Jurnal Teknik Informatika (JUTIF) Vol. 6, No. 4, August 2025, Page. 2467-2483
P-ISSN: 2723-3863 https://jutif.if.unsoed.ac.id
E-ISSN: 2723-3871 DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4965

A Comparative Analysis of Hyperparameter-Tuned XGBoost and LightGBM for Multiclass Rainfall Classification in Jakarta

Cokorda Gde Lanang Pringandana *[1], Kusnawi [2]

[1,2]Informatics, Faculty of Computer Science, Universitas Amikom Yogyakarta, Indonesia

Email: [1]coklanang@students.amikom.ac.id

Received : Jun 27, 2025; Revised : Jul 22, 2025; Accepted : Aug 16, 2025; Published : Aug 24, 2025

Abstract

The increasing frequency of extreme weather events in Jakarta has disrupted daily life and critical infrastructure, highlighting the urgent need for accurate rainfall prediction models to support disaster mitigation and early warning systems. This study aims to evaluate and compare the performance of two machine learning algorithms Extreme Gradient Boosting (XGBoost) and Light Gradient Boosting Machine (LightGBM) for multiclass rainfall classification using historical meteorological data. The dataset, which includes features such as temperature, humidity, wind speed, and rainfall, was preprocessed through mean imputation, oversampling to address class imbalance, one-hot encoding, and feature engineering. Both models were trained and tuned using RandomizedSearchCV and assessed through cross-validation and independent testing. The results show that XGBoost consistently outperformed LightGBM, achieving 94% accuracy compared to 91%. Furthermore, XGBoost demonstrated higher precision, recall, F1-score, and specificity across all rainfall categories, resulting in fewer misclassifications and more stable predictions. Confusion matrices confirmed its superior ability to distinguish between similar weather conditions such as cloudy and rainy classes. These findings indicate that XGBoost is more effective in capturing nonlinear interactions between weather features and is therefore better suited for use in complex tropical climates. The study concludes that XGBoost is the more reliable model and recommends its integration into real-time early warning systems to improve climate resilience and disaster preparedness in urban areas like Jakarta that are increasingly affected by climate variability.

Keywords : *Classification, Hyperparameter Tuning, LightGBM, Machine Learning, Rainfall Prediction, XGBoost*

1. INTRODUCTION

Urban centers in Southeast Asia, such as Jakarta, are increasingly vulnerable to extreme weather. With over 10 million residents, Jakarta frequently experiences localized flooding due to short, intense rainfall. Between 2017 and 2021, daily rainfall reached up to 137.70 mm in Central Jakarta and 130.11 mm in the West [1], causing disruptions to infrastructure, mobility, and public safety.

Timely and accurate rainfall prediction is essential for disaster mitigation and urban water management. Machine learning (ML) has been widely used for this purpose, thanks to its ability to capture nonlinear patterns among weather variables. Oversampling techniques can enhance classification accuracy beyond 90% [2]. Adiyasa et al. [3] utilized daily rainfall data from several Jakarta regions (Central, South, East, and North) as key inputs to train ML-based flood detection models for the 2016–2020 period.

Meteorological parameters including solar radiation, temperature, humidity, wind speed, and air pressure are known to influence precipitation patterns [4], [5]. However, the interactions among these variables are often nonlinear and influenced by regional atmospheric dynamics, especially in tropical regions like Jakarta. Conventional models, such as autoregressive and linear regression methods, often struggle to capture these complex relationships [4], [5]. Additionally, while annual rainfall in Jakarta has increased, the frequency of extreme events has declined, suggesting a nonlinear redistribution of rainfall across time [6]. This phenomenon is supported by recent research showing a shift in the distribution of rainfall intensity and seasonal onset in Jakarta over the last three decades, with higher occurrences of intense rainfall events particularly in southern zones [7]. Furthermore, the application of data-driven approaches such as Artificial Neural Networks (ANN) has proven effective in estimating rainfall using climatic variables—including temperature, humidity, solar radiation, and wind speed with high predictive accuracy and robustness against missing data problems [8]. These findings emphasize the importance of adopting flexible, adaptive models in response to evolving climate variability.

Ensemble-based ML models like XGBoost and LightGBM, built on the Gradient Boosting Decision Tree (GBDT) framework, excel in handling high-dimensional data and modeling complex relationships [9]. XGBoost has shown strong performance in climate prediction tasks with proper hyperparameter tuning [10]. It has also been used successfully for flood risk assessment in urban areas using historical meteorological data [11]. Furthermore, Maulita et al.[12] showed that Gradient Boosting Regressor performs competitively in flood probability prediction, with results nearly matching linear regression in terms of MAE and RMSE, reinforcing its applicability in disaster mitigation systems.

LightGBM, a more recent implementation of GBDT, improves training speed and reduces memory usage without sacrificing accuracy. It leverages innovations such as histogram-based learning and exclusive feature bundling, making it suitable for large-scale and near-real-time applications [13]. Several studies have demonstrated its competitive performance compared to XGBoost, particularly in scenarios with limited computational resources or imbalanced data. For instance, Soleha et al. [14] found that LightGBM achieved higher F1-scores in handling imbalanced classification tasks. Likewise, Handayani and Lailiah [15] reported that LightGBM outperformed XGBoost and Extra Trees in terms of accuracy and recall when applied to fetal health classification using SMOTE.

For instance, Wibawa et al. [13] reported strong performance by LightGBM in predicting daily air humidity, with $R^2 = 0.7981$ and RMSE = 0.0786. Yasper et al. [10] applied GridSearchCV for XGBoost hyperparameter tuning in binary rainfall classification and achieved 95% accuracy, highlighting the importance of model optimization. Similarly, Fauziah et al. [16] used XGBoost for monthly rainfall forecasting in Banyuwangi, successfully capturing seasonal trends despite high variability. Syahreza et al. [17] compared XGBoost, SVR, and Random Forest for daily temperature prediction, with XGBoost showing the highest accuracy ($R^2 = 0.8183$).

A notable study by Maharina et al. [18] emphasized the role of SMOTE and interaction-based features in improving flood prediction accuracy. While their study found LightGBM slightly outperformed XGBoost for flood prediction, this research focuses on multiclass rainfall intensity classification and finds the opposite XGBoost performs better. This contrast in problem focus and outcome forms the core novelty of this study.Other studies have explored different approaches, such as explainable machine learning and statistical pattern analysis to identify key climatic factors and detect anomalies in extreme rainfall conditions [19], [20]. In addition, Hermansyah et al.[21] applied XGBoost and LightGBM to predict flood events in Jakarta using weather data from 2016–2020. Their method involved data preprocessing, feature engineering, and SMOTE to address class imbalance. LightGBM slightly outperformed XGBoost, achieving 96.81% accuracy and an AUC of 0.9957. Interaction features like *RR_RH_interaction* and *Tx_ss_interaction* further enhanced model performance, with ten-fold cross-validation confirming its robustness.

Beyond rainfall modeling, XGBoost has also been widely applied in various environmental forecasting tasks. Sangaji and Sutabri [22] demonstrated that XGBoost provided the fastest training and prediction times in forest fire risk modeling, making it suitable for real-time decision-support systems. Kahfi et al. [23] developed a rainfall classification model by integrating sea surface indicators such as SST and IOD into a hybrid neural network with XGBoost, significantly improving classification accuracy. Alamsyah et al. [24] applied hyperparameter tuning using RandomizedSearchCV to enhance XGBoost performance in predicting forest fire fuel dryness, achieving an R^2 of 0.9820 and an MSE of 0.0210. Employed XGBoost to classify Air Pollution Index (ISPU) levels based on meteorological and emission data, supporting the development of early warning systems for urban air quality management. These four studies reaffirm the strong potential of GBDT-based machine learning methods, particularly XGBoost, in improving the accuracy and efficiency of environmental prediction and monitoring systems [25].

In response to the increasing frequency of extreme weather events in Jakarta, this study compares XGBoost and LightGBM in classifying rainfall intensity using historical meteorological data, including temperature, humidity, solar radiation, wind speed, and rainfall. Both models are evaluated based on accuracy, computational efficiency, robustness, and their ability to address class imbalance. The objective is to determine which algorithm provides better performance to support early warning systems and strengthen climate resilience in urban areas.

2. METHOD

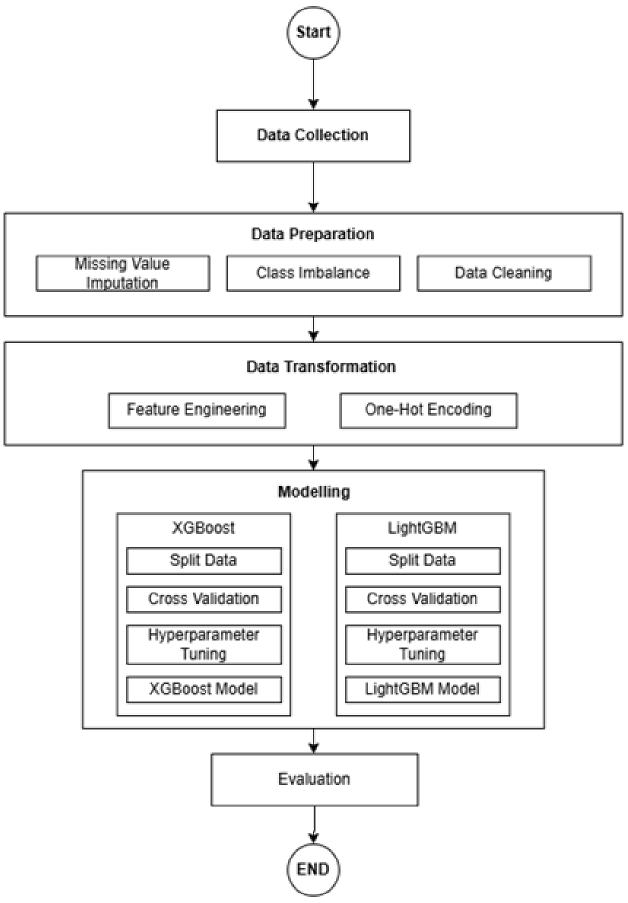

Figure 1. Model Overview

The overall methodology employed in this study is illustrated in Figure 1. The process begins with data collection, followed by a comprehensive data preparation phase that includes missing value imputation, handling class imbalance, and data cleaning. Once the dataset is prepared, it undergoes data transformation, which involves feature engineering and one-hot encoding to convert categorical variables into a machine-readable format.

The next phase is modelling, where both XGBoost and LightGBM pipelines are developed independently. Each pipeline includes data splitting, cross-validation, and hyperparameter tuning before training the respective models. Finally, the trained models are evaluated using appropriate performance metrics to assess their classification effectiveness. This structured pipeline ensures a fair and consistent comparison between the two algorithms.

2.1. Data Collection

The dataset used in this research is titled Data Meteorologi and was obtained from the Satu Data Jakarta portal, provided by BMKG in the DKI Jakarta Province. It consists of daily weather records in CSV format, with a file size of approximately 4.97 MB. As shown in Table 1, the dataset includes key meteorological variables such as air temperature (temp_air), humidity (hum_air), wind speed (ff), and pressure gradient (grad), along with the target variable label_cuaca, which is categorized into three classes: 0 = No Rain, 1 = Cloudy, and 2 = Rain. The completeness and relevance of these features make the dataset suitable for predictive analysis and the development of a weather classification model in this study.

Table 1. Dataset

Name	Type
periode_data	int64
tanggal	int64
stasiun_pengamatan	object
grad	float64
temp_air	float64
hum_air	float64
ff	float64
rain	float64

2.2. Data Preparation

This is an example of the use of sub-chapters in a paper. Sub-chapters are allowed to be included in all chapters, except in the conclusion.

Data preparation ensures quality before modeling by checking structure, types, and missing values. Mean imputation is used for temperature and humidity, and rainfall is set to zero if missing. Oversampling addresses class imbalance, while cleaning removes irrelevant, inconsistent, and duplicate data for reliable modeling.

2.2.1. Missing Value Imputation

Missing values arise from incomplete collection or data errors, which can reduce dataset quality and bias model outcomes. These are classified as MCAR, MAR, or MNAR, each requiring specific handling. Imputation methods, such as mean or regression-based techniques, are commonly applied to preserve data consistency and improve learning reliability [26].

2.2.2. Class Imbalance

Class imbalance occurs when one class dominates the dataset, leading to biased predictions and poor performance on minority classes. This can be mitigated through resampling techniques such as oversampling or undersampling [27] , which help the model learn equally from all classes and improve overall classification performance.

2.2.3. Data Cleaning

Data cleaning ensures validity by handling missing values, duplicates, and normalizing data. In sentiment analysis, steps like case folding and stemming improve structure [28], while removing irrelevant data boosts accuracy [29].

2.3. Data Transformation

After ensuring the data is clean and valid, data transformation is performed to convert it into a format suitable for modeling. This step enhances model training and ensures features are properly processed.

2.3.1. Feature Engineering

Feature engineering refines raw data into structured, informative inputs that enhance model accuracy and generalization. It involves creating, selecting, or modifying features to reduce noise and highlight patterns, which ultimately improves model reliability and predictive performance [30]. In this study, feature engineering includes not only one-hot encoding but also the creation of new features such as temp_minus_hum (temperature minus humidity) to capture relevant interactions and enrich the input data.

2.3.2. One-hot Encoding

One-hot encoding converts categorical features into binary vectors to eliminate ordinal bias in non-numeric data. While effective, this method can significantly increase feature dimensionality and lead to class imbalance issues, particularly with high-cardinality attributes. Moreover, it may reduce feature significance in distance-based algorithms without appropriate normalization [31].

2.4. Modeling

After data transformation, XGBoost and LightGBM models were developed and evaluated using cross-validation and independent testing to assess their accuracy, generalization, and robustness. This approach ensures reliable performance on both training and unseen data.

2.4.1. Split Data

Data splitting divides the dataset into training and testing sets to assess model generalization. Ratios like 80:20 or 70:30 help balance learning and evaluation, and it's done after preprocessing to ensure reliable results [32], [33]. In this study, an 80:20 split was used, along with the random_state parameter to ensure reproducibility and the stratify option to maintain proportional class distribution in both sets.

2.4.2. Cross Validation

Cross-validation is a robust technique to evaluate model performance by dividing data into several folds. The model is trained on k–1 folds and tested on the remaining fold, repeating the process k times. K-Fold Cross-Validation helps reduce bias and variance in evaluation, offering a more accurate estimate of real-world performance [34].

2.4.3. Hyperparameter Tuning

Hyperparameters are preset values like learning rate, number of trees (n_estimators), or tree depth (max_depth) that guide how a model learns. Unlike parameters learned during training, they are set beforehand and tuned using methods like Grid or Random Search. Proper tuning can significantly improve model performance and prediction accuracy in various tasks [35].

2.4.4. XGBoost

Extreme Gradient Boosting (XGBoost) is a scalable, tree-based ensemble algorithm that builds models in a stage-wise manner, where each new tree attempts to correct the errors made by previous trees. It combines gradient boosting with advanced techniques such as L1/L2 regularization to prevent overfitting, sparse-aware handling for missing values, and parallel processing for faster computation.

These features make XGBoost highly efficient and well-suited for large, structured datasets, including meteorological data used in rainfall prediction tasks [36], [37].

The model's objective function at iteration t combines a loss function and a regularization term, as shown in Equation (1):

$$L^{(t)} = \sum_{i=1}^{n} l\,(y_i, y_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad (1)$$

The first term evaluates prediction error, while the regularization term, as shown in Equation (2):

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \qquad (2)$$

penalizes complexity, where T is the number of leaves and w_j are leaf scores. XGBoost optimizes using a second-order Taylor expansion with both gradients and Hessians.

2.4.5. LightGBM

LightGBM is an efficient GBDT-based algorithm optimized for large and high-dimensional data. It uses techniques like histogram-based learning, EFB, and GOSS to boost speed and reduce memory usage [38], [39]. Trees are grown leaf-wise based on maximum loss reduction, with regularization to prevent overfitting. Its training relies on a second-order Taylor expansion to approximate the objective function at iteration t, as shown in Equation (3):

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \qquad (3)$$

This function combines first-order gradients g_i and second-order gradients h_i to optimize model performance efficiently at each boosting step, where $\Omega(f_t)$ represents the regularization term.

2.5. Evaluation

Evaluation metrics in classification, such as the confusion matrix, are key to assessing model performance by comparing predictions with actual labels. The matrix reveals correct and incorrect classifications, offering visual and numerical insights into model behavior. It also forms the basis for calculating key evaluation parameters as summarized in Table 2 including accuracy, precision, recall, and F1-score, enabling a thorough performance evaluation [40], [41].

Table 2. Evaluation Matrix

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

Evaluation metrics are essential for measuring classification performance:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (4)$$

$$Precision = \frac{TP}{TP + FP} \qquad (5)$$

$$Recall = \frac{TP}{TP + FN} \qquad (6)$$

$$F1 = 2x\,\frac{Precision\,.\,Recall}{Precision + Recall} \qquad (7)$$

$$Specificity = \frac{TN}{TN+FP} \qquad (8)$$

These evaluation metrics, as defined in Equations (4) to (8), offer a more comprehensive view of model performance, particularly under conditions of class imbalance. While accuracy (Equations 4) is commonly used, it can be misleading when the dataset is imbalanced, as it may reflect high performance by favoring the majority class. In contrast, precision (Equations 5) and recall (Equations 6) evaluate the model's ability to correctly identify positive cases, with F1-score (Equations 7) balancing the trade-off between them. Additionally, specificity (Equations 8) measures the ability to correctly identify negative cases. Together, these metrics provide a more balanced and insightful evaluation of classification models by addressing both types of prediction errors false positives and false negatives.

3. RESULT

3.1. Data Preparation Results

3.1.1. Missing Value Imputation

The imputation process was carried out to complete the dataset before transformation and modeling. Initial data exploration revealed missing values in the temp_air, hum_air, and rain columns. For temp_air and hum_air, the missing values were imputed using their respective mean values, as both features followed a normal distribution and showed no significant outliers. This approach ensures that the central tendency of the data remains intact. Meanwhile, missing values in the rain column were replaced with zeros, based on the reasonable assumption that no recorded value indicates no rainfall on those days. This step helped maintain data consistency while minimizing potential bias.

These steps were implemented using the fillna() function from the pandas library, a standard method for handling missing data in Python-based workflows. After applying imputation, the temp_air, hum_air, and rain columns were successfully completed and no longer contained missing values, ensuring they were ready for transformation and modeling. However, the grad and ff columns still had 189 and 230 missing entries, respectively, requiring further preprocessing such as interpolation or removal. A summary of missing values before and after imputation is shown in Table 3 to provide a clearer picture of data completeness.

Table 3. Missing Value Imputation

Predicted Positive	Predicted Negative
periode_data	0
tanggal	0
stasiun_pengamatan	0
grad	189
temp_air	0
hum_air	0
ff	230
rain	0

3.1.2. Class Imbalance

To address the class imbalance in the label_cuaca variable, an initial analysis revealed that the dataset exhibited a disproportionate distribution across the three predefined weather categories: 0 for Clear, 1 for Cloudy, and 2 for Rainy. Such imbalance can significantly hinder the learning process of classification models, as they may become biased toward predicting the majority class while neglecting underrepresented classes. This often results in inflated accuracy scores but poor recall and precision for minority categories, ultimately reducing the model's effectiveness in real-world applications.

To mitigate this issue, an oversampling strategy was employed during the data preprocessing phase. The original dataset was first divided into separate subsets based on the class labels to facilitate targeted resampling. For the underrepresented classes (Cloudy and Rainy), additional samples were generated through random resampling with replacement using the resample() function from the scikit-learn library. This technique increases the presence of minority classes in the dataset without altering the original feature distributions.

After generating sufficient samples, the resampled subsets were merged with the data from the majority class (Clear) to form a new, balanced dataset. To ensure randomness and prevent any sequential bias during training, the combined dataset was shuffled. This balancing process not only supports fairer and more representative model training but also enhances the model's ability to generalize across all class categories. The final class distribution after this preprocessing step can be seen in Table 4.

Table 4. Class Imbalance

label_cuaca	Predicted Negative
0	60944
1	60944
2	60944

3.1.3. Cleaning Data

After completing the imputation process, data cleaning was performed to further prepare the dataset for transformation and modeling. This step involved handling the remaining missing values and checking for duplicate entries. Specifically, missing data was still present in the grad and ff features. Given the importance of these numerical variables for model performance, rows containing missing values were removed using the dropna() method to avoid introducing bias or incomplete information.

Additionally, the dataset was examined for duplicate records using .duplicated().sum(), which helps identify redundant entries that could skew the model. Any detected duplicates were then removed using .drop_duplicates() to maintain data integrity and ensure that each observation contributed unique information to the training process.

3.2. Modeling Results

3.2.1. Split Data

This is an example of the use of sub-sub-chapters in a paper. Sub-Sub-chapters are allowed to be included in all chapters, except in the conclusion.

To evaluate the model effectively and ensure consistent and unbiased performance assessment, the dataset was divided into training and testing subsets using the train_test_split() function from sklearn.model_selection. An 80:20 ratio was applied, where 80% of the data was used for training to capture essential patterns, while the remaining 20% was reserved for evaluating the model's ability to generalize on unseen data. This split ratio is commonly adopted in machine learning workflows, offering a good balance between learning and validation.

Table 5. Split Data

label_cuaca	Predicted Negative
X_train	146008, 5
X_test	146008, 5

To ensure reproducibility of results, a fixed random_state value was assigned so that the same split could be replicated across different runs. Additionally, the stratify=y parameter was used to preserve the class distribution in both training and testing sets. This is especially important for datasets with class imbalance, as it guarantees that all weather categories are proportionally represented during both training and evaluation. This careful splitting strategy helps prevent biased performance metrics and supports a more robust classification model. The final distribution of each class in both subsets is summarized in Table 5.

3.2.2. Cross Validation

To evaluate the baseline performance of both models before hyperparameter tuning, a 5-fold cross-validation was performed using the cross_val_score() function from sklearn.model_selection. This method offers a more reliable performance estimate by reducing the variance of a single train-test split. Two models were initialized: XGBoost (with 'mlogloss' as the evaluation metric) and LightGBM. The training data was divided into five parts, where each model was trained on four folds and validated on the remaining one. Accuracy was used to measure performance in predicting weather categories. The baseline cross-validation results for both models are presented in Table 6.

Table 6. Cross Validation

Name	Accuracy
XGBoost	0.8579
LightGBM	0.8324

The results showed that XGBoost consistently achieved a higher average accuracy than LightGBM, indicating stronger baseline performance under default settings. This suggests that XGBoost may generalize better on the given dataset and is more effective in capturing the underlying patterns of the data. These findings provide a useful reference point for further enhancements, such as hyperparameter tuning, additional preprocessing, or feature engineering, to maximize model performance. Moreover, the comparison helps guide informed decisions in selecting the most suitable model for multiclass weather classification tasks.

3.2.3. Hyperparameter

To optimize both the XGBoost and LightGBM models, a comprehensive hyperparameter search space was defined and utilized in conjunction with Randomized Search, a tuning technique that randomly samples a fixed number of parameter combinations from predefined distributions. This approach allows for broader coverage of the parameter space compared to manual tuning and is significantly more efficient than exhaustive Grid Search, particularly when dealing with numerous hyperparameters and limited computational resources. By leveraging this method, the models have a higher chance of identifying optimal or near-optimal parameter sets that improve overall performance while maintaining reasonable training times.

For XGBoost, the tuning involved parameters such as max_depth, learning_rate, n_estimators, subsample, colsample_bytree, gamma, reg_lambda, and reg_alpha, which influence model complexity, learning dynamics, and regularization strength. These parameters were carefully adjusted to improve model accuracy, training stability, and to reduce the risk of overfitting, especially in the presence of noisy or imbalanced data. Similarly, the LightGBM model was optimized using a comparable set of parameters, enabling it to adapt tree structure, sampling techniques, and regularization strategies to better capture patterns in the data and improve overall predictive performance across different rainfall intensity classes.

By applying RandomizedSearchCV, each model was evaluated through cross-validation across multiple randomly selected combinations of hyperparameters. This method allows efficient exploration of the parameter space without the exhaustive cost of a full grid search. It balances computational efficiency with the goal of finding strong-performing configurations, selecting the one that produced the best average performance across folds. As a result, the models are better tuned for generalization and are less likely to overfit the training data. The optimal hyperparameter configurations for both XGBoost and LightGBM are summarized in Table 7, serving as the foundation for final model evaluation.

Table 7. Hyperparameter

Parameter	Values
max_depth	[5, 7, 9]
learning_rate	[0.01, 0.05, 0.1]
n_estimators	[200, 300, 500]
subsample	[0.6, 0.8, 1.0]
colsample_bytree	[0.6, 0.8, 1.0]
gamma	[0, 1, 5]
reg_lambda	[1, 5, 10]
reg_alpha	[0, 1, 5]

3.3. Evaluation

3.3.1. Evaluation Metrics Performance

XGBoost was correct 95% of the time, which is particularly important in domains like rainfall prediction, where false alarms (false positives) could lead to unnecessary public concern or resource allocation. The recall value of 93% reflects XGBoost's capability to detect actual occurrences of each class, such as true rainy days. The F1-score of 94%, being the harmonic mean of precision and recall, illustrates a balance between the two. Finally, a specificity of 96% shows that the model could also correctly identify negative instances (e.g., "no rain" days). See Table 8 for details.

Meanwhile, LightGBM achieved slightly lower values: accuracy of 91%, precision of 90%, recall of 89%, F1-score of 89%, and specificity of 94%. Although these results are still considered strong, they suggest that LightGBM may struggle more than XGBoost in correctly identifying minority classes or in balancing trade-offs between precision and recall. The complete evaluation for LightGBM is shown in Table 9.

Table 8. XGBoost Evaluation

Predicted Positive	Predicted Negative
Accuracy	0.94
Precision	0.95
Recall	0.93
F1-score	0.94
Specificity	0.96

Table 9. LightGBM Evaluation

Predicted Positive	Predicted Negative
Accuracy	0.91
Precision	0.90
Recall	0.89
F1-score	0.89
Specificity	0.94

These results indicate that both models are capable of performing multiclass classification of rainfall events with reasonable accuracy, demonstrating their potential for use in meteorological analysis. However, XGBoost consistently delivers more stable and reliable performance across various evaluation metrics, including accuracy, precision, recall, and F1-score. Its ability to generalize better, especially in handling imbalanced classes, makes it a stronger and more robust candidate for practical applications in weather prediction systems. This advantage is particularly valuable for early warning systems, where consistent and dependable predictions are critical for timely decision-making and disaster preparedness in urban environments.

3.3.2. Hyperparameter Tuning

To achieve optimal model performance, hyperparameter tuning was performed on both algorithms used in this study, namely XGBoost and LightGBM. Hyperparameters are external configuration settings that are not learned directly from the training data but significantly influence the model's learning process and outcomes. Proper tuning of these parameters helps balance bias and variance, improve generalization on unseen data, and enhance the model's ability to capture complex patterns in meteorological variables. Without tuning, models may underperform due to suboptimal configurations, regardless of the quality of the input data.

In this research, a manual tuning approach was applied by adjusting commonly used parameter values based on prior studies and empirical testing, due to computational constraints. Key parameters adjusted include n_estimators, max_depth, and learning_rate for XGBoost, as well as num_leaves, feature_fraction, and learning_rate for LightGBM.

The optimal sets of hyperparameters resulting in the best model performance are summarized in Table 10, following multiple rounds of evaluation and fine-tuning:

Table 10. Hyperparameter Best Value

Model	Parameter	Best Value
XGBoost	max_depth	5
	learning_rate	0,1
	n_estimators	200
	subsample	0.8
LightGBM	colsample_bytree	0.8
	gamma	5
	reg_lambda	1
	reg_alpha	1

These hyperparameter values were selected based on the combinations that yielded the highest accuracy and AUC scores during cross-validation. Although this tuning process did not yet involve automated techniques such as GridSearchCV or Optuna, it still contributed to improved model performance.

For future research, more systematic hyperparameter optimization methods may be applied to further enhance the predictive capability of the models.

3.3.3. Evaluation Metrics Performance

To better understand the classification performance of both models, confusion matrices were constructed for XGBoost and LightGBM, as shown in Figure 2. These matrices provide detailed insights into how well each model distinguishes between the three weather classes: 0 = No Rain, 1 = Cloudy, and 2 = Rain. The diagonal elements represent correctly classified instances for each class, indicating high confidence in those predictions. Meanwhile, the off-diagonal elements reflect misclassifications, revealing where the models tend to confuse one class with another. This analysis is crucial for identifying potential weaknesses, especially in borderline cases such as differentiating between cloudy and rainy conditions, which often share overlapping meteorological features. By examining these patterns, researchers can better understand model behavior, assess robustness across different weather scenarios, and make targeted improvements to enhance prediction reliability and support more accurate decision-making in climate-sensitive applications.

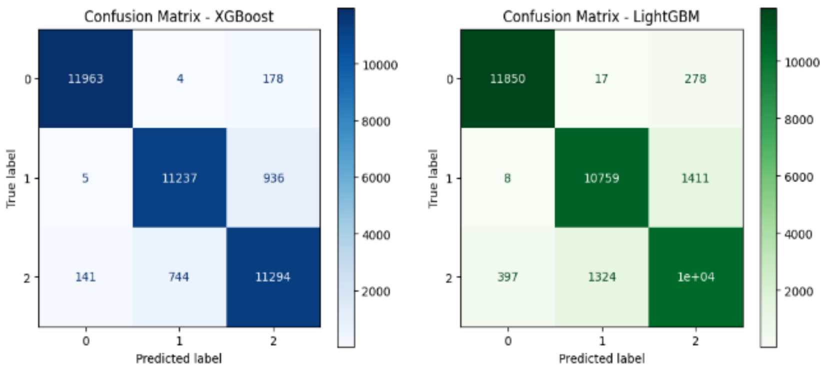

Figure 2. Confusion matrix of classification results

XGBoost Confusion Matrix:

- No Rain (Class 0): 11,963 instances were correctly classified. Only 4 were misclassified as Cloudy, and 178 as Rain.
- Cloudy (Class 1): 11,237 correctly classified, with 5 instances predicted as No Rain, and 936 predicted as Rain.
- Rain (Class 2): 11,294 were correctly identified, but 141 were predicted as No Rain, and 744 as Cloudy.

This matrix demonstrates that XGBoost has strong overall accuracy, with misclassifications being relatively low and mostly occurring between the Cloudy and Rain classes, which often exhibit similar atmospheric characteristics such as moderate humidity and temperature levels. These subtle overlaps can make it challenging for models to distinguish between the two categories, yet XGBoost still manages to maintain high precision and recall across all classes, highlighting its robustness in handling complex weather classification tasks.

- No Rain (Class 0): 11,850 correctly predicted, with 17 misclassified as Cloudy and 278 as Rain.
- Cloudy (Class 1): 10,759 instances correctly identified, 8 misclassified as No Rain, and 1,411 as Rain.
- Rain (Class 2): 10,000 correctly predicted, while 397 were predicted as No Rain, and 1,324 as Cloudy.

Compared to XGBoost, LightGBM shows higher misclassification rates, particularly for the Rain class, where a significant number of instances were confused with Cloudy. This suggests that LightGBM may have more difficulty distinguishing between mid-to-heavy precipitation patterns, possibly due to overlapping feature distributions or lower sensitivity to minority class boundaries.

3.3.4. Comparative Visualization

To provide a clear and intuitive comparison between the XGBoost and LightGBM models, a line chart was constructed to visualize five key evaluation metrics: accuracy, precision, recall, F1-score, and specificity, as shown in Figure 3. This visual comparison enables a direct assessment of the relative strengths and weaknesses of each model.

The blue line representing XGBoost consistently lies above the green line for LightGBM, indicating superior performance across all metrics. XGBoost achieved the highest value in specificity (0.96), followed by precision (0.95), and both accuracy and F1-score (0.94), with recall at 0.93. This balanced performance shows that the model is not only precise but also effective in detecting true positives and correctly identifying negatives.

In contrast, LightGBM achieved a maximum value of specificity at 0.94, but scored lower on other metrics, with values around 0.91 for accuracy, 0.90 for precision, and 0.89 for both recall and F1-score. The consistently lower scores suggest that while LightGBM performs reasonably well, it is igurerecall in imbalanced data scenarios.

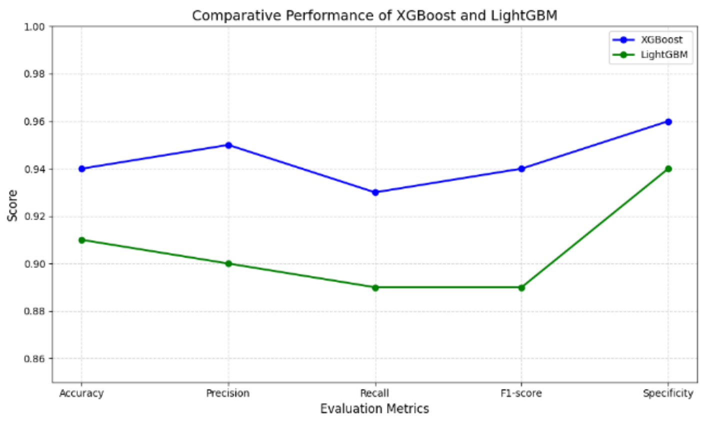

Figure 3. Comparative performance of XGBoost and LightGBM

These findings confirm that while both models perform well, XGBoost offers more stable and precise classificationan advantage that is particularly crucial in rainfall detection and early warning systems, where prediction errors such as false positives or false negatives can lead to significant operational inefficiencies, misallocation of resources, or even safety risks for the public.

4. DISCUSSIONS

The evaluation results show that the XGBoost algorithm outperforms LightGBM in multiclass rainfall classification in Jakarta. XGBoost achieved an accuracy of 94%, with precision, recall, and F1-score values of 95%, 93%, and 94%, respectively. Meanwhile, LightGBM reached an accuracy of 91% and an F1-score of 89%. These findings indicate that although both models are capable of performing classification tasks, XGBoost produces more stable results, particularly in tropical weather datasets that exhibit highly variable and complex patterns.

The performance difference can be attributed to the algorithmic approaches of each model. XGBoost incorporates L1 and L2 regularization as well as complexity-based tree pruning, which contribute to its ability to avoid overfitting and enhance generalization [10]. In contrast, LightGBM, while offering advantages in training speed and memory efficiency, appears to struggle in distinguishing overlapping patterns between rainfall classes, especially between “cloudy” and “rainy” conditions.

These results are consistent with previous studies. For instance, Maharina et al. found that although LightGBM was effective for flood classification, its performance could vary depending on interaction features and class imbalance handling [18]. Likewise, Yasper et al. demonstrated that hyperparameter optimization using GridSearchCV significantly improved the performance of XGBoost in binary rainfall classification tasks [10]. The present study extends these findings to a multiclass setting, showing that XGBoost remains superior in more complex prediction scenarios.

Furthermore, confusion matrix analysis reveals that XGBoost had fewer classification errors than LightGBM, particularly for the minority class “rain.” This advantage is crucial in the context of early warning systems, where classification errors could directly affect disaster mitigation efforts and operational decision-making [16].

Although XGBoost demonstrates a 3% higher accuracy, it is important to consider the trade-off with computational efficiency. This study did not explicitly measure training time or memory usage;

however, such factors could be critical when deploying models in resource-constrained environments. Discussing whether the performance gain justifies the potentially higher computational cost of XGBoost would offer valuable practical insight, especially for applications requiring frequent retraining or real-time prediction capabilities.

5. CONCLUSION

XGBoost and LightGBM algorithms were employed to classify rainfall intensity based on historical meteorological data in Jakarta. The evaluation results indicate that XGBoost outperforms LightGBM in terms of accuracy, consistency of evaluation metrics, and fewer classification errors. With an accuracy of 94% and an F1-score of 94%, XGBoost demonstrates higher reliability compared to LightGBM, which achieved an accuracy of 91% and an F1-score of 89%.

The consistent performance of XGBoost confirms its capability in handling complex and nonlinear tropical weather data, as well as its effectiveness in classifying all three rainfall categories in a balanced manner. Therefore, XGBoost is recommended as the more suitable model for implementation in urban weather prediction systems and early warning mechanisms for hydrometeorological disasters in regions like Jakarta. The integration of this model can enhance forecasting accuracy and support more responsive decision-making amid increasing climate variability.

For future research, several directions can be explored to enhance model performance and practical deployment. First, deep learning architectures such as Long Short-Term Memory (LSTM) or Transformer networks may be investigated, as they are designed to capture temporal dependencies inherent in meteorological time series data. Second, incorporating additional data sources—such as satellite imagery or weather radar data could enrich feature sets and improve prediction accuracy. Lastly, the most promising model from this study may be developed into a real-time early warning system prototype to assess its performance under operational conditions.

By including these concrete suggestions, this study not only highlights current findings but also contributes valuable insight for further exploration in the domain of data-driven weather prediction and climate risk mitigation.

CONFLICT OF INTEREST

The authors declares that there is no conflict of interest between the authors or with research object in this paper.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to the Meteorological, Climatological, and Geophysical Agency (BMKG) and the Satu Data Jakarta portal for providing access to the historical meteorological dataset used in this study. The authors also acknowledge the support of their respective academic institutions for facilitating the research infrastructure and technical resources needed to complete this work.

Additionally, the authors appreciate the developers and open-source communities behind the XGBoost and LightGBM libraries, whose tools and documentation significantly contributed to the modeling and evaluation processes. Their contributions have been instrumental in enabling accurate and reliable implementation of machine learning methods for rainfall classification.

REFERENCES

[1] R. L. Melanwati, E. Sumarminingsih, and H. Pramoedyo, "Transformasi Kota Cerdas dalam Mitigasi Banjir: Pemodelan Curah Hujan DKI Jakarta dengan Pendekatan Spatial Vector Autoregressive (SpVAR) dan Pemetaan Bobot Queen Contiguity," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 10, no. 6, pp. 1285–1294, 2023, doi: 10.25126/jtiik.1067537.

[2] F. Hamami and I. A. Dahlan, "Klasifikasi Cuaca Provinsi Dki Jakarta Menggunakan Algoritma Random Forest Dengan Teknik Oversampling," *J. Teknoinfo*, vol. 16, no. 1, p. 87, 2022, doi: 10.33365/jti.v16i1.1533.
[3] B. Adiyasa *et al.*, "Deteksi Bencana Banjir Berdasarkan Data Curah Hujan Di Daerah Jakarta Menggunakan Logistic Regression," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. 1–8, 2025, [Online]. Available: http://j-ptiik.ub.ac.id
[4] R. Fredyan, M. R. N. Majiid, and G. P. Kusuma, "Spatiotemporal Analysis for Rainfall Prediction Using Extreme Learning Machine Cluster," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 13, no. 6, pp. 2240–2248, 2023, doi: 10.18517/ijaseit.13.6.18214.
[5] M. Mutiara Ramadita and M. Y. Wijaya, "Prediksi Curah Hujan di Jakarta Menggunakan Model Hybrid (DWT-SVR-Prophet)," *Indones. J. Comput. Sci.*, vol. 13, no. 5, pp. 1–16, 2024, doi: https://doi.org/10.33022/ijcs.v1.
[6] C. R. Malino and M. Arsyad, "Analisis Parameter Curah Hujan dan Suhu Udara di Kota Makassar Terkait Fenomena Perubahan Iklim," *J. Sains dan Pendidik. Fis.*, vol. 17, no. 2, pp. 139–145, 2021, doi: 10.35580/jspf.v17i2.22167.
[7] D. A. S. Pertiwi, S. Sutisna, M. Supriyatno, Y. Norman, and J. A. I. Paski, "Kajian Perubahan Distribusi Frekuensi Curah Hujan Di Jakarta Periode 1991 - 2020," *J. Geogr. Edukasi dan Lingkung.*, vol. 8, no. 2, pp. 183–191, 2024, doi: 10.22236/jgel.v8i2.12882.
[8] A. Diando, L. M. Limantara, and S. Wahyuni, "Estimasi Tinggi Curah Hujan dari Data Klimatologi Menggunakan Model Artificial Neural Network (ANN) di Jakarta Pusat, Provinsi DKI Jakarta," *J. Teknol. dan Rekayasa Sumber Daya Air*, vol. 4, no. 1, pp. 15–24, 2023, doi: 10.21776/ub.jtresda.2024.004.01.002.
[9] N. Anggraini, S. J. Putra, L. K. Wardhani, F. D. U. Arif, N. Hakiem, and I. M. Shofi, "A Comparative Analysis of Random Forest, XGBoost, and LightGBM Algorithms for Emotion Classification in Reddit Comments," *J. Tek. Inform.*, vol. 17, no. 1, pp. 88–97, 2024, doi: 10.15408/jti.v17i1.38651.
[10] A. Yasper, D. Handoko, M. Putra, H. K. Aliwarga, and M. S. R. Rosid, "Hyperparameters Optimization in XGBoost Model for Rainfall Estimation: A Case Study in Pontianak City," *J. Penelit. Pendidik. IPA*, vol. 9, no. 9, pp. 7113–7121, 2023, doi: 10.29303/jppipa.v9i9.3890.
[11] M. F. Asnawi, H. H. Bisono, and M. A. Megantara, "Aplikasi Prediksi Banjir Menggunakan Algoritma XGBoost Berbasis Website," *J. Econ. Manag. Account. Technol.*, vol. 7, no. 2, pp. 379–389, 2024, doi: 10.32500/jematech.v7i2.7644.
[12] I. Maulita, C. R. A. Widiawati, and A. M. Wahid, "Analisis Komparatif Linear Regression, Random Forest, dan Gradient Boosting untuk Prediksi Banjir," *J. Pendidik. Dan Teknol. Indones.*, vol. 4, no. 8, pp. 369–379, 2024, [Online]. Available: https://doi.org/10.52436/1.jpti.599
[13] T. S. Wibawa, N. K. Ningrum, and A. Syahreza, "Comparison of CatBoost and LightGBM Models for Air Humidity Prediction," *J. Appl. Informatics Comput.*, vol. 9, no. 3, pp. 803–809, 2025, doi: doi.org/10.30871/jaic.v9i3.9570.
[14] P. Septiana Rizky, R. Haiban Hirzi, and U. Hidayaturrohman, "Perbandingan Metode LightGBM dan XGBoost dalam Menangani Data dengan Kelas Tidak Seimbang," *J Stat. J. Ilm. Teor. dan Apl. Stat.*, vol. 15, no. 2, pp. 228–236, 2022, doi: 10.36456/jstat.vol15.no2.a5548.
[15] K. Handayani and B. Lailiah, "Comparison of XGboost, Extra Trees, and LightGBM with SMOTE for Fetal Health Classification," *Sistemasi*, vol. 13, no. 3, p. 980, 2024, doi: 10.32520/stmsi.v13i3.3646.
[16] A. Fauziah, H. Hermanto, and M. A. Sukmarini, "Extreme Gradient Boosting pada Peramalan Pola Curah Hujan Bulanan Kabupaten Banyuwangi," *J. Kridatama Sains Dan Teknol.*, vol. 6, no. 02, pp. 430–440, 2024, doi: 10.53863/kst.v6i02.1154.
[17] V. No, A. Syahreza, N. K. Ningrum, and M. A. Syahrazy, "Perbandingan Kinerja Model Prediksi Cuaca : Random Forest , Support Vector Regression , dan XGBoost," *J. Pendidik. Inform.*, vol. 8, no. 2, pp. 526–534, 2024, doi: 10.29408/edumatic.v8i2.27640.
[18] M. K. H. Maharina, Tukino Paryono, Ahmad Fauzi, Jamaludin Indra, Sihabudin and L. T. Rizki, "Machine Learning Models for Predicting Flood Events Using Weather Data: An Evaluation of Logistic Regression, LightGBM, and XGBoost," *J. Appl. Data Sci.*, vol. 6, no. 1, pp. 496–507,

2025, doi: 10.47738/jads.v6i1.503.

[19] A. Wijayanto, A. Sugiharto, and R. Santoso, "Identifikasi Dini Curah Hujan Berpotensi Banjir Menggunakan Algoritma Long Short-Term Memory (Lstm) Dan Isolation Forest," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 11, no. 3, pp. 637–646, 2024, doi: 10.25126/jtiik.938718.

[20] H. Gani, A. D. Damayanti, M. M. Mubarak, and H. Gani, "An Explainable Machine Learning Model to Explain the Influential Climate Parameters Based on Rainfall Prediction," *J. ITMedia Inf. IT STMIK Handayani*, vol. 15, no. 2, pp. 98–110, 2024, doi: doi.org/10.37639/jti.v15i2.378.

[21] M. Hermansyah, A. Saikhu, and B. Amaliah, "Pemodelan Data Radiosonde Menggunakan Stacking Ensemble Untuk Klasifikasi Hujan," *JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.*, vol. 10, no. 2, pp. 1678–1687, 2025.

[22] D. Sangaji and T. Sutabri, "Analisis XGBoost dan Random Forest untuk Prediksi Curah Hujan dalam Mendukung Mitigasi Karhutla," *J. Pustaka AI*, vol. 5, no. 1, pp. 13–18, 2025, doi: doi.org/10.55382/jurnalpustakaai.v5i1.905.

[23] V. N. Juli, M. K. Aulia, E. Utaminingsih, and N. Prihatin, "Model Prediksi Risiko Kesehatan Perkotaan Berbasis Lingkungan dengan XGBoost," *Comput. Sci.*, vol. 5, no. 2, pp. 95–102, 2025.

[24] N. Alamsyah, B. Budiman, T. P. Yoga, and R. Y. R. Alamsyah, "Xgboost Hyperparameter Optimization Using Randomizedsearchcv for Accurate Forest Fire Drought Condition Prediction," *J. Pilar Nusa Mandiri*, vol. 20, no. 2, pp. 103–110, 2024, doi: 10.33480/pilar.v20i2.5569.

[25] A. F. B. Sajiwo, B. Rahmat, and A. Junaidi, "Klasifikasi Indeks Standar Pencemaran Udaran (Ispu) Menggunakan Algoritma Xgboost Dengan Teknik Imbalanced Data (Smote)," *J. Inform. dan Tek. Elektro Terap.*, vol. 12, no. 3, 2024, doi: 10.23960/jitet.v12i3.4699.

[26] M. Anwar Sadat, P. Pujiono, A. Pambudi, and S. Ibad, "Comparison of Algorithm Between Classification & Regression Trees and Support Vector Machine in Determining Student Acceptance in State Universities," *J. Tek. Inform.*, vol. 4, no. 6, pp. 1589–1604, 2024, doi: 10.52436/1.jutif.2023.4.6.1565.

[27] A. Diastama *et al.*, "Sentiment Analysis Classification In Women ' S E-Commerce Reviews Klasifikasi Sentimen Analisis Pada Women ' S E -Commerce Reviews Dengan Pendekatan Machine Learning," *J. Tek. Inform.*, vol. 5, no. 6, pp. 1549–1559, 2024.

[28] D. Rifaldi, Abdul Fadlil, and Herman, "Teknik Preprocessing Pada Text Mining Menggunakan Data Tweet 'Mental Health,'" *J. Pendidik. Teknol. Inf.*, vol. 3, no. 2, pp. 161–171, 2023, doi: 10.51454/decode.v3i2.131.

[29] T. Gori, A. Sunyoto, and H. Al Fatta, "Preprocessing Data dan Klasifikasi untuk Prediksi Kinerja Akademik Siswa," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 11, no. 1, pp. 215–224, 2024, doi: 10.25126/jtiik.20241118074.

[30] Z. Z. Alkaf *et al.*, "Unraveling Of Men'S Fragrance Preferences On Online Marketplaces : A Machine Learning Study Using Dbscan Clustering," *J. Tek. Inform.*, vol. 5, no. 6, pp. 1913–1920, 2025, doi: doi.org/10.52436/1.jutif.2024.5.6.4187.

[31] A. Agustiningsih, Y. Findawati, and I. Alnarus Kautsar, "Classification of Vocational High School Graduates' Ability in Industry Using Extreme Gradient Boosting (Xgboost), Random Forest, and Logistic Regression," *J. Tek. Inform.*, vol. 4, no. 4, pp. 977–985, 2023, doi: 10.52436/1.jutif.2023.4.4.945.

[32] A. I. Pradana *et al.*, "Perbandingan Data Untuk Memprediksi Ketepatan Studi Berdasarkan Atribut Keluarga Menggunakan Machine Learning," *J. Inform.*, vol. 8, no. 2, pp. 221–228, 2024, doi: 10.31000/jika.v8i2.10752.

[33] R. T. P. Sudewo, Y. Pratama, and E. Yanti, "Analisis Data Mining Untuk Prediksi Kanker Payudara Menggunakan Algoritma Klasifikasi," *J. Pustaka Data*, vol. 3, no. 2, pp. 62–69, 2023, doi: 10.55382/jurnalpustakadata.v3i2.656.

[34] R. N. Irawan, K. M. Hindrayani, and M. Idhom, "Penerapan Cross Validation sebagai Analisis Sentimen Pelayanan Publik Kereta Api Lokal Daop 8 Menggunakan Metode Multinomial Naïve Bayes," *G-Tech J. Teknol. Terap.*, vol. 8, no. 2, pp. 954–963, 2024, doi: 10.33379/gtech.v8i2.4117.

[35] D. S. Bhakti, A. Prasetyo, P. Arsi, C. S. Faculty, and U. A. Purwokerto, "Implementation of Hyperparameter Tuning in Random Forest Implementasi Hyperparameter Tuning Pada

Algoritma Random," *J. Tek. Inform.*, vol. 5, no. 4, pp. 63–69, 2024, doi: doi.org/10.52436/1.jutif.2024.5.4.2032.

[36] A. Bengnga and R. Ishak, "Penerapan XGBoost untuk Seleksi Atribut pada K-Means dalam Clustering Penerima KIP Kuliah," *Jambura J. Electr. Electron. Eng.*, vol. 5, no. 2, pp. 192–196, 2023, doi: 10.37905/jjeee.v5i2.20253.

[37] R. G. Gunawan, Erik Suanda Handika, and Edi Ismanto, "Pendekatan Machine Learning Dengan Menggunakan Algoritma Xgboost (Extreme Gradient Boosting) Untuk Peningkatan Kinerja Klasifikasi Serangan Syn," *J. Comput. Sci. Inf. Technol.*, vol. 3, no. 3, pp. 453–463, 2022, doi: 10.37859/coscitech.v3i3.4356.

[38] L. Sari, A. Romadloni, R. Lityaningrum, and H. D. Hastuti, "Implementation of LightGBM and Random Forest in Potential Customer Classification," *TIERS Inf. Technol. J.*, vol. 4, no. 1, pp. 43–55, 2023, doi: 10.38043/tiers.v4i1.4355.

[39] F. I. Kurniadi and P. D. Larasati, "Light Gradient Boosting Machine untuk Deteksi Penyakit Stroke," *J. SISKOM-KB (Sistem Komput. dan Kecerdasan Buatan)*, vol. 6, no. 1, pp. 67–72, 2022, doi: 10.47970/siskom-kb.v6i1.328.

[40] D. Normawati and S. A. Prayogi, "Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter," *J. Sains Komput. Inform.*, vol. 5, no. 2, pp. 697–711, 2021, doi: 10.30645/j-sakti.v5i2.369.

[41] Y. Afrillia, L. Rosnita, and D. Siska, "Analisis Sentimen Ciutan Twitter Terkait Penerapan Permendikbudristek Nomor 30 Tahun 2021 Menggunakan TextBlob dan Support Vector Machine," *G-Tech J. Teknol. Terap.*, vol. 6, no. 2, pp. 387–394, 2022, doi: 10.33379/gtech.v6i2.1778.