile oedema (True/False)
Oral Lesions	Boolean	Indicates presence of oral lesions (True/False)
Solitary Lesion	Boolean	Indicates presence of solitary lesion (True/False)
Swollen Tonsils	Boolean	Indicates swollen tonsils (True/False)
HIV Infection	Boolean	Indicates HIV infection (True/False)
Sexually Transmitted Infection	Boolean	Indicates presence of any STI (True/False)
Monkeypox	Categorical	Monkeypox status (Positive/Negative) as the target variable

B. Data Preprocessing

The data preprocessing stage is a crucial step in developing a machine learning model to enhance data quality and model performance [19]. In this study, the data preprocessing process consists of several key steps, including handling missing values, removing irrelevant features, feature encoding, and feature scaling. Each step is explained as follows:

1) Handling Missing Values

Missing values can introduce bias, reduce efficiency, and hinder the analysis process if not properly addressed [20]. In this study, missing values were found in the Systemic Illness attribute, with a total of 6,216 missing entries. To address this issue, we applied the Iterative Imputer method, which estimates missing values by modeling each feature with missing data as a function of other features in an iterative manner [21].

2) Removing Irrelevant Features

The removal of irrelevant features aims to reduce model complexity and improve prediction accuracy. In this study, the Patient_ID feature was removed because it had no direct correlation with the prediction and was only used as a unique patient identifier.

3) Feature Encoding

In this study, Label Encoding was applied to the Systemic Illness and MonkeyPox attributes to convert categorical values into numerical representations. This method directly transforms text data into integer values with nominal meaning, without considering order or ranking [22]. The MonkeyPox attribute was encoded into binary values, where Positive was mapped to 1 and Negative to 0. Other boolean attributes were already in True and False format, which were directly interpreted as 1 and 0 without additional encoding.

4) Feature Scaling

Feature scaling is a crucial step to ensure that all features have a uniform value range, which accelerates the

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

training process and enhances algorithm performance [23]. In this study, Standard Scaler was applied to standardize features by removing the mean and scaling to unit variance [24]. This method ensures that each feature has a mean of zero and a standard deviation of one, preventing certain features from dominating the model due to different scales [24]. The mathematical formulation of Standard Scaler is presented in Eq. (1):

$$Z = \frac{X_i - \bar{X}}{\sigma} \quad (1)$$

Where X_i represents the original value, $\bar{X}$ is the mean of the sample, and σ is the standard deviation [24].

C. Data Splitting

In this study, the dataset was divided into training and testing subsets using the train_test_split function from the scikit-learn library. This function splits arrays or matrices into random train and test subsets, facilitating the evaluation of machine learning models [25]. Specifically, 70% of the data was allocated for training, and 30% for testing, as indicated by the test_size=0.3 parameter. To ensure reproducibility of the results, the random_state=42 parameter was set [25]. Additionally, the stratify=y parameter was used to maintain the original distribution of the target variable across both subsets, which is particularly important when dealing with imbalanced datasets [25].

D. Data Resampling

Class imbalance in classification problems occurs when the number of observations in one class is significantly lower than in another [26]. An imbalanced class distribution can affect the learning process of the model, leading to bias toward the majority class while ignoring the minority class [2]. To address this imbalance, resampling techniques are applied to balance the class distribution, either through oversampling the minority class, undersampling the majority class, or a combination of both [27]. In the monkeypox dataset used in this study, the label distribution exhibits an imbalance, with 64% for the positive class and 36% for the negative class. To address this imbalance, this study employs three resampling methods: Synthetic Minority Over-sampling Technique (SMOTE), Edited Nearest Neighbors (ENN), and their combination (SMOTEENN).

The selection of SMOTE, ENN, and SMOTEENN in this study is based on the characteristics of the dataset, particularly the presence of class imbalance and the predominance of boolean-valued features. SMOTE was chosen for its ability to enhance the representation of the minority class by generating synthetic samples based on nearest neighbors without altering the majority class distribution [8]. ENN was employed to reduce noise from the majority class by removing samples misclassified by their neighbors [28]. The combination of both, known as SMOTEENN, integrates the strengths of SMOTE and ENN, allowing for class balance while improving training data quality [6], [10].

In the context of this monkeypox dataset, in which most of the input features are boolean, methods such as SMOTE and ENN remain relevant since they operate based on distances between samples (e.g., Euclidean), which can be effectively computed on binary-valued data. Alternative methods such as ADASYN were not applied in this study, as ADASYN may amplify noise in hard-to-classify regions [29]. Therefore, SMOTEENN was selected as the most appropriate resampling approach for the characteristics of the data in this study.

1) SMOTE

This technique generates synthetic data based on the k-Nearest Neighbors (k-NN) of each minority class instance, helping to mitigate overfitting caused by oversampling, although it may introduce noise [8]. According to [30], the distance between samples is calculated using the Euclidean formula, as shown in Eq. (2).

$$D = \sqrt{\sum_{k=1}^{n} (X_k - Y_k)^2} \quad (2)$$

Where D is the distance between two samples X and Y in an n-dimensional space. SMOTE selects K nearest neighbors and sets the sampling rate N based on the data imbalance. The process of generating new samples is described in Eq. (3) [30], where six nearest samples are selected and randomly connected.

$$X_{new} = x_i + rand(0,1) * |x_i - x_i| \quad (3)$$

The SMOTE algorithm [30] can be seen in Table 2.

Table 2. Steps of the SMOTE algorithm for oversampling the minority class in classification.

SMOTE	
Input:	Majority class sample set (X_{num}) and minority class sample set (X_{min}).
Output:	New sample set (X_{new}).
Step 1:	Calculate the Euclidean distance between two samples in X_{min} and obtain K nearest neighbor samples.
Step 2:	Calculate the sampling probability N.
Step 3:	Select the six nearest samples as a group, then randomly connect them using Equation (2) to generate new samples added to X_{new}.
Step 4:	Repeat Step 3 until the number of samples in X_{min} equals X_{num}.
END	

2) ENN

ENN removes samples considered anomalies if the majority of their nearest neighbors belong to a different class [28]. According to [31], the Edited Nearest Neighbors (ENN) technique eliminates data points whose class does not match the class of their K nearest neighbors. The K nearest neighbors of a sample X can be determined using Eq. (4) [31].

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

$$knn(X_i,k) = \{y_i \in X \big| dist(X_i,X_j) < dist(X_{ik},X_i)\} \quad (4)$$

where X_{ik} is the k-th nearest neighbor of sample X_i, and $dist(X_i,X_j)$ represents the Euclidean distance between X_i dan X_j. The ENN algorithm is described in Table 3 [31].

Table 3. Steps of the Edited Nearest Neighbors (ENN) algorithm for data undersampling.

ENN	
Input:	Dataset D, majority Sample X_{major}, number of nearest neighbours k, Under-sampling rate U.
Output:	Balanced dataset $X_{enn.}$
Step 1:	Set total deleted samples i.e. $C_{del} = 0$ and traverse all majority samples.
Step 2:	Find K nearest neighbors to X_{i_major} and store their indexes.
Step 3:	Using Eq. (3), find three nearest neighbours whose indexes are stores.
Step 4:	Compare the class of X_{i_major} to the class of neighbor samples. If found to be different from more than two classes then delete them.
Step 5:	$C_{del} = C_{del} + 1$; if $C_{del} <$ M, then go to step 3 Else, end operations on $X_{i\_major.}$
Step 6:	Deleted samples are removed from data set X and we get a balanced dataset as output $X_{enn.}$
END	

3) SMOTEENN

A hybrid technique that combines SMOTE and ENN to balance the dataset by retaining majority class samples while enhancing the representation of the minority class [6], [10]. The SMOTEENN algorithm is explained in Table 4 [32].

Table 4. Steps of the SMOTEENN algorithm combining SMOTE for oversampling and ENN for noise removal.

SMOTEENN	
Input:	Imbalance training data
Output:	Balanced training data
Step 1:	Randomly select X_i in minority classes.
Step 2:	Identify K nearest neighbors of X_i : ψ_{X_i}.
Step 3:	Generate $X_{new} = X_i + (\hat{X}_i - X_i) \times \delta$.
Step 4:	Does balancing ratio satisfy if no goto 1. else
Step 5:	Remove noise sample using ENN ENN { For every observation O Find the three nearest neighbors of O If O gets misclassified by its three nearest neighbours Then delete O End If End For }
END	

Table 4 describes the SMOTEENN algorithm as follows. Firstly, SMOTE determines the k-Nearest Neighbors (k-NNs), which is denoted by ψ_{X_i} for each minority sample $X_i \in \propto_{min}$. To generate a synthetic data sample X_{new} for X_i SMOTE randomly selects an element $\hat{X}_i$ in ψ_{X_i} and $\hat{X}_i$ in $\propto_{min}$. The feature vector of X_{new} is the sum of the feature vectors of X_i and the value, which can be obtained by multiplying the vector difference between $\hat{X}_i$ and X_i and a random value δ which is between 0 and 1. By doing so, we obtain a synthetic point along the line segment joining X_i and $\hat{X}_i$. Further, the Edited Nearest Neighbour (ENN) is applied to clean the overlapping of classes.

In this study, four resampling scenarios were conducted: No Resampling, SMOTE, ENN, and SMOTEENN.

E. Machine Learning Model

Machine learning enables researchers to discover hidden patterns and structures within data and utilize these patterns to develop models that support theoretical advancements [33]. Its ability to analyze and recognize patterns in unstructured data makes it a crucial tool for handling the increasing complexity and volume of data. By providing deeper insights and accurate predictions, machine learning expands the boundaries of human analysis, allowing for smarter and more efficient decision-making across various sectors [34].

In this study, three boosting-based machine learning algorithms Gradient Boosting, XGBoost, and LightGBM were utilized to classify monkeypox cases. These models were applied to the preprocessed and balanced monkeypox dataset to achieve optimal classification performance.

1) Gradient Boosting

This ensemble learning algorithm incrementally builds a predictive model by adding new decision trees to correct errors from the previous model. The process utilizes the gradient descent algorithm to minimize the loss function [35], [36]. The pseudocode for this algorithm is presented in Table 5 [37].

Table 5. Pseudocode of the Gradient Boosting algorithm for iterative model optimization using decision trees.

Gradient Boosting	
Input:	Training Data: = $(x_i,y_i)_{i=1}^n$ Where, x_i is a data point and y_i is the label for x_i Loss function: = $L\ (y_i,f(x))$
Output:	The decision tree function $f(x)$
Step 1:	Initialize the model $f(x)$: = argmin$\sum_{i=1}^n L(y_i,z)$.
Step 2:	$\boldsymbol{for}\ k\ =\ 1,2,3\ldots,K\ \boldsymbol{do}$
Step 3:	$\boldsymbol{for}\ I\ =\ 1,2,3\ldots,K\ \boldsymbol{do}$
Step 4:	By calculating the pseudo residual error:

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

$$R_{ki} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(y_i)}\right]_{f(x)=f_{k-1}(x)}$$

Step 5: **End**
Step 6: **End**
Step 7: By constructing a new Decision Tree $T_k(x;\theta_k)$, based on $R_{ki}, \theta_k = \{R_{kj} j = [1,2,3 \dots J]\}$
Step 8: $\boldsymbol{for}\ k = 1,2,3 \dots, K\ \boldsymbol{do}$
Step 9: $z_{kj} = argmin \sum_{x_i \in R_{ki}}^{n} L(y_i, f_{k-1}(x) + z)$
Step 10: **End**
Step 11: Updating the model $f_k(x) = f_{k-1}(x) + \sum_{j=1}^{j} z_{kj} I(x \hat{I} R_{kj})$
Step 12: $f(x) = \sum_{k=1}^{K} \sum_{j=1}^{J} z_{kj} I(x \hat{I} R_{kj})$

Table 5 outlines the key steps of the Gradient Boosting algorithm. In this implementation, Gradient Boosting uses decision trees as base learners, forming what is known as a Gradient Boosting Decision Tree (GBDT) model. The algorithm starts by initializing the model $f(x)$ with a value that minimizes the loss function. In each iteration k, the pseudo-residuals are computed to measure the error from the previous model. These residuals are then used to construct a new decision tree $T_k(x;\theta_k)$, which aims to correct the prediction errors. Once the tree is built, optimal weights for its nodes are determined by minimizing the loss function. The model is then updated by integrating the contributions of the newly added tree. This process is repeated until the maximum number of iterations is reached or a stopping criterion is met. By iteratively refining the model, Gradient Boosting improves its predictive accuracy by systematically reducing errors at each step.

2) EXtreme Gradient Boosting (XGBoost)

XGBoost is an advanced version of the Gradient Boosting algorithm, optimized for speed and performance by leveraging parallelization, regularization, and missing value handling. This model constructs decision trees sequentially based on residuals and combines them into a final prediction [38], [39]. The XGBoost algorithm is presented in Table 6 [40].

Table 6. Pseudocode of the XGBoost algorithm for gradient boosting with tree-based model optimization.

XGBoost	
Input:	Training data D, number of boosting rounds T
Output:	Final model $F_T(x)$
1	Initialize model $F_\theta(x)$ with a constant value
2	$\boldsymbol{for}\ t = 1\ to\ T\ \boldsymbol{do}$
3	Compute gradient g_i and hessian h_i for each data point
4	Build a new tree:
5	Start with root node containing all data
6	For each feature f do
7	Find best split based on gain formula
8	Split node into left and right children
9	Prune tree based on minimum gain threshold
10	Compute optimal weights for leaves
11	Update model:
12	$F_t(x) = F_{(t-1)}(x) + eta \times new\ tree$
13	End for
14	Return final model $F_t(x)$

Table 6 describes the XGBoost algorithm, which begins by initializing the initial model $F_\theta(x)$ with a constant value. In each iteration up to T boosting rounds, the algorithm calculates the gradient and hessian for each data point to determine the direction and magnitude of correction for the loss function. A new decision tree is then built by identifying the best split based on gain, followed by pruning to prevent overfitting. Subsequently, the optimal weights for each leaf node are computed, and the model is updated by adding the new tree scaled by the learning rate η. This process repeats until iteration T, resulting in the final model $F_T(x)$.

3) Light Gradient Boosting Machine (LightGBM)

LightGBM is a histogram-based boosting algorithm that employs a leaf-wise approach to construct trees asymmetrically, improving efficiency. Using gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB), LightGBM can process large datasets efficiently [41], [42], [43]. LightGBM applies a histogram-based algorithm to accelerate the search for optimal split points by discretizing continuous numerical data into multiple bins [44]. The pseudocode for the histogram-based LightGBM algorithm is shown in Table 7 [44].

Table 7. Pseudocode of the histogram-based LightGBM algorithm for efficient gradient boosting tree learning.

Histogram-based LightGBM	
Input:	I: sample data, GBDT(I):iterator
Output:	p_m, f_m, v_m
1	$\boldsymbol{for}\ each\ leaf\ p$ in GBDT(I) $\boldsymbol{do}$
2	$\boldsymbol{for}\ each\ f\ in\ I.\ Features\ \boldsymbol{do}$
3	H = new Histogram();
4	$\boldsymbol{for}\ i$ in (0, num_of_row) $\boldsymbol{do}$
5	H$[f.bins[i]].g$+= g_i;
6	H$[f.bins[i]].n$+= 1;
7	$\boldsymbol{for}\ i$ in (n, len(H)) $\boldsymbol{do}$
8	S_L+= $H[i].g$;
9	n_L+= $H[i].n$;
10	$S_R = S_P - S_L$;
11	$n_R = n_P - n_L$;
12	$\Delta loss = \frac{{S_L}^2}{n_L} + \frac{{S_R}^2}{n_R} + \frac{{S_P}^2}{n_P}$
13	$\boldsymbol{If}\ \Delta loss > \Delta loss(p_m, f_m, v_m)\ \boldsymbol{then}$
14	(p_m, f_m, v_m) = (p,f,H$[i].value$)

As shown in Table 7, the algorithm begins by iterating over each leaf in the Gradient Boosting Decision Tree (GBDT) model. For each feature in the dataset, a new histogram is created to compute the gradient distribution and sample count in each bin. The histogram construction process involves summing the gradients (g) and counting

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

the number of samples in each bin based on the grouped feature values. Once the histogram is built, the algorithm searches for the optimal split point by calculating the total gradient (S_L and S_R) and the sample count (n_L and n_R) on both sides of the split. Next, the algorithm computes the change in loss function ($\Delta loss$) based on the gradient difference before and after the split. If the $\Delta loss$ value is greater than the previous optimal value, the best split point is updated by storing the leaf index, selected feature, and optimal split value. By using this histogram-based approach, LightGBM reduces computational complexity, speeds up the training process, and enhances efficiency in handling large-scale datasets.

F. Hyperparameter Tuning

Hyperparameter tuning is the process of selecting the optimal combination of hyperparameters that yield the best performance [16]. Grid search is the method used in this study, where an exhaustive search is performed over the given parameter values to find the best combination [17]. This process systematically explores all predefined hyperparameter combinations in the form of a grid, as illustrated in Fig. 2 [17]. Each point in the grid represents a specific combination of hyperparameter values, and the model is trained and evaluated for every combination. This exhaustive search ensures that the best-performing set of hyperparameters is identified.

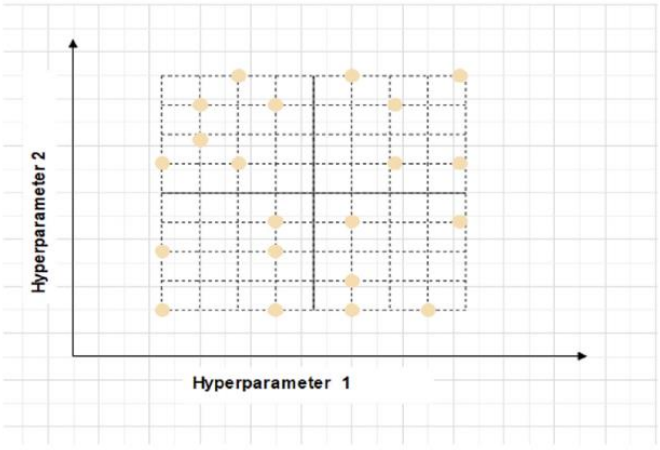

Fig. 2. Visualization of grid search for hyperparameter tuning, demonstrating a systematic approach to exploring different parameter values in machine learning [17].

In this study, hyperparameter tuning was performed for each boosting algorithm: Gradient Boosting, XGBoost, and LightGBM. The tuning process aimed to optimize model performance by exploring various combinations of hyperparameters using GridSearchCV with 3-fold cross-validation and ROC AUC as the evaluation metric. This method ensures that the selected parameters are not overfitted to a particular fold and provide more generalizable results.

Additionally, hyperparameter tuning was utilized as a strategy to prevent overfitting by carefully adjusting parameters that control model complexity and introduce regularization [45]. For example, reducing tree depth, applying subsampling, and setting appropriate learning rates have been shown to improve model generalization to unseen data [46].

In this study, the hyperparameters selected for tuning were chosen based on their substantial influence on model complexity, learning capacity, and regularization capability. For instance:

- **n_estimators** controls the number of boosting stages and affects both accuracy and training time;
- **learning_rate** adjusts the contribution of each tree, where smaller values often improve generalization when paired with more estimators;
- **max_depth** (for Gradient Boosting and XGBoost) and **num_leaves** (for LightGBM) control the complexity of the trees;
- **subsample** and **colsample_bytree** to prevent overfitting by introducing randomness during training.

These parameter functions are consistent with those discussed in [45], [46], who explain the critical role of hyperparameter configuration in optimizing model performance and preventing overfitting in ensemble learning algorithms. The complete list of hyperparameters explored for each model can be seen in Table 8.

Table 8. List of hyperparameters and their tested values for each boosting algorithm.

Algorithm	Hyperparameter	Value Tested
Gradient Boosting	n_estimators	[100, 200, 300]
	learning_rate	[0.01, 0.05, 0.1]
	max_depth	[3, 5, 7]
	subsample	[0.8, 1.0]
XGBoost	n_estimators	[100, 200, 300]
	learning_rate	[0.01, 0.05, 0.1]
	max_depth	[3, 5, 7]
	subsample	[0.8, 1.0]
	colsample_bytree	[0.8, 1.0]
LightGBM	n_estimators	[100, 200, 300]
	learning_rate	[0.01, 0.05, 0.1]
	num_leaves	[31, 50, 70]
	subsample	[0.8, 1.0]
	colsample_bytree	[0.8, 1.0]

G. Model Performance Evaluation

The performance evaluation of the classification model in this study was conducted using the Confusion Matrix and AUC-ROC. The Confusion Matrix is a table that presents information about the model's prediction results, consisting of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) [47]. These components form the Confusion Matrix, as shown in Table 9. The Confusion Matrix is used to assess the effectiveness of the classification model, particularly in the context of binary datasets [48].

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

Table 9. Confusion matrix showing true positive, false positive, false negative, and true negative classifications in model evaluation.

Actual Class	Predicted Class	
	True	**False**
True	True Positive (TP)	False Negative (FN)
False	False Positive (FP)	True Negative (TN)

In this study, the evaluation metrics used include accuracy, precision, recall, and F1-score:

Accuracy: Measures the proportion of correct predictions to the total data, as shown in Eq. (5) [49].

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (5)$$

Precision: Reflects the proportion of correctly predicted positive cases to the total predicted positive cases, calculated using Eq. (6) [50].

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

Recall: Indicates the model's ability to correctly identify all positive cases, as formulated in Eq. (7) [51].

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

F1-Score: Represents the harmonic mean of precision and recall, aiming to balance both in detecting as many positive cases as possible with high precision, as shown in Eq. (8) [52].

$$F1 - Score = \frac{2.Precision.Recall}{Precision + Recall} \quad (8)$$

Additionally, this study utilizes AUC-ROC to evaluate the model's performance across various classification thresholds [53]. The ROC curve is a probability plot that illustrates the relationship between the True Positive Rate (TPR) and the False Positive Rate (FPR), while AUC measures the area under this curve. The AUC value ranges from 0 to 1, with values closer to 1 indicating optimal model performance [53]. The equation for calculating AUC is presented in Eq. (9) [54].

$$AUC = \frac{\left(\frac{TP}{TP + FN}\right) \cdot \left(\frac{TN}{TN + FP}\right)}{2} \quad (9)$$

The interpretation of the AUC value provides insights into the model's ability to distinguish between positive and negative classes and is used to compare the performance of different classification models [55]. Additionally, it helps assess the model's robustness across different datasets and conditions, providing a more comprehensive evaluation of its effectiveness. Table 10 presents the AUC values from various classification experiments conducted in this study [55].

Table 10. Categories of classification performance based on AUC values ranging from 0.50 to 1.00.

AUC Values	Category
0.90 – 1.00	Excellent
0.80 – 0.90	Good
0.70 – 0.80	Fair
0.60 – 0.70	Poor
0.50 – 0.60	Failure

In the context of imbalanced datasets and public health applications such as monkeypox case classification, it is important to evaluate model performance using multiple metrics. Each metric provides a different perspective: while accuracy shows the overall correctness of predictions, it may be misleading when one class dominates. Precision and recall offer insights into how well the model detects actual cases and avoids false alarms. The F1-score provides a balanced measure, especially useful when both false positives and false negatives carry significant implications in health contexts. Meanwhile, the AUC-ROC evaluates the model's overall discriminatory power. For this reason, all these metrics are considered collectively to ensure a comprehensive evaluation of the model's robustness and practical applicability.

3. RESULTS

This section presents the experimental results of the study, including the analysis of Class distribution before and after resampling, hyperparameter tuning results, and model performance evaluation based on accuracy, precision, recall, F1-Score, and AUC-ROC metrics across four scenarios: without resampling, SMOTE, ENN, and SMOTEENN. Learning curve visualizations are also included in the model evaluation section to illustrate performance during the training and validation process.

A. Class Distribution Before and After Resampling

Class imbalance is a common issue in machine learning classification tasks, potentially leading to biased predictions that favor the majority class. In this study, the original dataset exhibits an imbalance, with 6,364 samples in class 0 and 11,136 samples in class 1, as illustrated in Fig. 3.

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

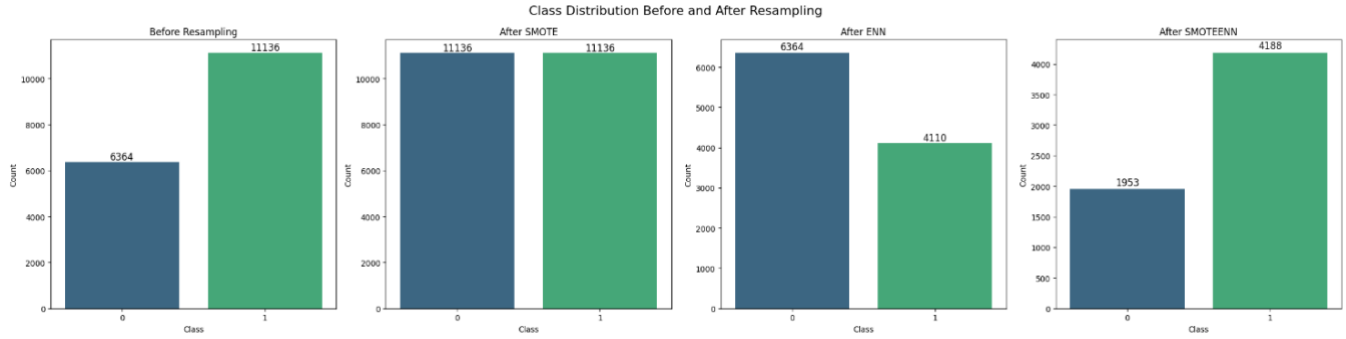

Fig. 3. Comparison of class distribution before resampling and after applying SMOTE, ENN, and SMOTEENN.

To address this issue, three resampling techniques were applied: SMOTE, ENN, and SMOTEENN, as depicted in Fig. 3. SMOTE balances the dataset by generating synthetic samples for the minority class, resulting in 11,136 samples per class. ENN improves data quality by removing noisy and redundant instances, reducing the dataset to 6,364 samples in class 0 and 4,110 in class 1. Among these methods, SMOTEENN proved to be the most effective, as it combines SMOTE's oversampling with ENN's noise filtering. This dual process significantly refines the dataset, yielding 1,953 samples in class 0 and 4,188 in class 1. The substantial reduction reflects SMOTEENN's selectivity in retaining informative data. These varying distributions can significantly influence model performance. SMOTEENN, in particular, helps improve generalization by reducing majority class bias while preserving important patterns from the minority class.

B. Hyperparameter Tuning Results

In Hyperparameter tuning was performed using GridSearchCV with ROC-AUC as the evaluation metric and 3-fold cross-validation. The ROC-AUC metric was chosen because it effectively measures the balance between sensitivity (recall) and specificity in binary classification, providing a more comprehensive evaluation than accuracy, especially for imbalanced datasets. By using ROC-AUC, the model's ability to distinguish between positive and negative classes can be assessed, which is crucial for disease classification tasks such as monkeypox.

The tuning process explored various predefined hyperparameter combinations for each model. The combination that yielded the highest ROC-AUC score was selected as the optimal configuration. The results of this tuning are summarized in Table 11.

Table 11. The best hyperparameter settings for Gradient Boosting, XGBoost, and LightGBM models using SMOTE, ENN, and SMOTEENN resampling methods.

Model	Resampling	n_estimators	learning_rate	max_depth	num_ leaves	sub sample	colsample _bytree
GB	Without resampling	100	0.05	3	-	0.8	-
	SMOTE	100	0.05	3	-	0.8	-
	ENN	100	0.05	3	-	0.8	-
	SMOTEENN	100	0.05	3	-	0.8	-
XGBoost	Without resampling	200	0.05	3	-	0.8	0.8
	SMOTE	200	0.05	3	-	0.8	0.8
	ENN	200	0.05	3	-	0.8	0.8
	SMOTEENN	200	0.05	3	-	0.8	0.8
LightGBM	Without resampling	100	0.01	-	31	0.8	0.8
	SMOTE	100	0.01	-	31	0.8	0.8
	ENN	100	0.01	-	31	0.8	0.8
	SMOTEENN	100	0.01	-	31	0.8	0.8

In Table 11, it can be observed that each model has different optimal hyperparameter configurations. The Gradient Boosting model tends to have a lower learning rate and fewer n_estimators compared to XGBoost, which requires more estimators to achieve its best performance. Meanwhile, LightGBM shows an optimal combination with

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

a higher number of num_leaves compared to the max_depth parameter in the other models, indicating a more complex yet efficient tree structure. Additionally, it can be seen that applying resampling techniques (SMOTE, ENN, and SMOTEENN) does not significantly alter the optimal hyperparameter configurations. This suggests that these boosting methods are relatively stable despite changes in class distribution after data balancing.

C. Model Evaluation

The classification model was evaluated using five key metrics: Accuracy, Precision, Recall, F1-Score, and AUC-ROC. The results of the model evaluation for different resampling techniques are presented in Tables 12–15. Additionally, a learning curve based on the F1-Score is provided in Fig. 4 to illustrate the model's performance across different training scenarios.

Table 12. Performance evaluation results of GB, XGBoost, and LightGBM models without resampling.

Evaluation	GB	XGBoost	LightGBM
Accuracy	0.69	0.69	0.68
Precision	0.68	0.68	0.70
Recall	0.69	0.69	0.68
F1-Score	0.66	0.67	0.61
AUC-ROC	0.70	0.70	0.70

Table 13. Performance evaluation results of GB, XGBoost, and LightGBM models with SMOTE.

Evaluation	GB	XGBoost	LightGBM
Accuracy	0.65	0.65	0.66
Precision	0.66	0.66	0.66
Recall	0.65	0.65	0.66
F1-Score	0.65	0.65	0.66
AUC-ROC	0.70	0.70	0.69

Table 14. Performance evaluation results of GB, XGBoost, and LightGBM models with ENN.

Evaluation	GB	XGBoost	LightGBM
Accuracy	0.56	0.56	0.52
Precision	0.66	0.66	0.66
Recall	0.56	0.56	0.52
F1-Score	0.55	0.56	0.50
AUC-ROC	0.69	0.69	0.69

Table 15. Performance evaluation results of GB, XGBoost, and LightGBM models with SMOTEENN.

Evaluation	GB	XGBoost	LightGBM
Accuracy	0.69	0.69	0.69
Precision	0.68	0.68	0.68
Recall	0.69	0.69	0.69
F1-Score	0.67	0.67	0.65
AUC-ROC	0.69	0.69	0.69

Table 12 presents the model evaluation results before applying any resampling techniques. The Gradient Boosting (GB), XGBoost, and LightGBM models achieved an accuracy of approximately 0.69, with F1-Scores ranging from 0.61 to 0.67. The relatively high AUC-ROC score (~0.70) suggests that the model was still capable of making reasonable predictions despite the class imbalance.

Table 13 displays the evaluation results after applying SMOTE, a technique aimed at balancing the dataset by generating synthetic samples. However, this approach led to a slight decrease in model accuracy to approximately 0.65–0.66, with F1-Scores remaining in a similar range. While class balance improved, this technique did not significantly enhance overall model performance.

Table 14 presents the results after applying Edited Nearest Neighbors (ENN), which focuses on removing outliers from the dataset. This method caused a substantial drop in accuracy to around 0.52–0.56, with F1-Scores decreasing to a range of 0.50–0.56. Although the AUC-ROC score remained relatively high (~0.69), the use of ENN appeared to negatively impact the overall classification performance.

Table 15 provides the results after applying SMOTEENN, a combination of SMOTE and ENN. This technique achieved a better balance between accuracy and class distribution. The models using SMOTEENN maintained an accuracy of approximately 0.69, similar to the models before resampling, but with a more stable F1-Score in the range of 0.65–0.67. Additionally, Precision and Recall were more balanced compared to other resampling methods, making SMOTEENN the most optimal technique in this study.

To further analyze the impact of each resampling method on model training behavior, learning curves were examined to observe how models generalize over different training sizes. Fig. 4 illustrates the learning curves based on the F1-Score for each resampling method. The F1-Score was chosen as the primary evaluation metric due to the class imbalance in the dataset, as it provides a balanced measure between Precision and Recall, making it more representative for model performance assessment, especially in imbalanced classification tasks.

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

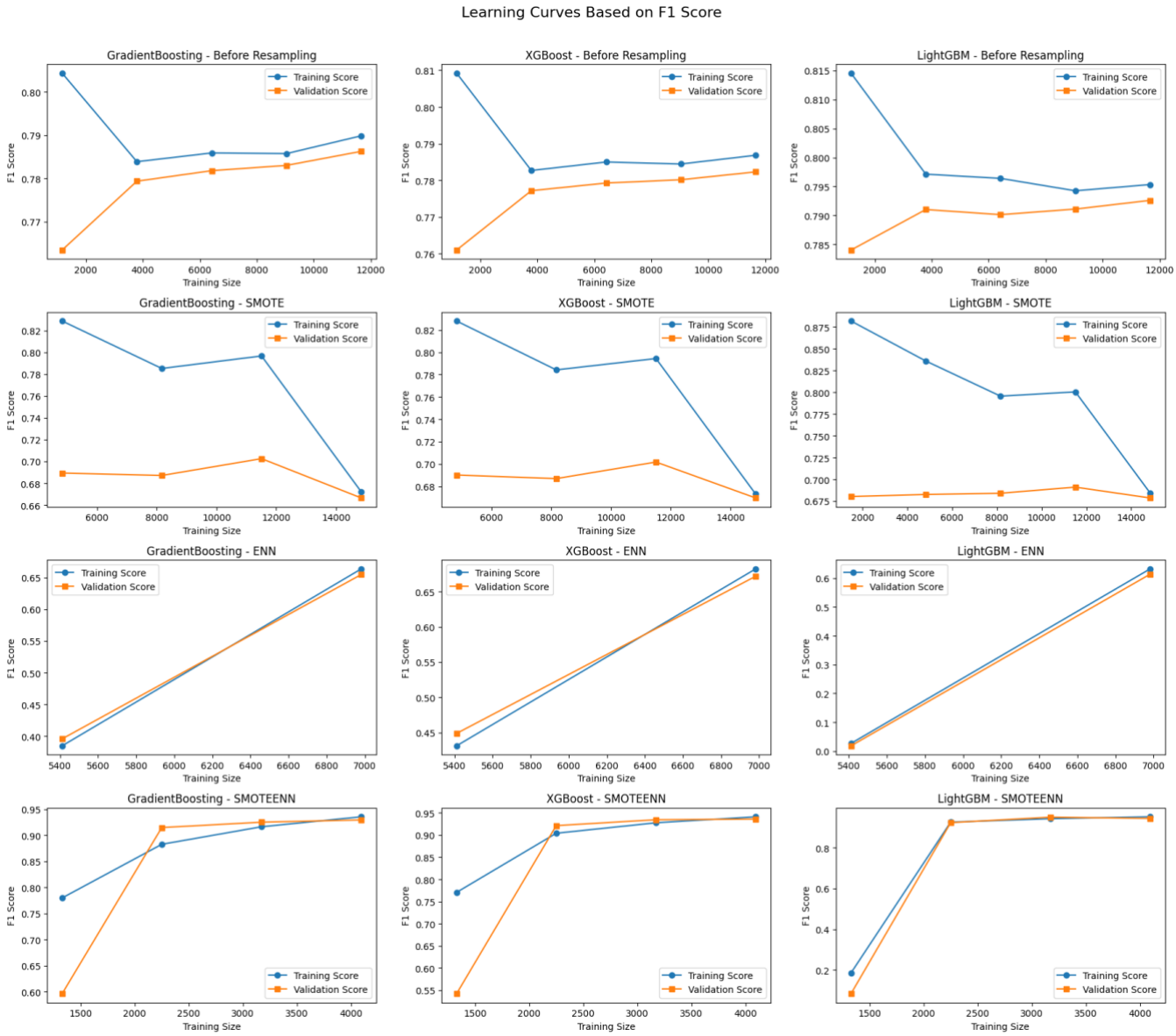

Fig. 4. Learning curve based on F1-Score for GradientBoosting, XGBoost, and LightGBM with different resampling techniques (without resampling, SMOTE, ENN, and SMOTEENN).

From Fig. 4, models without resampling (Before Resampling) show a clear gap between training and validation scores, indicating overfitting. Although training performance is high, the validation scores remain lower and stagnant, reflecting poor generalization. This highlights the importance of applying appropriate resampling methods to reduce bias toward the majority class. The inconsistency between training and validation scores also suggests that the models may be learning noise rather than meaningful patterns in the data.

When SMOTE is applied, the validation score slightly drops despite balanced class distribution, suggesting that synthetic data generation alone does not always improve model generalization. ENN, which removes noisy data, shows closer alignment between training and validation scores, indicating more stable learning. However, the overall performance remains lower, likely due to underfitting caused by excessive data removal. This implies that while noise reduction is beneficial, too much pruning may also eliminate useful data, leading to reduced learning capacity.

SMOTEENN shows the most promising results by providing a balanced increase in both training and validation scores, indicating better generalization. It effectively addresses class imbalance while enhancing data quality. This method offers a strong balance between oversampling and noise reduction, helping models learn more robust patterns.

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

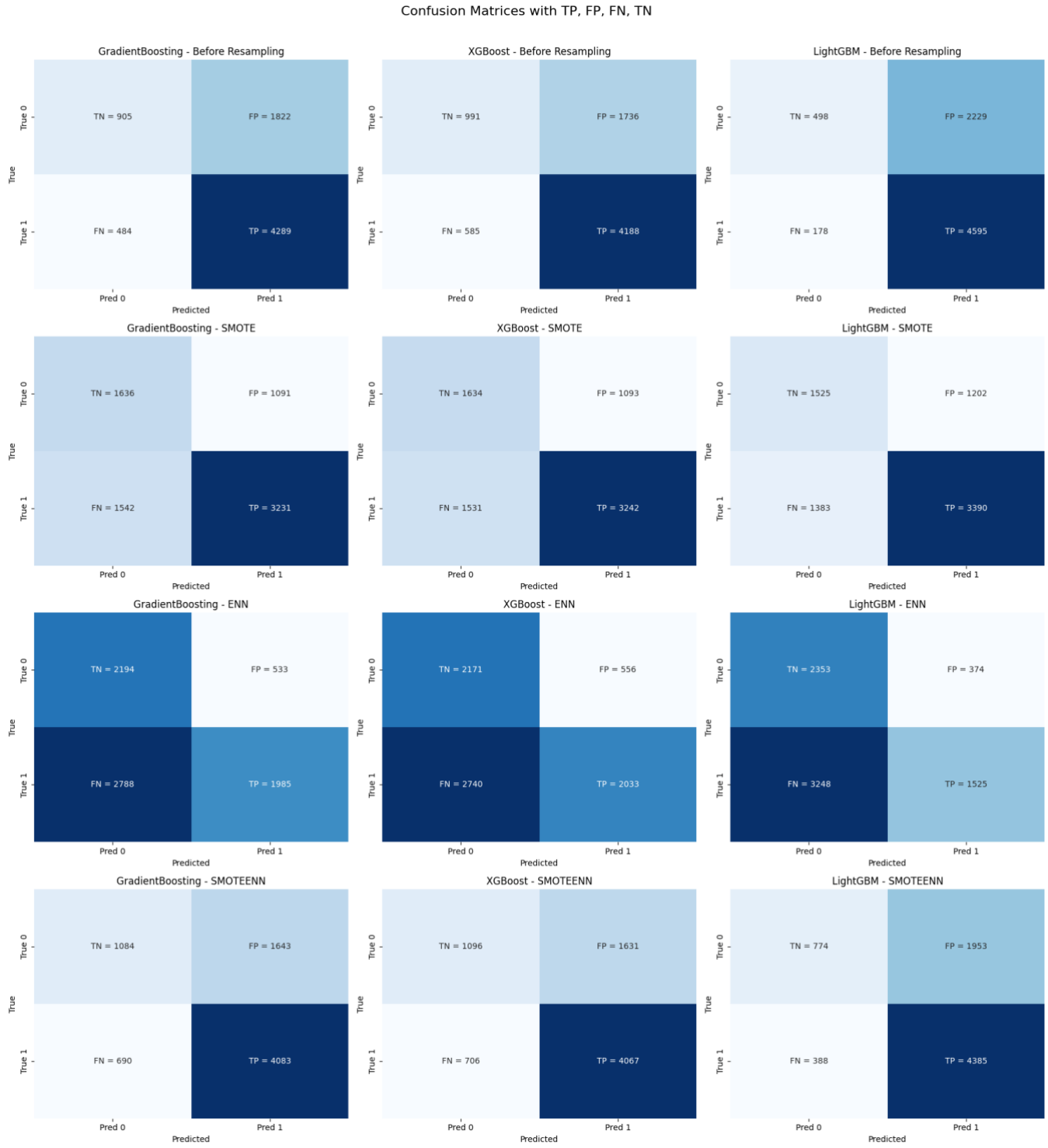

Fig. 5. Confusion matrices of all boosting models before and after applying resampling techniques

To gain a deeper understanding of model performance beyond accuracy and learning curves, confusion matrices were analyzed for each boosting algorithm both before and after the application of resampling techniques, as shown in Fig. 5. These matrices illustrate the distribution of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), providing valuable insights into types of prediction errors made by the models. False positives (FP) refer to cases that are actually negative but incorrectly predicted as positive by the model. Such errors can lead to unnecessary concern or intervention. More critically, false negatives (FN) represent actual positive cases that the model failed to detect, which can be dangerous in health-related domains like monkeypox prediction, as it may result in the disease spreading undetected.

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

Before resampling, all models tended to produce relatively high false negative values, indicating a significant number of missed monkeypox cases. After applying SMOTE, there was some improvement, with better balance between FP and FN, although FN remained considerable. When using ENN, most models became more conservative, resulting in higher FN but fewer FP. SMOTEENN appeared to provide the most balanced outcome, especially in the XGBoost and LightGBM models, which showed more reliable predictions across both classes. These findings emphasize the importance of addressing class imbalance in order to reduce false negatives and improve overall classification reliability, as illustrated in Fig. 5.

4. DISCUSSION

The results of this study indicate that the use of the SMOTEENN method has a significant impact on improving the classification performance of monkeypox cases compared to other resampling methods, such as SMOTE and ENN. From the model evaluation table, it is evident that the implementation of SMOTEENN achieved the highest accuracy of 69% for the Gradient Boosting, XGBoost, and LightGBM algorithms. Additionally, the F1-Score of 67% demonstrates a balance between precision (68%) and recall (69%), meaning that the model can classify cases effectively without compromising either of the key metrics.

In comparison with previous studies, the model developed in this study exhibits better improvements. The study conducted by Cindy et al. [3] using SVM with ROS and RUS techniques only achieved an accuracy of 36.5% and an AUC-ROC of 0.40. Meanwhile, the study by Arvita et al. [56], which used KNN, obtained an accuracy of 59.5%, recall of 64.78%, and precision of 58.68%. Additionally, the study by Anugrah et al. [4], which also used SVM, achieved an accuracy of 65%, with precision, recall, and F1-score of 65% each. Thus, this study successfully improved classification performance by up to 4% compared to previous studies by employing a more optimal approach using boosting algorithms and the SMOTEENN resampling technique. The comparison of this study's results with previous research can be seen in Table 16.

Table 16. Comparison of accuracy, precision, recall, F1-score, and ROC-AUC with previous research.

Research	Model and Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC
Cindy et al. [3]	SVM, ROS, RUS	36.50	-	-	-	0.40
Arvita et al. [56]	KNN	59.50	58.68	64.78	-	-
Anugrah et al. [4]	SVM	65.00	65.00	65.00	65.00	-
This Research	GradientBoosting, XGBoost, LightGBM, SMOTE, ENN, SMOTEENN	69.00	68.00	69.00	67.00	0.69

Although this method shows promising results, this study has several limitations. One of them is the higher computational time compared to other resampling methods. This is due to the combination of oversampling (SMOTE) and undersampling (ENN), which increases complexity during the model training process. Additionally, despite the improvement in accuracy, the AUC-ROC value remains at 0.69. This could be attributed to the fact that while SMOTEENN effectively balances class distribution, the undersampling process in ENN may remove informative samples, particularly from the minority class, which could limit the model's ability to distinguish between positive and negative cases. Furthermore, the low correlation of certain features with the target variable might also contribute to this limitation, making it difficult for the model to improve its discriminatory power. Another key limitation is the use of a synthetic dataset obtained from Kaggle. While such datasets are useful for prototyping and experimentation, they may not fully capture the complexity and variability of real-world monkeypox cases. As a result, there is a risk of bias or reduced generalizability when the model is applied to real clinical or epidemiological data.

Nonetheless, the implications of this study suggest that SMOTEENN can be an effective alternative for handling data imbalance in monkeypox cases, especially when combined with boosting algorithms such as Gradient Boosting, XGBoost, and LightGBM. Given the nature of monkeypox as an emerging infectious disease with uneven reporting and case distribution, the ability to balance datasets is crucial for developing accurate prediction models. Moreover, the approach used in this study could also be applied to other infectious diseases that suffer from class imbalance problems, such as dengue, tuberculosis, or COVID-19, where early detection and proper classification play a vital role in containment strategies.

In the context of public health and epidemiology, accurate classification of infectious disease cases supports better surveillance, early intervention, and targeted resource allocation. Thus, the findings of this study are aligned with current trends in public health informatics and disease outbreak management, where machine learning-based decision support systems are increasingly being used. Incorporating techniques like SMOTEENN into such systems can help improve

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

reliability in predictions, especially when real-world data is limited, sparse, or imbalanced.

5. CONCLUSION

This study evaluates the impact of the SMOTEENN method on the classification performance of monkeypox cases using the Gradient Boosting, XGBoost, and LightGBM algorithms. The results indicate that SMOTEENN can improve model accuracy up to 69%, with a precision of 68%, recall of 69%, and an F1-score of 67%. Compared to previous studies, this approach provides a significant accuracy improvement, particularly in handling data imbalance , which is a common challenge in epidemiological datasets of emerging diseases.

The real-world implication of these findings lies in their potential contribution to the early detection and classification of monkeypox cases, which can enhance outbreak monitoring, clinical decision-making, and public health response strategies. Although the study utilizes synthetic data, the proposed approach offers a viable starting point for further research involving real-world epidemiological datasets.

Future work should explore the application of SMOTEENN in other infectious diseases with similar class imbalance problems, and further validate the model's effectiveness using real data from healthcare providers or government surveillance systems. Moreover, addressing computational challenges, implementing feature selection strategies, and experimenting with deep learning architectures could further enhance prediction accuracy and efficiency.

REFERENCES

[1] K. Soni and A. K. Sinha, "Modeling and stability analysis of the transmission dynamics of Monkeypox with control intervention," Partial Differ. Equations Appl. Math., vol. 10, p. 100730, 2024, doi: https://doi.org/10.1016/j.padiff.2024.100730.

[2] Z. Khan, A. Ali, and S. Aldahmani, "Feature Selection via Robust Weighted Score for High Dimensional Binary Class-Imbalanced Gene Expression Data," Heliyon, vol. 10, p. e38547, 2024, doi: https://doi.org/10.1016/j.heliyon.2024.e38547.

[3] Cindy, T. Sabatini, and V. Itan, "Implementation of Support Vector Machine for Monkeypox Case Classification: An Oversampling and Undersampling Approach to Address Class Imbalance," J. Digit. Ecosyst. Nat. Sustain., vol. 4, no. 1, pp. 38–43, 2024, [Online]. Available: https://journal.uvers.ac.id/index.php/jodens/article/view/234

[4] W. Anugrah, E. Haerani, Yusra, and L. Oktavia, "Classification of Monkeypox Disease Using the Support Vector Machine Method," J. Comput. Syst. Informatics, vol. 5, no. 3, pp. 558–566, 2024, doi: https://doi.org/10.47065/josyc.v5i3.5149.

[5] D. B. Vukovi, V. Spitsin, A. Bragin, V. Leonova, and L. Spitsina, "Forecasting firm growth resumption post-stagnation," J. Open Innov. Technol. Mark. Complex., vol. 10, p. 100406, 2024, doi: https://doi.org/10.1016/j.joitmc.2024.100406.

[6] X. Feng, Y. Cai, and R. Xin, "Optimizing diabetes classification with a machine learning-based framework," BMC Bioinformatics, vol. 24, p. 428, 2023, doi: 10.1186/s12859-023-05467-x.

[7] A. Sutradhar et al., "Advancing thyroid care: An accurate trustworthy diagnostics system with interpretable AI and hybrid machine learning techniques," Heliyon, vol. 10, no. 17, p. e36556, 2024, doi: https://doi.org/10.1016/j.heliyon.2024.e36556.

[8] Y. B. Wah et al., "Machine Learning and Synthetic Minority Oversampling Techniques for Imbalanced Data: Improving Machine Failure Prediction," Comput. Mater. Contin., vol. 75, no. 3, pp. 4821–4841, 2023, doi: https://doi.org/10.32604/cmc.2023.034470.

[9] K. Afane and Y. Zhao, "Selecting Classifiers and Resampling Techniques for Imbalanced Datasets: A New Perspective," Procedia Comput. Sci., vol. 246, pp. 1150–1159, 2024, doi: https://doi.org/10.1016/j.procs.2024.09.539.

[10] M. Lin, X. Zhu, T. Hua, X. Tang, G. Tu, and X. Chen, "Detection of ionospheric scintillation based on xgboost model improved by smote-enn technique," Remote Sens., vol. 13, pp. 1–22, 2021, doi: https://doi.org/10.3390/rs13132577.

[11] Q. Gao et al., "Identification of Orphan Genes in Unbalanced Datasets Based on Ensemble Learning," Front. Genet., vol. 11, p. 820, 2020, doi: https://doi.org/10.3389/fgene.2020.00820.

[12] J. Lim, H. G. Jeon, Y. Seo, M. Kim, J. U. Moon, and S. H. Cho, "Survival Prediction Model for Patients with Hepatocellular Carcinoma and Extrahepatic Metastasis Based on XGBoost Algorithm," J. Hepatocell. Carcinoma, vol. 10, pp. 2251–2263, 2023, doi: https://doi.org/10.2147/jhc.s429903.

[13] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi, and M. Asadpour, "Boosting methods for multi-class imbalanced data classification: an experimental review," J. Big Data, vol. 7, no. 70, pp. 1–47, 2020, doi: 10.1186/s40537-020-00349-y.

[14] J. Montomoli et al., "Machine learning using the extreme gradient boosting (XGBoost) algorithm predicts 5-day delta of SOFA score at ICU admission in COVID-19 patients," 2021, Elsevier B.V. doi: https://doi.org/10.1016/j.jointm.2021.09.002.

[15] C. Daniel, "A robust LightGBM model for concrete tensile strength forecast to aid in resilience-based structure strategies," Heliyon, vol. 10, no. 20, p. e39679, 2024, doi: https://doi.org/10.1016/j.heliyon.2024.e39679.

[16] M. Ogunsanya, J. Isichei, and S. Desai, "Grid search hyperparameter tuning in additive

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

manufacturing processes," Manuf. Lett., vol. 35, pp. 1031–1042, 2023, doi: https://doi.org/10.1016/j.mfglet.2023.08.056.

[17] V. Ghate and S. Hemalatha C, "A comprehensive comparison of machine learning approaches with hyper-parameter tuning for smartphone sensor-based human activity recognition," Meas. Sensors, vol. 30, p. 100925, 2023, doi: https://doi.org/10.1016/j.measen.2023.100925.

[18] M. Ahmed, "Monkey-Pox PATIENTS Dataset," Kaggle. Accessed: Sep. 05, 2024. [Online]. Available: https://www.kaggle.com/datasets/muhammad4hmed/monkeypox-patients-dataset

[19] R. Safdari, A. Deghatipour, M. Gholamzadeh, and K. Maghooli, "Applying data mining techniques to classify patients with suspected hepatitis C virus infection," Intell. Med., vol. 2, no. 4, pp. 193–198, 2022, doi: https://doi.org/10.1016/j.imed.2021.12.003.

[20] S. García, S. Ramírez-Gallego, J. Luengo, J. M. Benítez, and F. Herrera, "Big data preprocessing: methods and prospects," Big Data Anal., vol. 1, no. 9, pp. 1–23, 2016, doi: 10.1186/s41044-016-0014-0.

[21] S. Developers, "sklearn.impute.IterativeImputer," scikit-learn. Accessed: Mar. 14, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.impute.IterativeImputer.html

[22] J. M. A. S. Dachi and P. Sitompul, "Comparison Analysis of the XGBoost Algorithm and Random Forest Ensemble Learning Algorithm in Credit Decision Classification," J. Ris. Rumpun Mat. Dan Ilmu Pengetah. Alam, vol. 2, no. 2, pp. 87–103, 2023, doi: https://doi.org/10.55606/jurrimipa.v2i2.1470.

[23] R. C R and D. S. C P, "Evaluating Deep Learning with different feature scaling techniques for EEG-based Music Entrainment Brain Computer Interface," e-Prime - Adv. Electr. Eng. Electron. Energy, vol. 7, p. 100448, 2024, doi: https://doi.org/10.1016/j.prime.2024.100448.

[24] V. R. Prasetyo, M. Mercifia, A. Averina, L. Sunyoto, and B. Budiarjo, "FILM RATING PREDICTION ON IMDB WEBSITE USING NEURAL NETWORK," Netw. Eng. Res. Oper., vol. 7, no. 1, pp. 1–8, 2022, doi: 10.21107/nero.v7i1.268.

[25] S.-L. Developers, "sklearn.model_selection.train_test_split," Scikit-Learn. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[26] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation," Expert Syst. Appl., vol. 244, p. 122778, 2024, doi: https://doi.org/10.1016/j.eswa.2023.122778.

[27] H. R. Sneha and B. Annappa, "Exploratory Analysis of Methods, Techniques, and Metrics to Handle Class Imbalance Problem," Procedia Comput. Sci., vol. 235, pp. 863–877, 2024, doi: https://doi.org/10.1016/j.procs.2024.04.082.

[28] Y. Zhu, C. Jia, F. Li, and J. Song, "Inspector: a lysine succinylation predictor based on edited nearest-neighbor undersampling and adaptive synthetic oversampling," Anal. Biochem., vol. 593, p. 113592, 2020, doi: https://doi.org/10.1016/j.ab.2020.113592.

[29] J. Beinecke and D. Heider, "Gaussian noise up-sampling is better suited than SMOTE and ADASYN for clinical decision making," BioData Min., vol. 14, no. 49, pp. 1–11, 2021, doi: https://doi.org/10.1186/s13040-021-00283-6.

[30] X. Li and Q. Zhou, "Research on improving SMOTE algorithms for unbalanced data set classification," Proc. - 2019 Int. Conf. Electron. Eng. Informatics, EEI 2019, pp. 476–480, 2019, doi: https://doi.org/10.1109/EEI48997.2019.00109.

[31] A. Chahal et al., "Predictive analytics technique based on hybrid sampling to manage unbalanced data in smart cities," Heliyon, vol. 10, p. e39275, 2024, doi: https://doi.org/10.1016/j.heliyon.2024.e39275.

[32] V. Kumar et al., "Addressing Binary Classification over Class Imbalanced Clinical Datasets Using Computationally Intelligent Techniques," Healthc., vol. 10, no. 7, pp. 1–28, 2022, doi: https://doi.org/10.3390/healthcare10071293.

[33] X. Shu and Y. Ye, "Knowledge Discovery: Methods from data mining and machine learning," Soc. Sci. Res., vol. 110, p. 102817, 2023, doi: https://doi.org/10.1016/j.ssresearch.2022.102817.

[34] I. Veza, A. Deniz Karaoglan, S. Akpinar, M. Spraggon, and M. Idris, "Machine learning of weighted superposition attraction algorithm for optimization diesel engine performance and emission fueled with butanol-diesel biofuel," Ain Shams Eng. J., p. 103126, 2024, doi: https://doi.org/10.1016/j.asej.2024.103126.

[35] B. Ouadi, A. Khatir, E. Magagnini, M. Mokadem, L. Abualigah, and A. Smerat, "Optimizing silt density index prediction in water treatment systems using pressure-based gradient boosting hybridized with Salp Swarm Algorithm," J. Water Process Eng., vol. 68, p. 106479, 2024, doi: https://doi.org/10.1016/j.jwpe.2024.106479.

[36] Z. Wu et al., "Integration of geographic features and bathymetric inversion in the Yangtze River's Nantong Channel using gradient boosting machine algorithm with ZY-1E satellite and multibeam data," Geomatica, vol. 76, p. 100027, 2024, doi: https://doi.org/10.1016/j.geomat.2024.100027.

[37] H. Zulfiqar et al., "Identification of cyclin protein

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

using gradient boost decision tree algorithm," Comput. Struct. Biotechnol. J., vol. 19, pp. 4123–4131, 2021, doi: https://doi.org/10.1016/j.csbj.2021.07.013.

[38] V. Safavi, A. Mohammadi Vaniar, N. Bazmohammadi, J. C. Vasquez, O. Keysan, and J. M. Guerrero, "Early prediction of battery remaining useful life using CNN-XGBoost model and Coati optimization algorithm," J. Energy Storage, vol. 98, p. 113176, 2024, doi: https://doi.org/10.1016/j.apenergy.2023.122048.

[39] J. Liu, F. Liu, and L. Wang, "Automated, economical, and environmentally-friendly asphalt mix design based on machine learning and multi-objective grey wolf optimization," J. Traffic Transp. Eng. (English Ed., vol. 11, no. 3, pp. 381–405, 2024, doi: https://doi.org/10.1016/j.jtte.2023.10.002.

[40] J. Brownle, "What is the XGBoost Algorithm," XGBoosting. Accessed: Feb. 19, 2025. [Online]. Available: https://xgboosting.com/what-is-the-xgboost-algorithm/

[41] X. Mao et al., "A variable weight combination prediction model for climate in a greenhouse based on BiGRU-Attention and LightGBM," Comput. Electron. Agric., vol. 219, p. 108818, 2024, doi: https://doi.org/10.1016/j.compag.2024.108818.

[42] Y. Wang and T. Wang, "Application of improved LightGBM model in blood glucose prediction," Appl. Sci., vol. 10, no. 9, pp. 1–16, 2020, doi: https://doi.org/10.3390/app10093227.

[43] M. Seyyedattar, S. Zendehboudi, A. Ghamartale, and M. Afshar, "Advancing hydrogen storage predictions in metal-organic frameworks: A comparative study of LightGBM and random forest models with data enhancement," Int. J. Hydrogen Energy, vol. 69, pp. 158–172, 2024, doi: https://doi.org/10.1016/j.ijhydene.2024.04.230.

[44] Z. Pan, S. Fang, and H. Wang, "LightGBM Technique and Differential Evolution Algorithm-Based Multi-Objective Optimization Design of DS-APMM," IEEE Trans. Energy Convers., vol. 36, no. 1, pp. 441–455, 2021, doi: https://doi.org/10.1109/TEC.2020.3009480.

[45] B. Bischl et al., "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," Wiley Interdiscip. Rev. Data Min. Knowl. Discov., vol. 13, no. 2, 2023, doi: https://doi.org/10.1002/widm.1484.

[46] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," Artif. Intell. Rev., vol. 54, no. 3, pp. 1937–1967, 2021, doi: https://doi.org/10.1007/s10462-020-09896-5.

[47] M. N. J. R and P. R, "Performance Analysis of Text Classification Algorithms using Confusion Matrix," Int. J. Eng. Tech. Res., vol. 6, no. 4, pp. 75–78, 2016.

[48] Muhamad Fawwaz Akbar, Muhammad Itqan Mazdadi, Muliadi, Triando Hamonangan Saragih, and Friska Abadi, "Implementation of Information Gain Ratio and Particle Swarm Optimization in the Sentiment Analysis Classification of Covid-19 Vaccine Using Support Vector Machine," J. Electron. Electromed. Eng. Med. Informatics, vol. 5, no. 4, pp. 261–270, 2023, doi: http://dx.doi.org/10.35882/jeeemi.v5i4.328.

[49] M. Mahmud et al., "Implementation of C5.0 Algorithm using Chi-Square Feature Selection for Early Detection of Hepatitis C Disease," J. Electron. Electromed. Eng. Med. Informatics, vol. 6, no. 2, pp. 116–124, 2024, doi: https://doi.org/10.35882/jeeemi.v6i2.384.

[50] A. Tajali, T. H. Saragih, M. I. Mazdadi, I. Budiman, and A. Farmadi, "The Impactness of SMOTE as Imbalance Class Handling for Myocardial Infarction Complication Classification using Machine Learning Approach with Data Imputation and Hyperparameter," Indones. J. Electron. Electromed. Eng. Med. Informatics, vol. 6, no. 4, pp. 227–239, 2024, doi: https://doi.org/10.35882/ijeeemi.v6i4.13.

[51] Y. F. Zamzam, T. H. Saragih, R. Herteno, Muliadi, D. T. Nugrahadi, and P. H. Huynh, "Comparison of CatBoost and Random Forest Methods for Lung Cancer Classification using Hyperparameter Tuning Bayesian Optimization-based," J. Electron. Electromed. Eng. Med. Informatics, vol. 6, no. 2, pp. 125–136, 2024, doi: https://doi.org/10.35882/jeeemi.v6i2.382.

[52] M. Shirdel, M. Di Mauro, and A. Liotta, "Worthiness Benchmark: A novel concept for analyzing binary classification evaluation metrics," Inf. Sci. (Ny)., vol. 678, p. 120882, 2024, doi: https://doi.org/10.1016/j.ins.2024.120882.

[53] M. Krzywicka and A. Wosiak, "Sensitivity of Standard Evaluation Metrics for Disease Classification and Progression Assessment Based on Whole-Body Imaging," Procedia Comput. Sci., vol. 225, pp. 4314–4323, 2023, doi: https://doi.org/10.1016/j.procs.2023.10.428.

[54] D. Fitria, T. H. Saragih, Muliadi, D. Kartini, and F. Indriani, "Classification of Appendicitis in Children Using SVM with KNN Imputation and SMOTE Approach to Improve Prediction Quality," J. Electron. Electromed. Eng. Med. Informatics, vol. 6, no. 3, pp. 302–311, 2024, doi: https://doi.org/10.35882/jeeemi.v6i3.470.

[55] A. M. Akbar, R. Herteno, S. W. Saputro, M. R. Faisal, and R. A. Nugroho, "Optimizing Software Defect Prediction Models: Integrating Hybrid Grey Wolf and Particle Swarm Optimization for Enhanced Feature Selection with Popular Gradient Boosting Algorithm," J. Electron. Electromed. Eng. Med. Informatics, vol. 6, no. 2, pp. 169–181, 2024, doi: https://doi.org/10.35882/jeeemi.v6i2.388.

[56] Y. Arvita, Suyanti, and A. Siswanto, "Classification

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

of Monkeypox Disease Using the K-Nearest Neighbor Algorithm," Process. J. Ilm. Sist. Informasi, Teknol. Inf. dan Sist. Komput., vol. 19, no. 2, pp. 137–142, 2024, doi: https://doi.org/10.33998/processor.2024.19.2.1616 .

AUTHOR BIOGRAPHY

Laifansan Siena is a Computer Science student at Lambung Mangkurat University, South Kalimantan, where she has been studying since 2021. Her research interests focus on Data Science, particularly in machine learning for disease classification. Currently, she is working on her final project, which involves the classification of monkeypox cases using boosting algorithms and resampling techniques to improve prediction accuracy. Through this research, she aims to explore various machine learning models and enhance their effectiveness in handling imbalanced datasets. She is passionate about applying data science and artificial intelligence in the healthcare sector to develop more accurate predictive models. In the future, she hopes to contribute to technological advancements by integrating AI solutions into real-world applications, especially in medical diagnosis and epidemiology. She can be contacted at laifansansiena89@gmail.com.

Triando Hamonangan Saragih, currently holding the position of a lecturer within the Department of Computer Science at Lambung Mangkurat University, is heavily immersed in the realm of academia, with a profound focus on the multifaceted domain of Data Science. His academic pursuits commenced with the successful completion of his bachelor's degree in Informatics at the esteemed Brawijaya University, located in the vibrant city of Malang, back in the year 2016. Building upon this foundational achievement, he proceeded to further enhance his scholarly credentials by enrolling in a master's program in Computer Science at Brawijaya University, Malang, culminating in the conferral of his advanced degree in 2018. The research field he is involved in is Data Science. Email: triando.saragih@ulm.ac.id. Orcid ID: 0000-0003-4346-3323.

Radityo Adi Nugroho received his bachelor's degree in Informatics from the Islamic University of Indonesia in 2004 and a master's degree in Computer Science from Gadjah Mada University in 2007. Currently, he is an assistant professor in the Department of Computer Science at Lambung Mangkurat University. In addition, he served as the Head of the Center for Information and Communication Technology at ULM from 2014 to 2023. He is actively involved in research and development, focusing on software defect prediction and computer vision to enhance system reliability and optimize visual data processing across various applications. He can be contacted at email: radityo.adi@ulm.ac.id.

Dwi Kartini earned her Bachelor's and Master's degrees in Computer Science from the Faculty of Computer Science at Putra Indonesia "YPTK" in Padang, Indonesia. She is also a lecturer in the Department of Computer Science, where she teaches a variety of subjects including linear algebra, discrete mathematics, research methods, and others. Her research interests focus on the applications of Artificial Intelligence and Data Mining. Currently, she serves as an assistant professor in the Department of Computer Science, Faculty of Mathematics and Natural Sciences at Lambung Mangkurat University in Banjarbaru, Indonesia, and is the head of the Computer Science study program. She can be contacted at dwikartini@ulm.ac.id.

Muliadi is a lecturer in the Department of Computer Science at Lambung Mangkurat University, where he specializes in Artificial Intelligence, Decision Support Systems, and Data Science. His expertise aims to develop intelligent solutions for decision making and optimize data analysis for various industrial and academic needs. His academic journey began with a bachelor's degree in Informatics Engineering from STMIK Akakom in 2004 and a master's degree in Computer Science from Gadjah Mada University in 2009. With expertise in Data Science, he also brings valuable skills in Start-up Business Development, Digital Entrepreneurship, and Data Management Staff, supporting technological innovation and efficient data management in both business and academic fields. He can be contacted at email: muliadi@ulm.ac.id.

Dr. Wahyu Caesarendra received a Bachelor of Engineering (Mechanical) degree from Diponegoro University, Indonesia in 2005. He received New University for Regional Innovation (NURI) and Brain Korea 21 (BK21) scholarships for Master study in 2008 and obtained his Master of Engineering (Mechanical) degree from Pukyong National University, South Korea in 2010. In 2011, Wahyu Caesarendra was awarded of University Postgraduate Award (UPA) and International Postgraduate Tuition Award (IPTA) from the University of Wollongong. He received a Doctor of Philosophy (Mechanical) degree from the University of Wollongong in 2015. He worked as Postdoctoral

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77

Research Fellow in Rolls-Royce@NTU Corp Lab, School of Mechanical and Aerospace Engineering, Nanyang Technological University (NTU), Singapore from February 2017 to September 2018. He was a Visiting Assistant Professor at the National Taiwan University of Science and Technology (NTUST) from August 5-11, 2019. He was also a visiting academic at Faculty of Mechanical Engineering, Politechnika Opolska from July to December 2023 and a part-time Associate Professor at Faculty of Mechanical Engineering, Opole University of Technology, Poland in October – December 2024.

Corresponding author: Triando Hamonangan Saragih, triando.saragih@ulm.ac.id, Department of Computer Science, Lambung Mangkurat University, Jl. A. Yani Km 36, Banjarbaru 70714, Kalimantan Selatan, Indonesia.
DOI: https://doi.org/10.35882/ijeeemi.v7i2.77