Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer
Vol. 23, No. 2, March 2024, pp. 477∼490
ISSN: 2476-9843, accredited by Kemenristekdikti, Decree No: 200/M/KPT/2020
DOI: 10.30812/matrik.v23i2.3555

Stroke Prediction with Enhanced Gradient Boosting Classifier and Strategic Hyperparameter

Dela Ananda Setyarini , Agnes Ayu Maharani Dyah Gayatri , Christian Sri Kusuma Aditya , Didih Rizki Chandranegara
Universitas Muhammadiyah Malang, Malang, Indonesia

Article Info

Article history:

Received November 13, 2023
Revised February 02, 2024
Accepted March 30, 2024

Keywords:

Comparison Boosting
Hyperparameter
Machine Learning
Stroke

ABSTRACT

A stroke is a medical condition that occurs when the blood supply to the brain is interrupted. Stroke can cause damage to the brain that can potentially affect a person's function and ability to move, speak, think, and feel normally. The effect of stroke on health emphasizes the importance of stroke detection, so an effective model is needed in predicting stroke. This research aimed to find a new approach that can improve the performance of stroke prediction by comparing four derivative algorithms from Gradient Boosting by adding hyperparameters tuning. The addition of hyperparameters was used to find the best combination of parameter values that can improve the model accuracy. The methods used in this research were Categorical Boosting, Histogram Gradient Boosting, Light Gradient Boosting, and Extreme Gradient Boosting. The research involved retrieving, cleaning, and analyzing data and then the model performance was evaluated with a confusion matrix and execution time. The results obtained were Light Gradient Boosting with Hyperparameter RandomSearchCV achieved the highest accuracy at 95% among the algorithms tested, while also being the fastest in execution. The contribution of this research to the medical field can help doctors and patients predict the occurrence of stroke early and reduce serious consequences.

Corresponding Author:

Christian Sri Kusuma Aditya, +6281359659715,
Faculty of Engineering and Department of Informatics Engineering,
Universitas Muhammadiyah Malang, Malang, Indonesia,
Email: christianskaditya@umm.ac.id.

How to Cite:
D. A. Setyarini, A. A. M. D. Gayatri, C. S. K. Aditya, and D. R. Chandranegara, "Stroke Prediction with Enhanced Gradient Boosting Classifier and Strategic Hyperparameter", *MATRIK: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer*, Vol. 23, No. 2, pp. 477-490, Mar, 2024.

1. INTRODUCTION

According to the online site www.stroke.org, stroke is a dangerous disease that can damage brain cells by attacking arteries so that blood vessels that carry oxygen and nutrients become blocked [1]. There are two types of stroke: ischemic and hemorrhagic. The American Heart Association (AHA) estimates that 87% of strokes are ischemic strokes. Only 10-15% of strokes are estimated to be hemorrhagic strokes with a higher mortality rate [2]. According to Furuta et al., stroke is one of the most serious medical problems facing Asia, Europe, and North America [3]. The World Health Organization (WHO) explains that of the ten diseases that cause death in Indonesia, stroke is ranked first, with the number of deaths reaching 131.8 per 100,000 population [4]. Stroke is classified as a fatal disease generally suffered by people over 65 years of age [5]. The WHO Statistical Report identifies stroke as one of the leading causes of disability in the world [6]. According to Statistics Canada, 36% of stroke survivors after five years experience disability, and 40% require assistance with daily activities [7]. Several aspects cause the disease, but it does not rule out the possibility that there is also a solution to the problem, one of which is the sophistication of technology. With more advanced technology, a person will see the cause or predict the occurrence of a disease through the data obtained.

Several journals have discussed different types of methods and processing stages, but they have used the same datasets and case studies. In the journal classification (Hager Ahmet et al.) entitled ”Stroke Prediction using Distributed Machine Learning Based on Apache Spark,” which uses several machine learning algorithms, some of the algorithms used include random forest with the greatest accuracy of 90%, Decision Tree, Support Vector Machine (SVM), and Logistic Regression [8]. In the journal (Tahia Tazin et al.) entitled ”Stroke Disease Detection and Prediction Using Robust Learning Approaches,” using several algorithms, namely Decision Tree, Voting Classifier, Logistic Regression, and Random Forest has the highest accuracy of 96%, Decision Tree with 94% accuracy, Voting Classifier with 91% accuracy, and Logistic Classifier with 79% accuracy [9]. Then, the journal (Minhaz Uddin Emon et al.) entitled ”Performance Analysis of Machine Learning Approaches in Stroke Prediction” with the same dataset and using several algorithms for stroke disease classification, namely the K-Nearest Neighbors Algorithm with an accuracy of 87%, Logistic Regression and Gaussian with a n accuracy of 78%, AdaBoost with 94% accuracy, XGBoost and GBC with 96% accuracy, Stochastic Gradient Descent with 65% accuracy, DTC with 91% accuracy, QDA and MLP with 79% accuracy, and Weighted Voting with the highest accuracy of 97% [10]. The main reference is the journal (Prismaharadi Aji Riyantoko et al.) entitled ”Exploratory Data Analysis and Machine Learning Algorithms to Classifying Stroke Disease” using the K-Nearest Neighbors method, XGBoost, Random Forest, Gradient Boost, Decision Tree, SVM, Stochastic Gradient Descent with each having the same accuracy of 94%. In comparison, Nave Bayes has the lowest accuracy of 20% [11].

Based on previous literature, no research uses or specifically reviews the derivative of Gradient Boosting. Therefore, this research is focused on exploring the prediction of the Gradient Boosting derivative classification model and applying RandomSearchCV and GridSearchCV hyperparameters to increase model performance. Thus, the best parameter configuration and Gradient Boosting model can be found. In research [12], the model is optimized based on grid-based hyperparameter tuning. The optimized parameters are the number of trees in a forest, the maximum number of features to split in the child node, the level of the trees in each decision tree, and the criterion used for splitting. The experimental results exhibit perfect classification on three datasets, namely 2-class leukemia, Ovarian, and SRBCT, by scoring 100% and 0.97 test accuracy on two datasets, namely 3-class Leukemia and MLL.

Hyperparameter optimization is a crucial step in fine-tuning predictive models. However, this process often involves significant computational costs, primarily because it requires time-consuming model training to assess the effectiveness of various sets of candidate hyperparameter values. [13]. Other research [14] investigates the chosen method, RandomSearch, for adjusting the hyperparameters of Support Vector Machines (SVMs), which Experimental results indicate that the predictive performance of models using Random Search is on par with those utilizing meta-heuristics and Grid Search. Random Search achieves this comparable performance while incurring a lower computational cost. This finding suggests that the optimization of SVM hyperparameters can be achieved efficiently through Random Search, offering a promising alternative to more computationally intensive methods like meta-heuristics and exhaustive Grid Search.

Boosting classifiers encounter difficulties, particularly in balancing accuracy and efficiency. Traditional implementations of boosting classifiers require scanning all data instances for every feature to estimate information gain across all possible split points. Consequently, the computational complexities of these implementations are directly proportional to the number of features and instances. This results in significant time consumption, especially when dealing with large datasets. So, this research will compare several algorithm derivatives of Gradient Boosting to find new approaches to improve performance and data accuracy. In order to achieve this goal, this research will also use optimizers GridSearch and RandomSearch to tweak model performance for optimal results. Selecting the right set of hyperparameters can be crucial for achieving optimal performance, and improper tuning might result in overfitting or underfitting. The method will later be combined with hyperparameter GridSearch and RandomSearch to determine the performance or feasibility of the method in processing data with a case study of stroke disease.

This research involves several steps, from data capture to cleaning and analysis. Hyperparameter tuning is used using GridSearchCV and RandomSearchCV techniques to ensure that the model used achieves the highest level of optimization according to the characteristics of the data. Model performance evaluation is done using a Confusion Matrix and evaluation matrix, as well as measuring execution time.

2. RESEARCH METHOD

Figure 1 represents the research methodology, which involves a series of stages: loading the dataset, data preprocessing, data splitting, the classification stage, hyperparameter tuning, and the evaluation of the classification process.

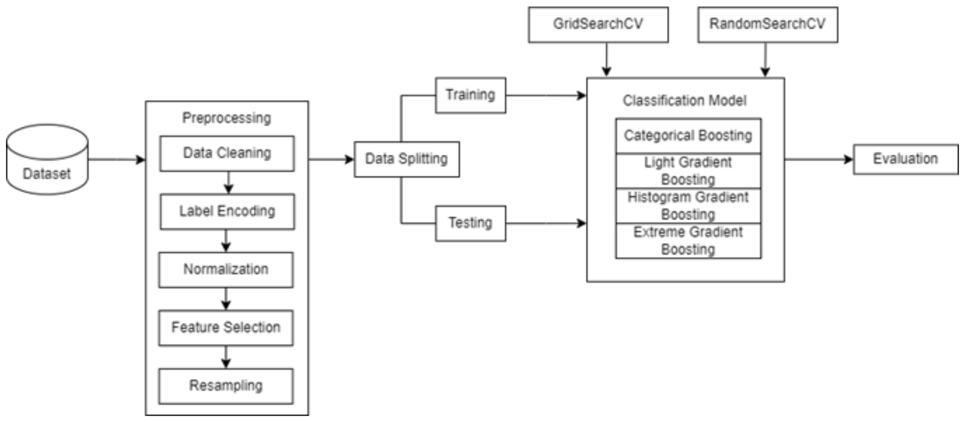

Figure 1. Research Methodology

2.1. Data Source

The dataset in this study consists of 5110 samples and 12 features obtained from Kaggle with the link https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset, where the ”id” feature will be removed because it does not provide relevant or significant information in influencing the classification results of stroke disease prediction. The target class used in this study is the ’stroke’ feature, with a value of 1 for having a stroke and 0 for not having a stroke. Information for each feature in this dataset can be seen in Table 1.

Table 1. Data description

Attribute	Information
gender	”Male”, ”Female” or ”Other”
age	age of the patient
hypertension	0 if the patient does not have hypertension, 1 if the patient has hypertension
heart_disease	0 if the patient does not have any heart diseases, 1 if the patient has a heart disease
ever_married	”No” or ”Yes”
work_type	”children”, ”Govt_jov”, ”Never_worked”, ”Private” or ”Self-employed”
residence_type	”Rural” or ”Urban”
avg_glucose_level	average glucose level in blood
bmi	body mass index
smoking_status	”formerly smoked”, ”never smoked”, ”smokes” or ”Unknown”∗
Stroke	1 if the patient had a stroke or 0 if not

2.2. Data Cleaning

Data cleaning is a data cleansing process that involves evaluating data quality and making changes, updates, or deleting incorrect, incomplete, and inaccurate data. The data cleaning process can be done manually or automatically depending on the needs and complexity [15]. In data cleaning, several types of data problems need to be considered, such as missing values, outliers, duplicates, inconsistencies, and mislabels [16]. In this research dataset, data cleaning is done to overcome missing values and outliers in the features of bmi, smoking status, and avg glucose level. Handling missing data can be done with two approaches: ignoring or filling in missing values. In missing value filling, there is a single imputation procedure, which estimates the correct value for the missing value from the dataset by analyzing other responses and selecting the most likely response. The estimated value can be calculated using the available feature values' mean, median, or mode [17]. In the bmi feature, missing values are filled in using the average value. While in the smoking status feature, the "unknown" value will be changed to the mode value to fill the unknown value. Furthermore, outliers are handled on the avg glucose level and bmi features. For handling outliers avg glucose level uses an outlier factor of 0.5, and bmi uses an outlier factor of 1.0 due to differences in data quality, which can be seen in Figure 2.

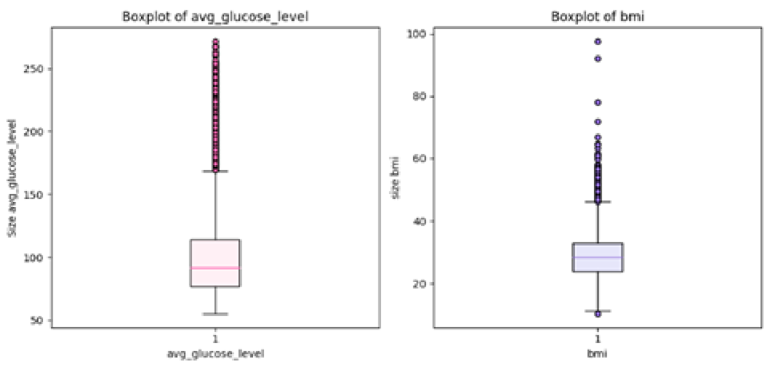

Figure 2. Data before handling outliers

Outlier handling aims to eliminate data considered an outlier in a dataset. The value removed is 332 with the total initial data of 5110, so the number after handling outliers is 4778. The data results after outlier handling can be seen in Figure 3.

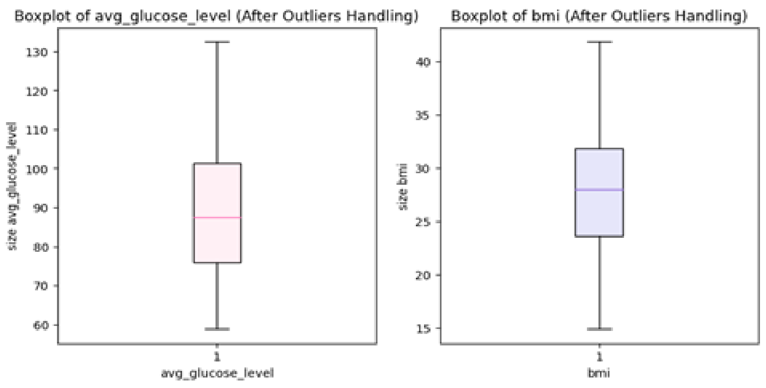

Figure 3. Data after handling outliers 2

2.3. Label Encoding

Label Encoding is the process of encoding labels by converting labels into quantitative form so machines can read them [18]. In label encoding, there is a label encoder. Chris Moffitt explains that label encoders are one of the encoding techniques in machine learning that are used to convert categorical and text data into numbers [19]. This technique encodes values between 0 and n-1, where n is the total number of classes of the attribute [20]. In this study, label encoding is performed using a label encoder to address features with binary values and discrete categories. Features such as gender, ever_married, work_type, residence_type, and smoking_status with categorical types are converted to numeric. These changes can be seen in Table 2 and Table 3.

Table 2. Before encoding

gender	**ever_married**	**Work_type**	**Residence_type**	**smoking_status**
Male	Yes	Private	Urban	formerly smoked
Female	Yes	Self-employed	Rural	never smoked
Male	Yes	Private	Rural	never smoked
Female	Yes	Private	Urban	smokes
Female	Yes	Self-employed	Rural	never smoked
...	...	...	...	...
Female	Yes	Private	Urban	never smoked
Female	Yes	Self-employed	Urban	never smoked
Female	Yes	Self-employed	Rural	never smoked
Male	Yes	Private	Rural	formerly smoked
Female	Yes	Govt_job	Urban	Unknown

Table 3. After encoding

gender	ever_married	Work_type	Residence_type	smoking_status
1	1	2	1	0
0	1	3	0	1
1	1	2	0	1
0	1	2	1	2
0	1	3	0	1
...	...	...	...	...
0	1	2	1	1
0	1	3	1	1
0	1	3	0	1
1	1	2	0	0
0	1	0	1	1

2.4. Data Normalization

Normalization is a scaling technique used in data preprocessing in machine learning to convert numerical values in a database to a uniform scale. The scale used in this data is min-max scalar because many algorithms depend on the distance between multiple points (samples). Normalization is not always necessary for every data set in a model, but normalization is usually required when the range of variables in the machine learning model has significant variation [21]. In this study, the range is narrowed by normalizing 0 to 1. The results of the normalized dataset values can be seen in Table 4.

Table 4. Data normalization

gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0.50	0.85	0.00	1.00	1.00	0.50	1.00	0.96	0.57	0.00	1.00
0.00	0.79	0.00	0.00	1.00	0.75	0.00	0.90	0.38	0.50	1.00
0.50	0.98	0.00	1.00	1.00	0.50	0.00	0.62	0.47	0.50	1.00
0.00	0.67	0.00	0.00	1.00	0.50	1.00	0.85	0.52	1.00	1.00
0.00	0.97	1.00	0.00	1.00	0.75	0.00	0.85	0.27	0.50	1.00
...	...	...	...	...	...	...	...	...	...	...
0.00	0.98	1.00	0.00	1.00	0.50	1.00	0.34	0.38	0.50	0.00
0.00	0.99	0.00	0.00	1.00	0.75	1.00	0.76	0.65	0.50	0.00
0.00	0.54	0.00	0.00	1.00	0.75	0.00	0.33	0.43	0.50	0.00
0.50	0.69	0.00	0.00	1.00	0.50	0.00	0.84	0.30	0.00	0.00
0.00	0.63	0.00	0.00	1.00	0.00	1.00	0.36	0.32	0.50	0.00

2.5. Feature Selection

In this research, the feature selection method used is a correlation matrix using the Pearson correlation coefficient approach. This approach is appropriate when both variables represent interval or ratio measurement scales [22]. The Pearson correlation coefficient is obtained by dividing the covariance value of two variables by the product of their standard deviations. Pearson correlation is useful for measuring the strength of the linear relationship between two continuous variables, such as X and Y, where the values are in the range [-1, +1]. As the coefficient approaches zero, the relationship between the two variables is less significant (closer to uncorrelated). The closer to -1 or +1, the stronger the relationship between the variables. A coefficient value of -1 indicates a perfect decreasing linear relationship, while +1 indicates a perfect increasing linear relationship (correlation). If variables X and Y are independent, then the Pearson correlation coefficient value will be zero [23].

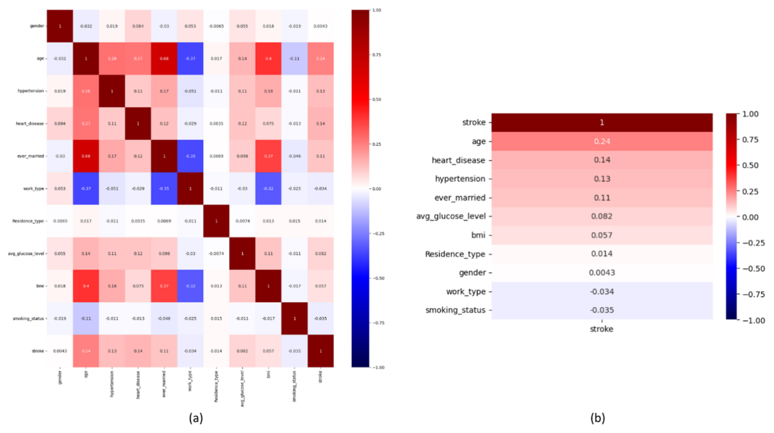

Figure 4. (a) Correlation matrix between features, and (b) correlation with target classq

The correlation matrix between features and correlation with the target has been sorted, as shown in Figure 4. The selected features are those with correlation coefficients close to +1 or in the range of 0 to 1. Therefore, the features used in this study are age, heart disease, hypertension, ever_married, avg_glucose_level, bmi, Residence_type, and gender.

2.6. Resampling

Distribution imbalance can affect classification results. One effective technique to overcome this problem is to use resampling methods. Resampling is divided into two categories, namely oversampling (adding samples to the minority class) and undersampling (reducing samples to the majority class) [24]. According to Weiss et al., undersampling risks losing important concepts due to data deletion. In addition, when the number of minority observations is limited, undersampling to achieve a balanced distribution results in a too small dataset, which may limit classification performance. SMOTE is a statistical technique that increases the number of minority samples in a dataset by generating new instances [25]. According to Pan, in SMOTE, adjacent instances in feature space are selected, a line is drawn between them, and then new samples are drawn at locations along the line [26]. In this study, an oversampling technique using SMOTE (Synthetic Minority Oversampling Technique) is used, where it is known that the initial number of samples in class 0 is 4542 and class 1 is 236, indicating a class imbalance. Therefore, resampling was carried out so that the number of samples in class 0 and class 1 became 4542. A comparison before and after resampling can be seen in Figure 5.

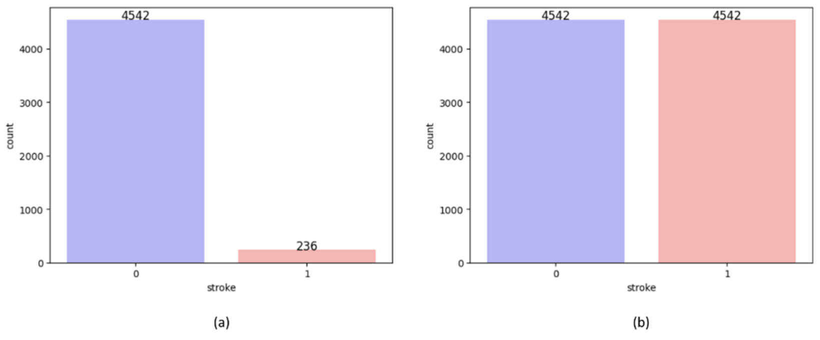

Figure 5. Number of samples for the dataset, (a) before resampling, and (b) after resampling

2.7. Data Splitting

Data splitting is a way to validate the model by dividing the data into two groups called training data and testing data [27]. In data splitting, the ratio used is 80:20, which means 80% of the data is used for training and 20% for testing. There is no rule to split the dataset, and the 80:20 split is based on the Pareto Principle. According to Isaac et al., the Pareto approach is one of the best techniques for data splitting in AI and ML studies [28].

2.8. Hyper Parameter Tuning GridSearch & RandomSearch

All machine learning models have a set of hyperparameters or arguments that the practitioner must specify. The best hyperparameter is subjective and differs for every dataset. Hyperparameter optimization is a technique that involves searching through a range of values to find a subset of results that achieve the best performance on a given dataset. Two popular techniques are used to perform hyperparameter optimization: GridSearch and RandomSearch. The parameter information and values for each classification model can be seen in Table 5.

Table 5. Parameters value

Classification Model	Parameter	ParameterValue	Best Parameter GridSearch	Best Parameter RandomSearch
Categorical Boosting	depth	[4, 6, 8]	6	8
	iterations	[100, 200, 300]	300	200
	12_leaf_reg	[1, 3, 5]	1	5
	learning_rate	[0.01, 0.1, 0.2]	0.2	0.1
Light Gradient Boosting	learning_rate	[0.01, 0.1, 0.2]	0.1	0.2
	max_depth	[4, 6, 8]	8	6
	n_estimators	[100, 200, 300]	300	300
	num_leaves	[20, 30, 40]	20	30
Histogram Gradient Boosting	learning_rate	[0.01, 0.1, 0.2]	0.2	0.2
	max_depth	[4, 6, 8]	8	6
	max_iter	[100, 200, 300]	300	300
	min_samples_leaf	[2, 4, 6]	4	4
Extreme Gradient Boosting	n_estimators	[100, 200, 300]	300	200
	max_depth	[4, 6, 8]	8	8
	colsample_bytree	[0.8, 0.9, 1.0]	0.8	1.0
	subsample	[0.8, 0.9, 1.0]	0.9	0.9
	learning_rate	[0.01, 0.1, 0.2]	0.1	0.2

GridSearch is a tuning technique used to obtain the optimal value of hyperparameters by searching through various combinations within a specified range [29]. While RandomSearch chooses a configuration randomly and repeats this process until the specified resources are exhausted [30].

2.9. Categorical Boosting

Categorical Boosting (CatBoost) is a gradient-boosting framework that uses binary decision trees as base predictors. CatBoost has two main differences from other boosting algorithms, namely the use of the concept of ordered boosting with random permutations to avoid overfitting and the use of a consistent splitting criterion across all nodes, resulting in a symmetric and fast-executing tree [31]. In addition, CatBoost also has the ability to handle the problems of gradient bias and prediction drift, thus improving the generalization ability and robustness of the algorithm [32].

2.10. Light Gradient Boosting

A Light Gradient Boosting Machine (LightGBM) is a fast and efficient framework of gradient boosting that relies on a decision tree algorithm. LightGBM was given the prefix ”Light” due to its high speed compared to other Gradient Boosting Decision Tree algorithms [33]. The main benefit of LightGBM is the dramatic speedup of the training algorithm, which, in many cases, results in more effective models. The algorithm is designed with the help of two new techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). OSS is a technique used to improve efficiency in model training. It works by ignoring most of the data instances with small gradients and using those with large enough gradients to estimate the information gain. This allows

LightGBM to estimate information gain accurately using a smaller dataset, thereby reducing the computational burden and speeding up model training. EFB is a technique used to handle feature density in datasets. EFB works by clustering features with exclusive relationships, thereby reducing the number of features without sacrificing their important information [34].

2.11. Histogram Gradient Boosting

Histogram Gradient Boosting (HGB) is an experimental implementation of Gradient Boosting Trees inspired by the workings of Light Gradient Boosting [35]. According to Bentejac et al., the histogram gradient boosting (HGB) algorithm is a variation of enhanced gradient boosting (GB), widely used for Machine Learning tasks related to classification and regression. According to Pedregosa et al., this algorithm was developed to solve the biggest drawback of the GB algorithm, which is the long time required to train large datasets [36]. Using histograms, HGB groups continuous data samples into a number of bins with constant values. This technique reduces the different values for each feature to a few small values. With this reduction, the decision tree can work with a smaller number of features. This results in a more efficient decision tree with less separation between nodes, thus improving and speeding up the tree implementation [37].

2.12. Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is developing a tree-based model using the ensemble gradient boosting method. XGBoost can handle common problems in a dataset, namely unbalanced data [38], so the results are more accurate. In other words, XGBoost has a big advantage, one of which is handling classes that rarely appear in the data or classes with a low frequency. The model can handle the missing values on its own. During training, the model learns whether missing values should be in the right or left node [39]. Another thing about XGBoost is that it can calculate feature importance, which shows which predictor columns (or variables) are more influential on the prediction outcome. XGBoost is a sequential learning algorithm combining weak learners to deliver an algorithm with improved or high predicting accuracy. At any stage of algorithm development; the model outcomes are weighed based on the outcomes of previous stage [40].

2.13. Evaluation

Performance testing is an important step in research to evaluate the accuracy of a model before it is applied [41]. This study uses Accuracy, Area Under the Curve (AUC), Confusion Matrix, and Execution Time. Accuracy is to find out how well the model makes predictions. The confusion matrix measures the extent to which the model correctly identifies several different classes and the extent to which there is overlap between these classes in the classification process [42]. Area Under the Curve (AUC) is used to calculate the performance of the model in distinguishing between positive and negative classes [43]. Execution Time to determine how much time is used in the process.

3. RESULT AND ANALYSIS

3.1. Comparison of Confusion Matrix

The following in Figure 6 are the test results based on the confusion matrix of each classification model, along with a combination of hyperparameter tuning. The objective of the model is to increase the values of True Positives (TP) and True Negatives (TN) while bringing the values of False Positives (FP) and False Negatives (FN) to zero. It can be seen that CatBoost and LightGBM, both using GridSearch and RandomSearch, have high TP and TN values, although there are slight differences. Both CatBoost and LightGBM have characteristics that can handle missing values efficiently during the training process. In addition, CatBoost includes built-in regularization techniques to reduce overfitting, which can be beneficial when dealing with complex datasets or limited amounts of data.

On the other hand, HGB has the most FN and FP values, but the difference in value is not too significant compared to others. HGB might be less robust when using noisy data than other algorithms. Noisy data points, outliers, or errors in the dataset can have a more pronounced impact on the model's performance. HGB is susceptible to overfitting, especially when the model complexity is high, or the dataset is small. Overfitting occurs when the model learns the training data too well, capturing noise and outliers that do not generalize well to unseen data.

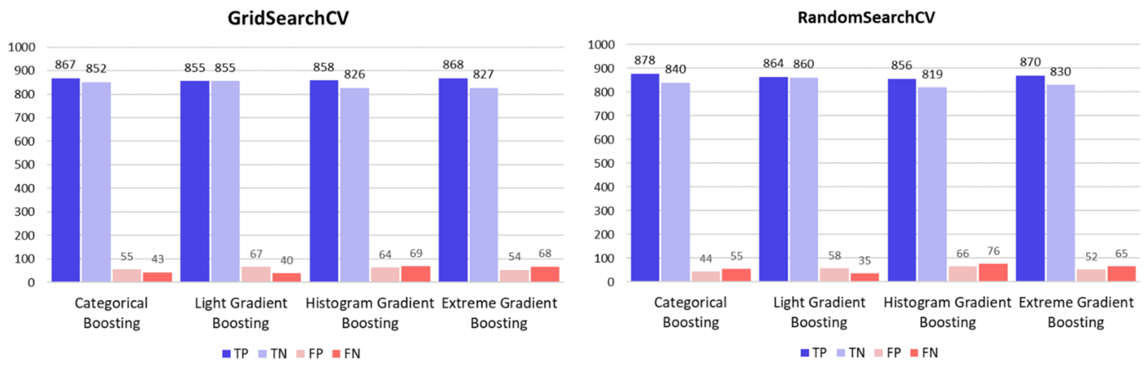

Figure 6. Comparison of confusion matrix in graph

3.2. Comparison of Evaluation Metrics and Time Execution

The findings of this research are the use of GridSearch and RandomSearch hyperparameters applied to the entire boosting classification model, which produces almost the same accuracy. Light Gradient Boosting also gets the same good results as Categorical Boosting with 95% because it has a regularization effect to prevent overfitting. However, the accuracy value when using GridSearch has a slight decrease in accuracy, precision, and recall values.

Combining the Light Gradient Boosting model with RandomSearch is better than GridSearch with a difference of 1%. However, in classifying other models, such as Histogram Gradient Boosting, GridSearch has a high value. In terms of performance, GridSearch and RandomSearch are not significantly different. n GridSearch, GridSearch creates a grid of all possible hyperparameter combinations and performs cross-validation on each combination to determine which hyperparameter set yields the best model performance. In contrast, RandomSearch performs random hyperparameter selection, which speeds up time and allows for the efficient discovery of a good parameter combination.

Table 6. Evaluation metrics

Classification Model	Accuracy (%)	Class	Precision (%)	Recall (%)	F1-Score (%)	AUC (%)
CatBoost + GridSearch	95	1	95	94	95	99
		0	94	95	95	
CatBoost + RandomSearch	95	1	94	95	95	99
		0	95	94	94	
LightGBM + GridSearch	94	1	96	93	94	99
		0	93	96	94	
LightGBM + RandomSearch	95	1	96	94	95	99
		0	94	96	95	
HGB + GridSearch	93	1	93	93	93	98
		0	93	92	93	
HGB + RandomSearch	92	1	92	93	92	98
		0	93	92	92	
XGBoost + GridSearch	93	1	93	94	93	99
		0	94	92	93	
XGBoost + RandomSearch	94	1	93	94	94	99
		0	94	93	93	

In this study, a time comparison was also carried out to determine how fast each algorithm and hyperparameter is processing the dataset used. Due to the ever-changing results, three trials were conducted. The results were concluded by looking at the pattern in the execution time results, where Light Gradient Boosting using GridSearch and RandomSearch has the fastest time in 3 trials. In contrast, Categorical Boosting has a time that is not as fast as Light Gradient Boosting but has better performance on the GridSearch Hyperparameter. The results of execution time per second can be seen in Tables 6, 7, and 8.

Table 7. Execution time 1 in sec

Classification Model	GridSearch	RandomSearch
Categorical Boosting	658.69	658.69
Light Gradient Boosting	94.29	31.21
Histogram Gradient Boosting	535.23	155.27
Extreme Gradient Boosting	1589.70	169.01

Table 8. Execution time 2 in sec

Classification Model	GridSearch	RandomSearch
Categorical Boosting	669.60	669.60
Light Gradient Boosting	128.38	45.30
Histogram Gradient Boosting	664.69	151.77
Extreme Gradient Boosting	1626.80	163.20

Table 9. Execution time 3 in sec

Classification Model	GridSearch	RandomSearch
Categorical Boosting	720.49	720.49
Light Gradient Boosting	98.96	34.34
Histogram Gradient Boosting	548.09	143.91
Extreme Gradient Boosting	1277.04	149.31

RandomSearch emphasizes exploration of the hyperparameter space, while GridSearch focuses more on exploitation. GridSearch can be computationally expensive, and this is proven by the average time required for the training process compared to RandomSearch, especially when the search space is large.

3.3. Comparison with Other Methods in Previous Research

In this journal, the Categorical Boosting method managed to achieve an accuracy rate of 95%. The results of this study are in line with or supported by ”Exploratory Data Analysis and Machine Learning Algorithms for Classifying Stroke Diseases” by Prismaharadi Aji Riyantoko et al. In the journal, the K-Nearest Neighbors method was tested, and an accuracy rate of 94% was obtained. The statement shows that Categorical Boosting significantly surpasses the performance of the KNN algorithm used in previous studies. KNN is prone to overfitting, mainly due to sensitivity to the parameter k's value. Although feature selection and dimensionality reduction techniques can be applied to prevent overfitting, an inappropriate value of k may result in a either too complex or too simple model. Catboost efficiently utilizes categorical features through ordered boosting to help overcome the overfitting problem that often occurs in the KNN algorithm. The comparison results with other methods in previous research can be seen in Table 10.

Table 10. Comparison with previous research

Research	Accuracy
K-Nearest Neighbors [11]	94%
Categorical Boosting	95%

4. CONCLUSION

This study uses four algorithms, which are derivatives of Gradient Boosting. Two hyperparameters are used on each model to determine the model's performance that matches this dataset. Several preprocessing stages are carried out during the process, and then the data will be divided into training and testing data. The next step is to evaluate the model's performance using the confusion matrix and AUC graph. The confusion matrix provides a more detailed picture of the performance of each model by calculating True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) from the prediction results. With the information from the confusion matrix, we can measure each model's accuracy, precision, recall, and f1-score to evaluate the extent to which the model is effective in predicting stroke cases. In this study, Categorical Boosting with GridSearch CV and RandomSearch CV hyperparameters produced the greatest accuracy of 95% and AUC of 99%, and Light Gradient Boosting with RandomSearch CV hyperparameter tuning has the potential to improve the efficiency and accuracy of stroke prediction models as it has the fastest execution time and similar results to Categorical Boosting. These results show that they better predict stroke cases than other models.

In future work, we intend to add pre- and post-processing hyper-parameters that may increase the complexity of the tuning problem being addressed and expand the experiments to include harder problems (datasets) and other ML algorithms, mainly those with a larger number of hyper-parameters. Consider combining both approaches, starting with RandomSearch to narrow down the search space and then using GridSearch for finer-grained exploration around promising regions.

5. ACKNOWLEDGEMENTS

This manuscript is based on research supported by the Universitas Muhammadiyah Malang. The authors would also express their gratitude to the UMM Informatics Laboratory, which has supported the implementation of this research.

6. DECLARATIONS

AUTHOR CONTIBUTION

Dela Ananda Setyarini: Conceptualization, Methodology, Validation, Formal analysis, Resources, Writing original draft, Visualization. Agnes Ayu Maharani Dyah Gayatri: Conceptualization, Methodology, Software, Validation, Resources, Data curation. Christian Sri Kusuma Aditya: Software, Formal analysis, Investigation, Writing, Review & Editing. Didih Rizki Chandranegara: Software, Formal analysis, Investigation.

FUNDING STATEMENT

The cost of publishing an accredited national journal entitled Comparison of Boosting Classifier with Hyperparameter Tuning in Stroke Predictive Model is not supported by the private-public or commercial.

COMPETING INTEREST

All authors reported this article with no competing financial interests or personal conflicts.

REFERENCES

[1] H. Al-Zubaidi, M. Dweik, and A. Al-Mousa, "Stroke Prediction Using Machine Learning Classification Methods," in *2022 International Arab Conference on Information Technology (ACIT)*. IEEE, nov 2022, pp. 1–8, https://doi.org/10.1109/ACIT57182.2022.10022050.

[2] P. Govindarajan, R. K. Soundarapandian, A. H. Gandomi, R. Patan, P. Jayaraman, and R. Manikandan, "Classification of stroke disease using machine learning algorithms," *Neural Computing and Applications*, vol. 32, no. 3, pp. 817–828, feb 2020, https://doi.org/10.1007/s00521-019-04041-y.

[3] M. GholamAzad, J. Pourmahmoud, A. R. Atashi, M. Farhoudi, and R. Deljavan Anvari, "Predicting of Stroke Risk Based On Clinical Symptoms Using the Logistic Regression Method," *Int. J. Industrial Mathematics*, vol. 14, no. 2, https://doi.org/10.30495/ijim.2022.64325.1559.

[4] N. A. Arifuddin, I. W. R. Pinastawa, N. Anugraha, and M. G. Pradana, "Classification of Stroke Opportunities with Neural Network and K-Nearest Neighbor Approaches," *SinkrOn*, vol. 8, no. 2, pp. 688–693, apr 2023, https://doi.org/10.33395/sinkron.v8i2.12228.

[5] B. Imran, "Classification of stroke patients using data mining with AdaBoost , Decision Tree and Random Forest models," no. December, 2022, https://doi.org/10.33096/ilkom.v14i3.1328.

[6] T. R. G, S. Bhattacharya, P. K. R. Maddikunta, S. Hakak, W. Z. Khan, A. K. Bashir, A. Jolfaei, and U. Tariq, "Antlion re-sampling based deep neural network model for classification of imbalanced multimodal stroke dataset," *Multimedia Tools and Applications*, vol. 81, no. 29, pp. 41 429–41 453, dec 2022, https://doi.org/10.1007/s11042-020-09988-y.

[7] Sydney Caulfeild, Sarah Pak, Nathanael Yao, and Hoz Rashid, "Stroke Prediction," *Journal of Mechanics Engineering and Automation*, vol. 11, no. 6, dec 2021, https://doi.org/10.17265/2159-5275/2021.06.004.

[8] H. Saleh, S. F. Abd-El Ghany, E. M. G. Younis, N. Omran, H. Ahmed, E. M. G. Youn, N. F. Omran, and A. A. Ali, "Stroke Prediction using Distributed Machine Learning Based on Apache Spark Hybrid Online-Offline Ranking based on Metaheuristic Methods with Click Models View project My Articles After PhD for Promotion to Associated Professor Job View project Stroke Prediction using Distributed Machine Learning Based on Apache Spark," *International Journal of Advanced Science and Technology*, vol. 28, no. 15, pp. 89–97, 2019, https://doi.org/10.13140/RG.2.2.13478.68162.

[9] T. Tazin, M. N. Alam, N. N. Dola, M. S. Bari, S. Bourouis, and M. Monirujjaman Khan, "Stroke Disease Detection and Prediction Using Robust Learning Approaches," *Journal of Healthcare Engineering*, vol. 2021, pp. 1–12, nov 2021, https://doi.org/10.1155/2021/7633381.

[10] M. U. Emon, M. S. Keya, T. I. Meghla, and M. Rahman, "Performance Analysis of Machine Learning Approaches in Stroke Prediction," no. January, 2021, https://doi.org/10.1109/ICECA49313.2020.9297525.

[11] P. A. Riyantoko, T. M. Fahrudin, K. M. Hindrayani, and M. Idhom, "Exploratory Data Analysis and Machin e Learning Algorithms to Classifying Stroke Disease," *IJCONSIST JOURNALS*, vol. 2, no. 02, pp. 77–82, jun 2021, https://doi.org/10.33005/ijconsist.v2i02.49.

[12] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi, and M. Asadpour, "Boosting methods for multi-class imbalanced data classification: an experimental review," *Journal of Big Data*, vol. 7, no. 1, p. 70, dec 2020, https://doi.org/10.1186/s40537-020-00349-y.

[13] F. Alzamzami, M. Hoda, and A. El Saddik, "Light Gradient Boosting Machine for General Sentiment Classification on Short Texts: A Comparative Evaluation," *IEEE Access*, vol. 8, pp. 101 840–101 858, 2020, https://doi.org/10.1109/ACCESS.2020.2997330.

[14] M. Robben, M. S. Nasr, A. Das, M. Huber, J. Jaworski, J. Weidanz, J. Luber, M. Sadegh Nasr, M. Hu-Ber, and J. . Luber, "Selection of an Ideal Machine Learning Framework for Predicting Perturbation Effects on Network Topology of Bacterial KEGG Pathways," *The 13th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics, August 07â10, 2022, Chicago, IL*, vol. 1, https://doi.org/10.1101/2022.07.21.501034.

[15] M. A. S. Yudono, A. D. W. M. Sidik, I. H. Kusumah, A. Suryana, A. P. Junfithrana, A. Nugraha, M. Artiyasa, E. Edwinanto, and Y. Imamulhak, "Bitcoin USD Closing Price (BTC-USD) Comparison Using Simple Moving Average And Radial Basis Function Neural Network Methods," *FIDELITY : Jurnal Teknik Elektro*, vol. 4, no. 2, pp. 29–34, may 2022, https://doi.org/10.52005/fidelity.v4i2.74.

[16] P. Li, X. Rao, J. Blase, Y. Zhang, X. Chu, and C. Zhang, "CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Tasks," apr 2019.

[17] W. M. Hameed and N. A. Ali, "Comparison of Seventeen Missing Value Imputation Techniques," *Journal of Hunan University Natural Sciences*, vol. 49, no. 7, pp. 26–36, jul 2022, https://doi.org/10.55463/issn.1674-2974.49.7.4.

[18] A. K. Venkitaraman and V. S. R. Kosuru, "Hybrid Deep Learning Mechanism for Charging Control and Management of Electric Vehicles," *European Journal of Electrical Engineering and Computer Science*, vol. 7, no. 1, pp. 38–46, jan 2023, https://doi.org/10.24018/ejece.2023.7.1.485.

[19] C. L. Krishna and P. V. S. Reddy, "Principal Component Analysis on Mixed Data For Deep Neural Network Classifier in Banking System," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 5, pp. 129–134, may 2019, https://doi.org/10.26438/ijcse/v7i5.129134.

[20] A. Srinivasan and J. Basilio, "Predicting revenue generation in an online retail website using machine learning algorithm MSc Research Project in Data Analytics."

[21] S. Srivastava, R. Kumar Yadav, V. Narayan, and P. K. Mall, "An Ensemble Learning Approach For Chronic Kidney Disease Classification," *Journal of Pharmaceutical Negative Results* , vol. 13, https://doi.org/10.47750/pnr.2022.13.S10.279.

[22] E. Isaac and E. Chikweru, "Test for Significance of Pearson's Correlation Coefficient ( )."

[23] Y. Xia, "Correlation and association analyses in microbiome study integrating multiomics in health and disease," 2020, pp. 309–491, https://doi.org/10.1016/bs.pmbts.2020.04.003.

[24] M. A. Janicka, M. A. Lango, and J. E. Stefanowski, "Using Information on Class Interrelations to Improve Classification of Multiclass Imbalanced Data: A New Resampling Algorithm," vol. 29, no. 4, pp. 769–781, 2019, https://doi.org/10.2478/amcs-2019-0057.

[25] R. Ghorbani and R. Ghousi, "Comparing Different Resampling Methods in Predicting Students' Performance Using Machine Learning Techniques," *IEEE Access*, vol. 8, pp. 67 899–67 911, 2020, https://doi.org/10.1109/ACCESS.2020.2986809.

[26] K. Aditya, G. Wasis Wicaksono, H. Abi Sarwan Heryawan, and C. Sri Kusuma Aditya, "Sentiment Analysis of the 2024 Presidential Candidates Using SMOTE and Long Short Term Memory," vol. 8, no. 2, pp. 279–286, 2023, https://doi.org/10.32493/informatika.v8i2.32210.

[27] V. R. Joseph, "Optimal ratio for data splitting," *Statistical Analysis and Data Mining*, vol. 15, no. 4, pp. 531–538, aug 2022, https://doi.org/10.1002/sam.11583.

[28] I. K. Nti, A. F. Adekoya, B. A. Weyori, and O. Nyarko-Boateng, "Applications of artificial intelligence in engineering and manufacturing: a systematic review," *Journal of Intelligent Manufacturing*, vol. 33, no. 6, pp. 1581–1601, aug 2022, https://doi.org/10.1007/s10845-021-01771-6.

[29] T. T. Ngoc, L. V. Dai, and C. M. Thuyen, "Support Vector Regression based on Grid Search method of Hyperparameters for Load Forecasting."

[30] L. Zahedi, F. G. Mohammadi, S. Rezapour, M. W. Ohland, and M. H. Amini, "Search Algorithms for Automated Hyper-Parameter Tuning," apr 2021.

[31] E. S. Solano, P. Dehghanian, and C. M. Affonso, "Solar Radiation Forecasting Using Machine Learning and Ensemble Feature Selection," *Energies*, vol. 15, no. 19, p. 7049, sep 2022, https://doi.org/10.3390/en15197049.

[32] F. Zhou, H. Pan, Z. Gao, X. Huang, G. Qian, Y. Zhu, and F. Xiao, "Fire Prediction Based on CatBoost Algorithm," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–9, jul 2021, https://doi.org/10.1155/2021/1929137.

[33] L. Sari, A. Romadloni, R. Lityaningrum, and H. D. Hastuti, "Implementation of LightGBM and Random Forest in Potential Customer Classification," *TIERS Information Technology Journal*, vol. 4, no. 1, pp. 43–55, jun 2023, https://doi.org/10.38043/tiers.v4i1.4355.

[34] D. D. Rufo, T. G. Debelee, A. Ibenthal, and W. G. Negera, "Diagnosis of Diabetes Mellitus Using Gradient Boosting Machine (LightGBM)," *Diagnostics*, vol. 11, no. 9, p. 1714, sep 2021, https://doi.org/10.3390/diagnostics11091714.

[35] J. Khiari and C. Olaverri-Monreal, "Boosting Algorithms for Delivery Time Prediction in Transportation Logistics," in *IEEE International Conference on Data Mining Workshops, ICDMW*, vol. 2020-November. IEEE Computer Society, nov 2020, pp. 251–258, https://doi.org/10.1109/ICDMW51313.2020.00043.

[36] G. Marvin, L. Grbčić, S. Družeta, and L. Kranjčević, "Water distribution network leak localization with histogram-based gradient boosting," *Journal of Hydroinformatics*, vol. 25, no. 3, pp. 663–684, may 2023, https://doi.org/10.2166/hydro.2023.102.

[37] M. Ibrahim, H. Abdelraouf, K. M. Amin, and N. Semary, "International Journal of Computers and Information (IJCI) Keystroke dynamics based user authentication using Histogram Gradient Boosting."

[38] R. D. Abdu-Aljabar and O. A. Awad, "A Comparative analysis study of lung cancer detection and relapse prediction using XGBoost classifier," *IOP Conference Series: Materials Science and Engineering*, vol. 1076, no. 1, p. 012048, feb 2021, https://doi.org/10.1088/1757-899X/1076/1/012048.

[39] G. Abdurrahman and M. Sintawati, "Implementation of xgboost for classification of parkinson's disease," in *Journal of Physics: Conference Series*, vol. 1538, no. 1. Institute of Physics Publishing, jun 2020, https://doi.org/10.1088/1742-6596/1538/1/012024.

[40] A. Shebl, D. Abriha, A. S. Fahil, H. A. El-Dokouny, A. A. Elrasheed, and Á. Csámer, "PRISMA hyperspectral data for lithological mapping in the Egyptian Eastern Desert: Evaluating the support vector machine, random forest, and XG boost machine learning algorithms," *Ore Geology Reviews*, vol. 161, p. 105652, oct 2023, https://doi.org/10.1016/j.oregeorev.2023.105652.

[41] E. S. Theel, J. Harring, H. Hilgart, and D. Granger, "Performance Characteristics of Four High-Throughput Immunoassays for Detection of IgG Antibodies against SARS-CoV-2," 2020.

[42] M. Heydarian, T. E. Doyle, and R. Samavi, "MLCM: Multi-Label Confusion Matrix," *IEEE Access*, vol. 10, pp. 19 083–19 095, 2022, https://doi.org/10.1109/ACCESS.2022.3151048.

[43] B. He, D. Dong, Y. She, C. Zhou, M. Fang, Y. Zhu, H. Zhang, Z. Huang, T. Jiang, J. Tian, and C. Chen, "Predicting response to immunotherapy in advanced non-small-cell lung cancer using tumor mutational burden radiomic biomarker," *Journal for ImmunoTherapy of Cancer*, vol. 8, no. 2, jul 2020, https://doi.org/10.1136/jitc-2020-000550.