Mathematical Biosciences and Engineering

MBE, 23(12): 3470–3498.
DOI: 10.3934/mbe.2026119
Received: 22 August 2026
Revised: 28 September 2026
Accepted: 21 October 2026
Published: 19 November 2026

https://www.aimspress.com/journal/MBE

Research article

Hypertension classification using a balanced bagging decision tree optimized by Latin hypercube sampling–grey wolf optimization

Farikhin[1,*], Muhammad Sam'an[2] and Beta Noranita[3]

[1] Department of Mathematics, Diponegoro University, Semarang, Indonesia
[2] Department of Informatics, Muhammadiyah Semarang University, Semarang, Indonesia
[3] Department of Informatics, Diponegoro University, Semarang, Indonesia

* **Correspondence:** Email: farikhin@live.undip.ac.id; Tel: +6282133369088.

Abstract: Hypertension classification using machine learning is challenging when class distributions are substantially imbalanced because optimization based solely on overall accuracy may favor the majority class and increase false-negative predictions. This study evaluates a Latin hypercube sampling, grey wolf optimization, and balanced bagging decision tree framework for hypertension classification using 48,139 observations with 18 leakage-free predictors, comprising 83.61% non-hypertension and 16.39% hypertension cases. The study investigates five optimization objectives, namely accuracy, balanced accuracy, F1-score, recall, and a screening-oriented weighted objective, together with a Pareto-based multi-objective formulation and decision threshold analysis. The results show that the fitness function substantially affects the sensitivity and specificity trade-off. Accuracy-oriented optimization favored high specificity but produced low recall, whereas F1-score, balanced accuracy, recall, and screening oriented objectives shifted the search toward configurations with greater sensitivity to hypertension cases. The F1-score-based configuration provided a comparatively balanced error profile, while recall-based optimization achieved the highest sensitivity. A threshold of 0.45 further reduced false negatives from 966 to 220, corresponding to an approximately 77.2% reduction in missed hypertension cases relative to the accuracy-oriented configuration. Comparisons with imbalance-aware and class-weighted baseline models showed that LHS-GWO-BBC-DT did not universally outperform all alternatives. These findings demonstrate that fitness function design and threshold selection are critical components of hyperparameter optimization for imbalanced hypertension classification and should be evaluated together with accuracy, recall, specificity, PR-AUC, and other class-sensitive metrics.

Keywords: hypertension; balanced bagging classifier; decision tree; grey wolf optimization; Latin hypercube sampling; hyperparameter optimization

1. Introduction

Hypertension remains a major public health concern in Indonesia. According to the 2023 Indonesian Health Survey, the prevalence of hypertension based on direct blood pressure measurements was 10.7% among individuals aged 18–24 years and 17.4% among those aged 25–34 years [1]. In the same age groups, the prevalence of physician-diagnosed hypertension was 0.4% and 1.8%, respectively [1]. At the national level, the prevalence of hypertension among Indonesians aged 18 years and older was 30.8% based on blood pressure measurements [2]. Data from the Free Health Checkup program further showed that approximately one in five participants was identified as having hypertension among more than 8.2 million individuals examined as of June 12, 2025 [3]. These findings support the need for reliable approaches to identify individuals with hypertension and to support screening and preventive assessment.

The increasing availability of demographic, anthropometric, behavioral, and health-related data has encouraged the use of machine learning for hypertension classification and risk assessment. Decision tree (DT), random forest (RF), gradient boosting (GB), and logistic regression (LR) have been compared using IFLS-5 data, with LR achieving an AUC of 0.829, an accuracy of 89.6%, a recall of 0.896, a precision of 0.878, and an F1-score of 0.877 [4]. RF, GB, extreme gradient boosting (XGBoost), and LR have also been evaluated using non-laboratory risk factors from a large Indonesian population, where XGBoost achieved a sensitivity of 0.97 and an AUC of 0.75 in external validation [5]. Other studies have applied LR, RF, GB, and support vector machine models to hypertension and undiagnosed hypertension classification [6], while XGBoost combined with SHapley additive exPlanations has been used to analyze the coexistence of diabetes and hypertension [7]. Naïve Bayes, artificial neural networks, LR, and DT have also been applied to predict blood-pressure-related outcomes [8]. These studies show that machine-learning performance in hypertension-related tasks varies according to the predictors, class distribution, model architecture, and evaluation protocol.

Further developments in hypertension modeling have incorporated feature selection, class-imbalance handling, ensemble learning, interpretability, and hyperparameter optimization. Machine-learning algorithms combined with least absolute shrinkage and selection operator-based feature selection have been used to predict stroke risk among patients with hypertension [9]. LR, RF, and GB have been evaluated using medication, demographic, and treatment-related variables [10], while RF and LR have been applied to hypertension prediction using metabolic profiles and demographic characteristics [11]. More comprehensive frameworks have combined random oversampling, multiple feature-selection methods, ensemble classifiers, and hyperparameter search strategies including random search, grid search, Bayesian search, and genetic algorithms [12]. RF, light gradient boosting machine, XGBoost, GB, and k-nearest neighbors have also been compared for hypertension-risk assessment [13].

Decision trees are widely used for classification because its rule-based structure is interpretable and can represent nonlinear relationships among predictors [14]. However, DT performance is sensitive to class imbalance. When the majority class dominates the training data, the tree-construction process can favor decision boundaries associated with the majority class and provide weaker

representation of minority-class patterns [15]. Conventional approaches to class imbalance include oversampling and undersampling [16]. Oversampling increases minority-class representation but can introduce replicated or synthetic observations, whereas undersampling can discard potentially informative majority-class observations [17]. The balanced bagging classifier (BBC) provides an alternative ensemble-based strategy by constructing multiple base learners from independently balanced subsets of the training data and aggregating their predictions [18]. Nevertheless, BBC performance still depends on the configuration of ensemble- and tree-level hyperparameters, including the number of estimators, sample and feature proportions, sampling strategy, maximum tree depth, minimum numbers of samples required for node splitting and leaf formation, and pruning parameters [19].

The dependence of BBC on multiple hyperparameters motivates the use of search methods that can evaluate combinations of continuous, integer, Boolean, and categorical parameters within a predefined search space [20]. Grey wolf optimization (GWO) is a population-based metaheuristic inspired by the social hierarchy and hunting behavior of grey wolves [21]. In hyperparameter optimization, each wolf represents a candidate model configuration, and its fitness is calculated from the validation performance of the corresponding model [22]. The three leading solutions, conventionally denoted as alpha, beta, and delta, guide the remaining candidate solutions during the search process [23]. GWO has been used to optimize DT hyperparameters in diabetes classification [24], fraud detection [25], brain-tumor classification [26], stroke prediction [27], and engineering prediction problems [28]. These studies establish the feasibility of GWO for DT hyperparameter tuning, but they do not eliminate the need to examine how initialization strategy and fitness-function design affect the resulting model under class imbalance.

One factor that affects GWO search behavior is the distribution of the initial population. Standard GWO commonly initializes candidate positions through random sampling [29]. Random initialization can produce uneven coverage of a multidimensional search space, leaving some regions less represented during the early search stages [21,30]. Zhu et al. [31] employed Latin hypercube sampling (LHS) with grey wolf optimization (GWO) to enhance the initialization and performance of populations in inspection-node deployment. Shields and Zhang [32] improved LHS by incorporating partially stratified sampling, which boosts sampling efficiency and reduces variance from variable interactions. Collectively, these studies demonstrate that stratified sampling is crucial for achieving a more even distribution of candidate solutions and enhancing optimization performance. In this study, LHS is used only as an initialization strategy to provide a more systematically distributed starting population across the 12-dimensional BBC-DT hyperparameter space. It is not treated as a new optimization algorithm. Its contribution is evaluated through direct comparison with conventional randomly initialized GWO under the same search space and computational budget.

A second and more important issue concerns the fitness function used during hyperparameter optimization. In an imbalanced dataset, an accuracy-based objective does not explicitly account for minority-class sensitivity. This issue is relevant to the present dataset, in which 83.61% of observations belong to the non-hypertension class and 16.39% belong to the hypertension class. Under this distribution, an optimization procedure can favor configurations that correctly classify a large proportion of non-hypertension cases while failing to identify a substantial number of hypertension cases. For a screening application, this behavior is undesirable because a false-negative prediction represents an individual with hypertension who is not identified by the model for further assessment. Therefore, the optimization objective should reflect the intended screening function rather than relying solely on overall accuracy. Balanced accuracy, F1-score, recall, MCC, and PR-AUC provide

complementary measures for evaluating minority-class performance and class-balanced discrimination.

The reviewed literature provides limited direct evidence on how alternative fitness functions affect the operating characteristics of the same metaheuristic-optimized classifier when the classifier structure, search space, population size, and computational budget are held constant. In particular, direct comparisons among accuracy, balanced-accuracy, F1-score, recall, and multi-objective criteria remain uncommon in the studies considered here. In addition, the effect of LHS initialization cannot be isolated reliably without direct comparison against randomly initialized GWO across repeated stochastic runs. A single train–validation–test experiment also provides limited evidence regarding the generalization of a high-dimensional metaheuristic hyperparameter search. These issues motivate a revised evaluation framework that separates the effects of fitness-function design, initialization strategy, stochastic variability, decision threshold, and out-of-sample validation.

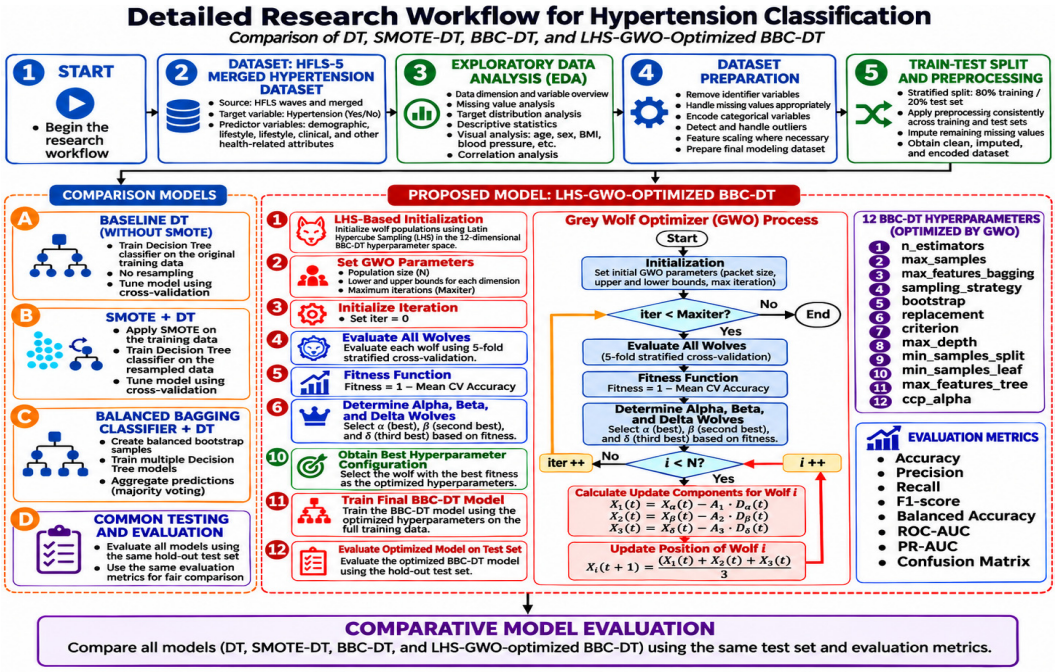

Figure 1. Workflow of the proposed algorithm.

Based on these considerations, this study investigates class-sensitive hyperparameter optimization of a balanced bagging decision tree for hypertension screening using IFLS-5 data. The study does not claim the development of a new metaheuristic algorithm. Instead, its contribution lies in systematically examining how optimization objectives and initialization strategies affect BBC-DT performance under substantial class imbalance. The overall research workflow, including data preparation, model development, hyperparameter optimization, and comparative evaluation, is illustrated in Figure 1. Specifically, the study: (1) compares accuracy, balanced accuracy, F1-score, recall, and a screening-oriented weighted criterion as alternative fitness functions for LHS-GWO; (2) evaluates a Pareto-based multi-objective formulation that simultaneously considers sensitivity and specificity; (3) directly compares conventional randomly initialized GWO and LHS-initialized GWO under an identical computational budget; (4) compares the optimized BBC-DT framework with established imbalance-aware classifiers, including AdaBoost, EasyEnsemble, RUSBoost, class-

weighted XGBoost, and class-weighted LightGBM, as well as alternative hyperparameter search methods including random search, genetic algorithm, and particle swarm optimization; and (5) assesses performance stability and generalization using repeated stochastic optimization, repeated stratified cross-validation, threshold sensitivity analysis, bootstrap confidence intervals, statistical significance testing, and nested cross-validation. Through this design, the study evaluates how the choice of optimization objective and initialization strategy affects the sensitivity–specificity trade-off of BBC-DT, with particular emphasis on hypertension-case detection in a screening context.

2. Materials and methods

2.1. Dataset

The dataset used in this study was derived from the fifth wave of the Indonesia family life survey (IFLS-5). The merged dataset contained 48,139 observations and 62 variables. The target variable, hypertension_status, was binary, with label 0 representing non-hypertension and label 1 representing hypertension. A total of 40,250 observations (83.61%) belonged to the non-hypertension class, whereas 7889 observations (16.39%) belonged to the hypertension class. This distribution corresponded to an approximate class ratio of 5.10:1 and indicated substantial class imbalance. After data preparation, 18 predictor variables and one target variable were retained for model development. The dataset was divided using a stratified 80:20 train–test split with a random state of 42. This procedure produced 38,511 observations for model development and 9628 observations for held-out testing. The test set was kept separate from hyperparameter optimization and model selection and was used only for final evaluation.

2.2. Exploratory data analysis

Exploratory data analysis (EDA) was conducted before model development to characterize the distribution of the target variable, identify variables with substantial missingness, examine hypertension proportions across body mass index (BMI) categories, and assess monotonic relationships among the principal study variables. The analysis was descriptive and was used to support subsequent data-preparation decisions rather than to select the final predictive model.

Figure 2(a) illustrates the distribution of hypertension status across the 48,139 observations. Of these, 40,250 (83.61%) were classified as non-hypertensive, while 7889 (16.39%) were classified as hypertensive. Thus, the non-hypertension class was approximately 5.10 times larger than the hypertension class. This distribution confirmed substantial class imbalance and provided the methodological basis for evaluating imbalance-aware learning strategies rather than relying solely on conventional overall accuracy.

Figure 2(b) summarizes the percentage of missing values across the selected variables. Several variables exhibited extensive missingness. In particular, reason, whr, hip_circumference, waist_circumference, and us05a contained more than 70% missing values. These variables were therefore excluded from the modeling dataset before classifier development. The missing-value inspection was used to distinguish variables that could reasonably be retained and imputed from variables whose information content was too limited for the present analysis.

Figure 2(c) shows the proportion of hypertension across BMI categories. The proportion of

hypertensive observations was higher in the overweight and obesity categories than in the underweight and normal-weight categories. This pattern was used descriptively to characterize the relationship between body-size status and hypertension occurrence in the study population. BMI was therefore retained as a predictor in the subsequent modeling stage. The EDA was not used to infer a causal relationship between BMI and hypertension.

Figure 2(d) presents the Spearman correlation matrix for the principal numerical variables. Positive correlations were observed between hypertension_status and mean systolic blood pressure, mean diastolic blood pressure, age, BMI, hypertension medication status, and body weight. Spearman correlation was used because it summarizes monotonic associations without requiring the variables to satisfy a linear-relationship assumption. These correlations were interpreted descriptively and were not used as evidence of causality.

An important distinction was made between variables used for descriptive exploration and variables retained for predictive modeling. Blood-pressure measurements, aggregated systolic and diastolic blood-pressure variables, variables used to construct the hypertension label, and hypertension_medication_flag could be included in EDA to describe their relationships with hypertension_status; however, they were excluded from the predictor set because they were directly involved in determining the target and would therefore introduce target leakage if used for model training. The modeling dataset was subsequently restricted to leakage-free predictors.

The EDA therefore served four specific purposes: it confirmed the degree of class imbalance, identified variables that required exclusion because of excessive missingness, described the distribution of hypertension across BMI categories, and distinguished descriptive variables from leakage-free predictors. These observations informed the preprocessing and model-development procedures described in the following section.

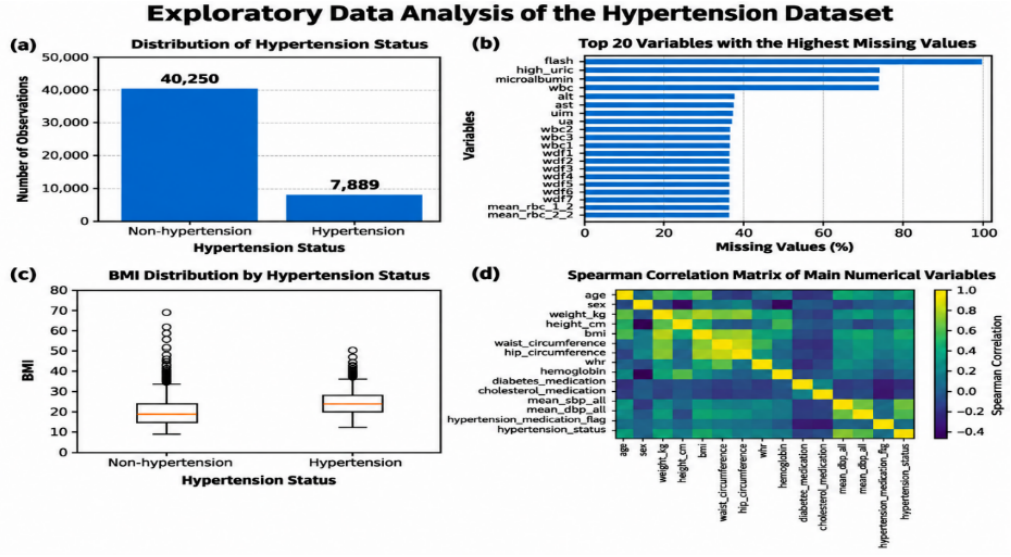

Figure 2. Exploratory data analysis of the hypertension dataset: (a) distribution of hypertension status, (b) percentage of missing values in selected variables, (c) proportion of hypertension across BMI categories, and (d) Spearman correlation matrix of the main study variables.

2.3. Data preparation

Data preparation was performed after exploratory data analysis and before model development. The procedure was designed to produce a leakage-free modeling dataset while preserving variables that remained suitable for classification. The original merged IFLS-5 dataset contained 48,139 observations and 62 variables. Preprocessing consisted of variable exclusion, missing-value handling, target–predictor separation, stratified data splitting, and training-only imputation.

Identifier variables, including pidlink, hhid14, hhid14_9, and pid14, were removed because they functioned only as record identifiers and were not intended to contribute predictive information. Administrative and interview-process variables related to interview timing, language, supervision, verification, and interview outcomes were also excluded. Constant variables were removed because they contained no discriminatory information, and variables with more than 70% missing values were excluded because their observed information was considered insufficient for the present modeling analysis.

Particular attention was given to the prevention of target leakage. Variables directly involved in constructing hypertension_status were excluded from the predictor set. These variables included the individual blood-pressure measurements us07a1, us07a2, us07b1, us07b2, us07c1, and us07c2; the related measurement-session variables us07ap, us07bp, and us07cp; aggregated blood-pressure variables mean_sbp_2_3, mean_dbp_2_3, mean_sbp_all, and mean_dbp_all; the label-construction variables sbp_for_label and dbp_for_label; and hypertension_medication_flag. Their exclusion was necessary because they were directly related to the definition of the target and could otherwise allow the model to infer hypertension_status from variables that effectively encoded the outcome itself. The manuscript already states that blood-pressure measurements, aggregated systolic and diastolic variables, labeling variables, and hypertension_medication_flag were removed to reduce leakage risk.

After applying the exclusion criteria, 18 predictor variables were retained for modeling: us18aa, us18ab, diabetes_medication, cholesterol_medication, weight_kg, height_cm, hemoglobin, us09a, us09b, us09c, us20a, us20b, us20c, us20d, sex, age, us13aa, and bmi. The target variable was hypertension_status, with 0 representing non-hypertension and 1 representing hypertension. This preprocessing stage therefore produced a final modeling matrix containing 18 predictors and one binary target.

The data were then divided using a stratified 80:20 train–test split with random_state = 42. Stratification preserved the original class proportions in both subsets and produced 38,511 training observations and 9628 held-out testing observations. The test set was not used during hyperparameter optimization, cross-validation, threshold selection, or model selection. It was retained exclusively for final out-of-sample evaluation.

Missing values in the retained predictors were handled using median imputation. Importantly, the imputation parameters were estimated using the training data only. The fitted imputer was subsequently applied to both the training and test sets, ensuring that information from the held-out test set did not influence the preprocessing stage. This procedure was consistent with the original methodology, which also fitted the imputer on the training partition before applying it to the testing data.

Feature scaling was not applied because the primary classification models were tree-based and therefore did not require standardization of predictor magnitudes. One-hot encoding was also not applied in the original modeling pipeline because the retained predictors were handled in their existing numerical representation. This decision was retained to maintain consistency with the original experiment and to avoid introducing additional transformations that were not part of the initial study.

2.4. Model development

Model development was organized into three levels: baseline classification, imbalance-aware ensemble learning, and hyperparameter optimization. A standalone decision tree was first trained as a conventional baseline. SMOTE–Tomek combined with a decision tree was then evaluated as a resampling-based imbalance-handling method. Balanced bagging with a decision tree was subsequently used as the principal ensemble classifier. Hyperparameter optimization of BBC-DT was performed using grey wolf optimization, with Latin hypercube sampling employed as an alternative population-initialization strategy. The revised experiments further extended this framework by evaluating class-sensitive and multi-objective fitness functions while preserving the same BBC-DT search space.

2.4.1. Decision tree

A decision tree (DT) was used as the base classifier. The model recursively divided the training data by selecting the split that produced the largest reduction in node impurity. For the Gini criterion, the impurity at node t was calculated as

$$\mathrm{G(t)} = 1 - \sum_{c=1}^{C} p_c^2, \tag{1}$$

where p_c^2 denotes the proportion of observations belonging to class c. For a candidate split, the reduction in impurity was calculated as

$$\Delta G = \mathrm{G(t)} - \frac{n_L}{n} G(t_L) - \frac{n_R}{n} G(t_R), \tag{2}$$

where n_L and n_R denote the numbers of observations in the left and right child nodes, respectively, and n denotes the number of observations in the parent node. The split with the largest reduction in impurity was selected. A baseline DT model was trained on the original imputed training data without applying a resampling procedure.

2.4.2. SMOTE-Tomek with decision tree

The dataset contained approximately 5.10 times more non-hypertensive observations than hypertensive observations. SMOTE–Tomek was therefore applied only to the training data before fitting the DT model. SMOTE generated minority-class observations by interpolating between an existing minority-class observation and one of its nearest minority-class neighbors

$$x_{new} = x_i + \lambda\left(x_j - x_i\right), \tag{3}$$

where x_i denotes a minority-class observation, x_j denotes one of its nearest minority-class neighbors, and $\lambda \in [0,1]$ is a randomly generated interpolation coefficient. Tomek links were subsequently applied to remove selected observations from different classes that were located close to one another near the class boundary.

SMOTE increased the number of minority-class observations by generating synthetic data through interpolation. These generated observations were not directly observed in the original dataset.

Therefore, a balanced bagging classifier was also developed as an alternative approach that balanced the training subsets without generating synthetic predictor values.

2.4.3. Balanced bagging classifier with decision tree

BBC generated several balanced subsets from the training data. Each balanced subset was used to train one DT model. Let M denote the total number of DT models in the ensemble, and let $\mathcal{D}_m^{bal}, m = 1,2,\cdots,M$ denote the balanced training subset used for the mth DT model. After training, the prediction produced by the mth DT model for an observation x was denoted by $f(x) \in [0,1]$. $f_m(x) = 0$ represents a non-hypertensive prediction, whereas $f_m(x) = 1$ represents a hypertensive prediction. Therefore, each DT model independently assigns observation x to one of the two classes. The final BBC prediction was determined by majority voting across all DT models:

$$\hat{y} = \arg\max_{c\in\{0,1\}} \sum_{m=1}^{M} I(f_m(x) = c), \tag{4}$$

where c represents the class label and $I(\cdot)$ is an indicator function defined as

$$I(f_m(x) = c) = f(x) = \begin{cases} 1, & if\ the\ m-th\ DT\ predicts\ class\ c, \\ x, & otherwise. \end{cases} \tag{5}$$

Thus, $\sum_{m=1}^{M} I(f_m(x) = c)$ counts the number of DT models that assign the observation to the class. The class receiving the largest number of votes was selected as the final BBC prediction. For example, if the BBC contained 100 DT models and 68 models predicted hypertension while 32 models predicted non-hypertension, the final prediction was hypertension. When class probabilities are required, let $p_m(c|x)$ denote the probability assigned to the class by the mth DT model. The final probability was calculated by averaging the probabilities from all DT models:

$$\hat{p}(y = c|x) = \frac{1}{M} \sum_{m=1}^{M} p_m\,(c|x). \tag{6}$$

The initial BBC-DT model used 100 DT estimators, automatic class balancing, bootstrap sampling, and no replacement during the internal balancing procedure. The hyperparameters controlling the bagging process and the structure of the DT models are presented in Table 1.

Table 1. Hyperparameters of the balanced bagging classifier with a decision tree base estimator.

No.	Hyperparameter	Data type	Description
1	n_estimators	Integer	Specifies the number of DT models included in the ensemble.
2	max_samples	Float	Specifies the proportion of training observations used to construct each base estimator.
3	max_features_bagging	Float	Specifies the proportion of predictor variables sampled for each base estimator.
4	sampling_strategy	Float	Defines the class ratio applied during the internal balancing procedure.
5	bootstrap	Boolean	Determines whether observations are sampled with replacement when each bagging subset is constructed.
6	replacement	Boolean	Determines whether observations are sampled with replacement during the class-balancing procedure.
7	criterion	Categorical	Specifies the impurity measure used to evaluate candidate node splits, including Gini impurity, entropy, and log loss.
8	max_depth	Integer	Specifies the maximum depth of each DT model.
9	min_samples_split	Integer	Specifies the minimum number of observations required to split an internal node.
10	min_samples_leaf	Integer	Specifies the minimum number of observations required in a terminal node.
11	max_features_tree	Float	Specifies the proportion of predictor variables considered when a node split is selected.
12	ccp_alpha	Float	Controls cost-complexity pruning and the structural complexity of each DT model.

2.4.4. Original grey wolf optimization

Grey wolf optimization (GWO) is a population-based metaheuristic that represents candidate solutions as grey wolves. The search process follows a leadership hierarchy consisting of the three best candidate solutions, denoted by α, β, and δ, while the remaining candidate solutions are denoted by ω. The wolf represents the best solution at the current iteration, followed by the and wolves.

Let the position of the-th wolf at iteration be expressed as $X_i(t) = [x_{i1}(t), x_{i2}(t), \cdots, x_{id}(t)]$, where d denotes the number of decision variables. The control parameter a decreases linearly from 2 to 0 according to

$$a(t) = 2 - 2\frac{t}{T}, \tag{7}$$

where t denotes the current iteration and T denotes the maximum number of iterations.

The coefficient vectors are calculated as

$$A_j = 2a(t)r_{1j} - a(t), \tag{8}$$

$$C_j = 2r_{2j}, j \quad = 1,2,3, \tag{9}$$

where r_{1j} and r_{2j} are random vectors generated within $[0,1]$.
The distances between the ith wolf and the three leading wolves are calculated as

$$D_\alpha = |C_1 \odot X_\alpha - X_1|, \tag{10}$$

$$D_\beta = \left|C_2 \odot X_\beta - X_2\right|, \tag{11}$$

$$D_\delta = |C_3 \odot X_\delta - X_3|, \tag{12}$$

where $\odot$ denotes element-wise multiplication.
Three candidate positions are generated based on the positions of the α, β, and δ wolves:

$$X_{i1} = X_\alpha - A_1 \odot D_\alpha, \tag{13}$$

$$X_{i2} = X_\beta - A_2 \odot D_\beta, \tag{14}$$

$$X_{i3} = X_\delta - A_3 \odot D_\delta. \tag{15}$$

The position of the ith wolf is updated by averaging the three candidate positions:

$$X_i(t+1) = \frac{X_{i1} + X_{i2} + X_{i3}}{3}. \tag{16}$$

This process is repeated until the maximum number of iterations is reached.

2.4.5. Latin hypercube sampling

In the original GWO, the initial positions of the wolves are commonly generated using random sampling. In this study, LHS was used to generate the initial population. For each search dimension, the normalized interval [0,1] was divided into N equal intervals, where N denotes the number of wolves. One value was then randomly selected from each interval. The sampled value can be written as

$$x_{ij} = \frac{k-1+r_{ij}}{N}, 0 \le x_{ij} \le 1, \tag{17}$$

where x_{ij} denotes the initial position of the ith wolf in the jth dimension, $k \in \{1,2,3,\cdots,N\}$ denotes the interval assigned to that wolf, and r_{ij} is a random value within [0,1]. For each dimension, the N intervals were randomly assigned to the N wolves without repetition. This procedure placed one initial solution in each interval of every search dimension. Compared with purely random initialization, LHS provided a more systematically distributed initial population over the normalized search space.

2.4.6. LHS-initialized grey wolf optimization

The LHS-GWO method replaced the random initialization stage of the original GWO with Latin hypercube sampling. Each wolf represented one candidate configuration of the BBC-DT

hyperparameters. Because 12 hyperparameters were optimized, each wolf was represented by a 12-dimensional position vector: $X_i(t) = [x_{i1}(t), x_{i2}(t), \cdots, x_{i12}(t)], 0 \leq x_{ij} \leq 1$, where i denotes the wolf index and j denotes the hyperparameter index. Each normalized position value was converted into the actual value of the corresponding hyperparameter using

$$p_{ij} = L_j + x_{ij}(U_j - L_j), \tag{18}$$

where p_{ij} denotes the actual value of the jth hyperparameter for the ith wolf, while U_j and L_j denote the lower and upper bounds of the jth hyperparameter, respectively. Integer-valued hyperparameters were rounded to the nearest valid integer, whereas Boolean and categorical hyperparameters were mapped to their available candidate values.

The optimization used 15 wolves and 25 iterations. At each iteration, every wolf represented one BBC-DT hyperparameter configuration and was evaluated using the defined fitness function. The candidate solutions were then ranked according to their fitness values. Because the optimization problem was formulated as a minimization problem, the three candidates with the lowest fitness values were assigned as the α, β, and δ wolves. The positions of the remaining wolves were subsequently updated using the original GWO equations based on the positions of α, β, and δ. After each update, all position values were restricted to $0 \leq x_{ij} \leq 1$, before being converted back into valid BBC-DT hyperparameter values. This procedure was repeated until 25 iterations were completed.

2.4.7. BBC-DT hyperparameter optimization using LHS-GWO

LHS-GWO was applied to optimize 12 hyperparameters associated with the balanced bagging classifier and its DT base estimator. The search space consisted of nine numerical hyperparameters and three categorical hyperparameters. Each wolf represented one candidate BBC-DT hyperparameter configuration. The position of the ith wolf was expressed as $X_i(t) = [x_{i1}(t), x_{i2}(t), \cdots, x_{i12}(t)]$, where each position value satisfied $0 \leq x_{ij} \leq 1$. The normalized position x_{ij} was converted into the actual value of the corresponding numerical hyperparameter using Eq (18). Integer-valued hyperparameters were rounded to the nearest valid integer. Boolean and categorical hyperparameters were mapped to one of the available candidate values. The search ranges used for the 12 BBC-DT hyperparameters are presented in Table 2.

Table 2. Hyperparameters of the balanced bagging classifier with a decision tree base estimator.

No.	Hyperparameter	Data type	Search range or candidate values
1	n_estimators	Integer	50–500
2	max_samples	Float	0.50–1.00
3	max_features_bagging	Float	0.50–1.00
4	sampling_strategy	Float	0.22–1.00
5	bootstrap	Boolean	False, True
6	replacement	Boolean	False, True
7	criterion	Categorical	gini, entropy, log_loss
8	max_depth	Integer	3–40
9	min_samples_split	Integer	2–50
10	min_samples_leaf	Integer	1–30
11	max_features_tree	Float	0.30–1.00
12	ccp_alpha	Float	0.00–0.02

2.5. Optimization objectives and fitness functions

The original LHS-GWO framework evaluated each BBC-DT candidate configuration using the mean accuracy obtained from stratified cross-validation. Because GWO was implemented as a minimization algorithm, the original fitness function was defined as one minus the mean cross-validation accuracy. This formulation was retained in the revised study as a reference condition to reproduce the original optimization behavior. However, the dataset was substantially imbalanced, with 83.61% non-hypertension and 16.39% hypertension observations. Under this distribution, an accuracy-oriented search can favor candidate configurations that classify the majority class correctly while providing limited sensitivity to hypertension cases. Therefore, the revised experiments evaluated alternative class-sensitive fitness functions while preserving the same BBC-DT architecture, hyperparameter search space, and optimization procedure.

For each scalar objective, the cross-validation score was first calculated across the training folds. Let $S_k(\theta)$ denote the performance score obtained by candidate hyperparameter configuration θ in fold k, and let K denote the number of folds. The mean cross-validation score was calculated as

$$\bar{S}(\theta) = \frac{1}{K}\sum_{k=1}^{K} S_k(\theta). \tag{19}$$

Because the GWO implementation minimized the objective function, each maximization-based performance measure was converted into a fitness value according to

$$f(\theta) = 1 - \bar{S}(\theta). \tag{20}$$

A smaller fitness value therefore represented a better candidate configuration.

2.5.1. Accuracy-based fitness

The accuracy-based objective reproduced the fitness function used in the original study. For fold k, accuracy was calculated as

$$ACC_k = \frac{TP_k + TN_k}{TP_k + TN_k + FP_k + FN_k}. \tag{21}$$

The mean cross-validation accuracy was

$$\overline{ACC}_i = \frac{1}{K}\sum_{k=1}^{K} Acc_{ik},\ k = 5, \tag{22}$$

where ACC_{ik} denotes the accuracy obtained by the ith hyperparameter configuration in the kth fold. Because GWO was implemented as a minimization procedure, the fitness value of the ith wolf was defined as

$$f_i = 1 - \overline{ACC}_i. \tag{23}$$

2.5.2. Balanced-accuracy fitness

Balanced accuracy was evaluated to provide equal weighting to the positive- and negative-class detection rates. For each fold,

$$BA_k = \frac{Recall_k + Specificity_k}{2}. \tag{24}$$

The mean cross-validation balanced accuracy was calculated as

$$\overline{BA} = \frac{1}{K}\sum_{k=1}^{K} B\,A_k, \tag{25}$$

and the fitness function was

$$f_{BA} = 1 - \overline{BA}. \tag{26}$$

This objective reduced the direct influence of class prevalence because sensitivity and specificity contributed equally to the optimization score.

2.5.3. F1-score fitness

F1-score was evaluated as an objective that jointly considered precision and recall for the hypertension class. For each fold,

$$F1_k = 2\frac{Precision_k \times Recall_k}{Precision_k + Recall_k}. \tag{27}$$

The mean cross-validation F1-score was

$$\overline{F1} = \frac{1}{K}\sum_{k=1}^{K} F\,1_k, \tag{28}$$

and the fitness was defined as

$$f_{F1} = 1 - \overline{F1}. \tag{29}$$

This objective penalized configurations that achieved high recall at the expense of very low precision, or high precision at the expense of low recall.

2.5.4. Recall-based fitness

Recall was evaluated as a sensitivity-prioritized objective to examine the behavior of the optimization procedure when hypertension-case detection was given primary importance. For fold k,

$$Recall_k = \frac{TP_k}{TP_k + FN_k}. \tag{30}$$

The mean cross-validation recall was

$$\overline{Recall} = \frac{1}{K}\sum_{k=1}^{K} Recall_k \tag{31}$$

and the corresponding fitness function was

$$f_{REC} = 1 - \overline{Recall}. \tag{32}$$

This objective explicitly rewarded configurations that reduced false-negative predictions. It was included as an analytical condition rather than as an assumption that recall alone constitutes the optimal screening criterion.

2.5.5. Screening-oriented weighted fitness

A screening-oriented weighted objective was evaluated to examine a controlled trade-off among sensitivity, specificity, F1-score, and PR-AUC. For fold k, the score was defined as

$$S_{screen,k} = 0.40\,Recall_k + 0.20\,Specificity_k + 0.20\,F1_k + 0.20\,PR\text{-}AUC_k. \tag{33}$$

The mean screening-oriented score was

$$\overline{S}_{screen} = \frac{1}{K}\sum_{k=1}^{K} S_{screen,k} \tag{34}$$

and the corresponding fitness function was

$$f_{screen} = 1 - \overline{S}_{screen}. \tag{35}$$

A larger weight was assigned to recall because the revised analysis focused on the detection of hypertension cases in a screening context. The weights were specified as an analysis-oriented preference and were not interpreted as clinically validated utility weights. This distinction was maintained to avoid overstating the clinical basis of the weighting scheme.

2.5.6. Comparison of scalar fitness functions

All scalar fitness functions were evaluated under the same experimental conditions. The BBC-DT search space, LHS initialization procedure, GWO position-update equations, population size, and iteration budget were held constant. Therefore, differences among the optimized configurations were interpreted as consequences of the optimization objective rather than changes in the underlying model architecture or search domain. The scalar objective comparison included accuracy, balanced accuracy, F1-score, recall, and the screening-oriented weighted criterion. The resulting configurations were subsequently evaluated using the full set of classification metrics rather than only the metric used during optimization. This design allowed the trade-offs induced by each fitness function to be examined directly.

2.6. Pareto-based multi-objective optimization

A Pareto-based multi-objective formulation was evaluated to represent the sensitivity–specificity trade-off without combining both objectives into a single weighted score. The multi-objective problem was formulated as

$$\min f_1(\theta) = 1 - \overline{Recall}(\theta) \tag{36}$$

and

$$\min f_2(\theta) = 1 - \overline{Specificity}(\theta). \tag{37}$$

A candidate solution θ_a was said to dominate another solution θ_b if

$$f_j(\theta_a) \leq f_j(\theta_b).$$

Solutions that were not dominated by any other candidate were retained in a non-dominated archive. The archive therefore represented alternative BBC-DT configurations with different sensitivity–specificity trade-offs. When a single representative solution was required for comparison with scalar-objective models, the objective values were normalized and the solution with the smallest Euclidean distance to the ideal point was selected:

$$d_i = \sqrt{\left(\tilde{f}_{1,i}\right)^2 + \left(\tilde{f}_{2,i}\right)^2}, \tag{38}$$

where $\tilde{f}_{1,i}$ and $\tilde{f}_{2,i}$ denote the normalized objective values for the candidate.

The selected solution was referred to as the Pareto compromise solution. The procedure was used as a Pareto-based extension of the existing search framework and was not presented as a newly

developed multi-objective GWO algorithm.

2.7. Direct comparison of randomly initialized GWO and LHS-GWO

To isolate the effect of population initialization, conventional GWO with random initialization was compared directly with LHS-initialized GWO. Both methods used the same BBC-DT architecture, 12-dimensional hyperparameter search space, fitness function, number of wolves, number of iterations, cross-validation scheme, and random-seed schedule. The only systematic difference between the two methods was the procedure used to generate the initial population. This comparison was required because the original manuscript attributed improved search-space coverage to LHS but did not directly establish whether LHS produced a measurable advantage over conventional random initialization.

3. Results and discussion

3.1. Reproduction of the original BBC-DT experiment

As shown in Figure 3, the performance of the BBC with DT results from the combination of class balancing and ensemble learning. In an imbalanced dataset, a DT constructs decision rules that are strongly influenced by the majority class because it contains a larger number of observations. As a result, patterns associated with the minority class receive less representation during node splitting. Balanced bagging addresses this imbalance by constructing multiple balanced training subsets and fitting a DT to each subset. This process gives minority-class observations a more proportional contribution to the learning process of the individual classifiers. Combining ensemble learning with sampling enables the base classifiers to be trained on more balanced data distributions and reduces the influence of class imbalance during model training. Classifier diversity is another important component of balanced bagging. Each decision tree is trained on a different subset of the training data, allowing the individual classifiers to develop different tree structures and error patterns. When their predictions are aggregated through a voting mechanism, the final decision does not depend on a single decision tree. Instead, predictions from multiple base classifiers jointly determine the final classification. Classifier diversity is important in ensemble learning because classifiers with different prediction and error patterns provide complementary information when their outputs are combined.

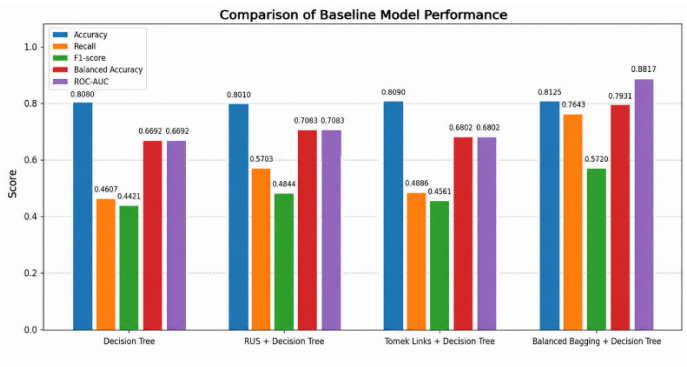

Figure 3. Comparison of the baseline model performance in terms of accuracy, recall, F1-score, balanced accuracy, and ROC-AUC.

After balanced bagging was used to reduce the influence of class imbalance and increase diversity among the base classifiers, LHS-GWO was applied to identify the BBC-DT hyperparameter configuration with the highest cross-validation accuracy. Figure 4 shows that the best cross-validation accuracy increased gradually during the early iterations, with more noticeable improvements up to approximately the eighth iteration, followed by smaller improvements until the highest value was reached at the sixteenth iteration and remained unchanged through the twenty-fifth iteration. This pattern indicates that better hyperparameter configurations were identified during the early stages of the search, whereas subsequent iterations produced progressively smaller improvements before the best value stabilized. During population initialization, Latin hypercube sampling divided each dimension of the search space into intervals and placed one sample within each interval, thereby providing stratified representation across the range of each dimension before the GWO position-update process was performed. This stratification supports the space-filling property of the initial design by maintaining sample representation across the range of each search dimension [32]. GWO then used the positions of α, β, and δ to guide candidate updates throughout the optimization process. The absence of further improvement after the sixteenth iteration indicates that, within the defined search space, population size, and number of iterations used in this study, no subsequent candidate achieved a higher cross-validation accuracy.

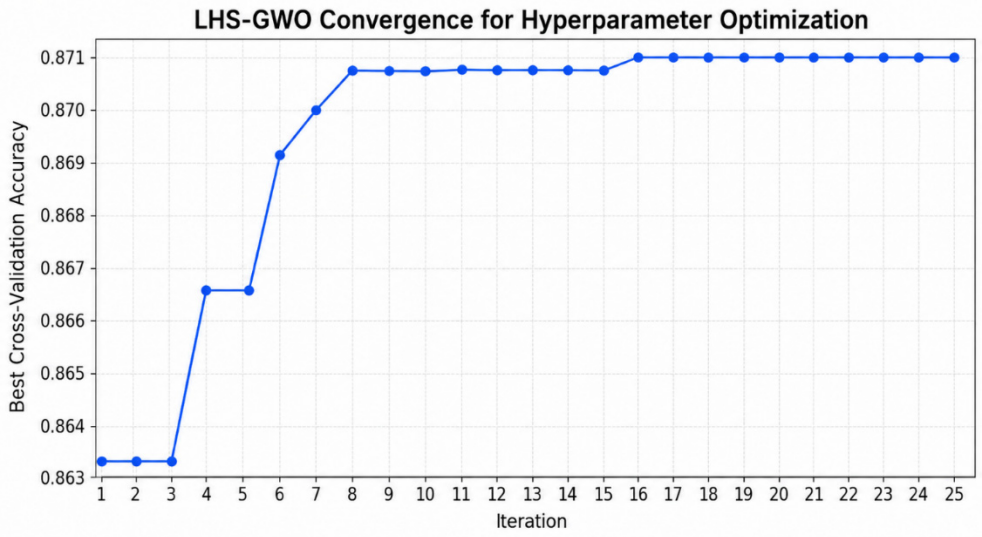

Figure 4. Convergence of LHS-GWO during BBC-DT hyperparameter optimization based on the best cross-validation accuracy.

After the LHS-GWO convergence pattern showed that the best value had stabilized during the final stage of the search, the best hyperparameter configuration was used to construct the optimized BBC-DT model. Table 3 shows that the optimization covered two groups of parameters: ensemble-level parameters in balanced bagging and structural parameters in the DT. Ensemble parameters such as n_estimators, max_samples, max_features_bagging, and sampling_strategy control the number of base classifiers and the proportions of samples and features used to construct the ensemble, while bootstrap and replacement determine the sampling mechanism used during the bagging process. At the DT level, criterion, max_depth, min_samples_split, min_samples_leaf, max_features_tree, and ccp_alpha control node construction, tree complexity, and pruning. These results show that LHS-GWO

optimized the ensemble-level and base-learner parameters simultaneously, so the final configuration reflects a joint adjustment of the balanced bagging mechanism and the decision tree structure.

Table 3. Optimal BBC-DT hyperparameter configuration obtained using LHS-GWO.

No.	Hyperparameter	Optimal Value
1	n_estimators	165
2	max_samples	0.667930
3	max_features_bagging	0.793685
4	sampling_strategy	0.241633
5	bootstrap	False
6	replacement	True
7	criterion	gini
8	max_depth	28
9	min_samples_split	12
10	min_samples_leaf	7
11	max_features_tree	0.744233
12	ccp_alpha	0.000702

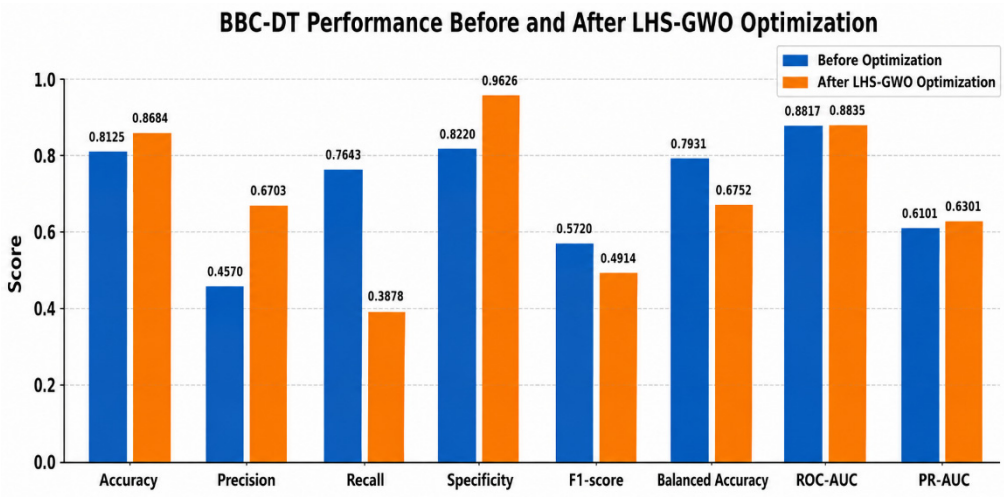

Figure 5. Performance comparison of BBC-DT before and after LHS-GWO hyperparameter optimization based on accuracy, precision, recall, specificity, F1-score, balanced accuracy, ROC-AUC, and PR-AUC.

After the best hyperparameter configuration was obtained using LHS-GWO, the performance of BBC-DT before and after optimization was compared in Figure 5. The results show that optimization increased accuracy, precision, specificity, ROC-AUC, and PR-AUC, but decreased recall, F1-score, and balanced accuracy. This pattern indicates a shift in the classification characteristics after optimization, whereby the model achieved a higher proportion of correct positive predictions and better identification of the non-hypertension class, while its ability to detect hypertension cases decreased. This pattern is consistent with the fitness function defined in Eq (20), which uses cross-validation accuracy to evaluate candidate hyperparameter configurations; therefore, the search process was directed towards configurations with higher overall accuracy rather than explicitly optimizing sensitivity to the hypertension class. In these results, this optimization direction is reflected in the increases in specificity and precision accompanied by decreases in recall and balanced accuracy. Therefore, the increase in accuracy after optimization was not accompanied by an improvement in the model's ability to identify hypertension cases, indicating that the choice of fitness function should be aligned with the classification objective when the detection of hypertension cases is a priority.

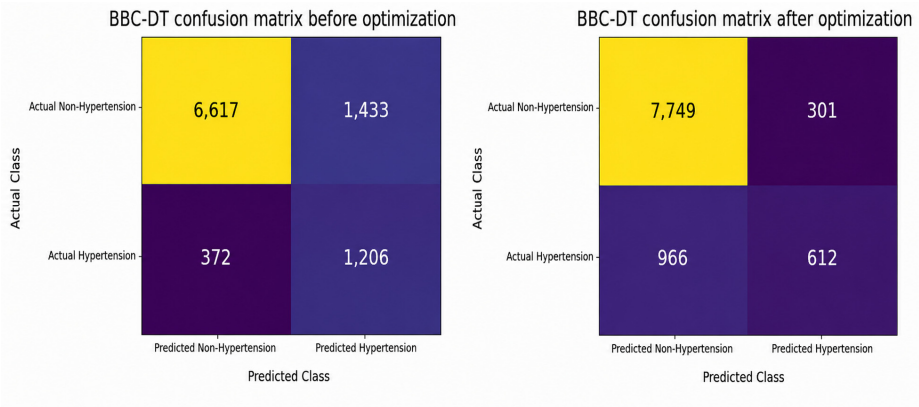

Figure 6. Comparison of the BBC-DT confusion matrices before and after LHS-GWO hyperparameter optimization.

As shown in Figure 6, LHS-GWO optimization changed the distribution of classification errors produced by BBC-DT. After optimization, the number of false positives decreased and the number of true negatives increased, reflecting an improvement in the model's ability to identify the non-hypertension class. However, these changes were accompanied by an increase in false negatives and a decrease in true positives, resulting in a larger number of hypertension cases that were not detected by the model. This pattern is consistent with the decrease in recall and balanced accuracy and the increase in specificity shown in Figure 4. The changes in the confusion matrix show that the improved identification of the non-hypertension class occurred alongside a reduced ability to detect the hypertension class. Thus, the decrease in false positives was accompanied by an increase in false negatives, reflecting a trade-off between identifying non-hypertension cases and detecting hypertension cases. These results show that the evaluation of models for imbalanced data should consider recall, balanced accuracy, and the number of false negatives together with accuracy and specificity when the detection of hypertension cases is the primary objective.

Based on the performance pattern and distribution of classification errors discussed in the previous section, the selection of the fitness function is an important aspect of hyperparameter

optimization for data with an imbalanced class distribution. For hypertension screening, the fitness function should not focus solely on accuracy but should also consider metrics that place greater emphasis on the ability to detect the hypertension class, such as recall, balanced accuracy, or F1-score. A multi-objective approach can also be used to consider several performance criteria simultaneously, thereby aligning the hyperparameter search process more closely with the screening objective and the class distribution characteristics of the analyzed data.

3.2. Comparison of baseline and imbalance-aware models

To assess the position of LHS-GWO-BBC-DT within a broader classification context, several baseline and imbalance-aware models were evaluated using the same test set. The evaluated models included decision tree (DT), SMOTE–Tomek-DT, BBC-DT, AdaBoost, EasyEnsemble, RUSBoost, class-weighted XGBoost, and class-weighted LightGBM. Their performance is summarized in Table 4.

Table 4. Performance comparison of baseline and imbalance-aware models.

Model	Accuracy	Precision	Recall	Specificity	F1	Balanced Accuracy	ROC-AUC	PR-AUC	MCC	Kappa
DT	0.8094	0.4249	0.4607	0.8778	0.4421	0.6692	0.6692	0.2841	0.3278	0.3274
SMOTE–Tomek-DT	0.8035	0.4176	0.5044	0.8621	0.4569	0.6833	0.6833	0.2919	0.3405	0.3383
BBC-DT	0.8125	0.4570	0.7643	0.8220	0.5720	0.7931	0.8817	0.6101	0.4865	0.4615
AdaBoost	0.8630	0.7897	0.2237	0.9883	0.3486	0.6060	0.8776	0.6005	0.3730	0.2978
EasyEnsemble	0.7590	0.3886	0.8200	0.7471	0.5273	0.7836	0.8805	0.6248	0.4414	0.3921
RUSBoost	0.7437	0.3715	0.8150	0.7297	0.5103	0.7723	0.7723	0.3331	0.4201	0.3680
XGBoost-weighted	0.7827	0.4189	0.8409	0.7713	0.5592	0.8061	0.8917	0.6465	0.4824	0.4357
LightGBM-weighted	0.7962	0.4346	0.8086	0.7938	0.5654	0.8012	0.8885	0.6380	0.4844	0.4476

As shown in Table 4, the evaluated algorithms produced different operating behaviors. AdaBoost exhibited a highly conservative positive-class prediction pattern. Its high specificity and precision were accompanied by substantially lower sensitivity to hypertension cases, indicating that its decision surface favored the majority class. EasyEnsemble and RUSBoost exhibited the opposite behavior. Both methods increased sensitivity to the hypertension class through their internal balancing mechanisms, but this improvement was accompanied by reduced ability to preserve correct predictions for the non-hypertension class. Class-weighted XGBoost and LightGBM produced more stable trade-offs between sensitivity and specificity. Their weighting mechanisms increased the training penalty associated with minority-class errors, thereby reducing the tendency of the learning process to favor the majority class. Among the evaluated baseline models, class-weighted XGBoost provided the strongest overall balance between minority-class detection and ranking performance. BBC-DT also showed comparatively balanced behavior because class balancing was performed at the bootstrap-

subset level before each decision tree was fitted. Consequently, hypertension cases received more proportional representation across the base estimators, which helped preserve recall and overall agreement between predicted and actual classes.

3.3. Effect of the fitness function on LHS-GWO-BBC-DT

The next experiment evaluated the effect of five fitness functions on the behavior of LHS-GWO-BBC-DT. All configurations used the same classifier architecture, search space, population size, and GWO update mechanism. The only factor varied across experiments was the objective function used to evaluate candidate solutions. The results are presented in Table 5.

Table 5. Performance of LHS-GWO-BBC-DT under different fitness functions.

Fitness	Accuracy	Precision	Recall	Specificity	F1	Balanced Accuracy	ROC-AUC	PR-AUC	MCC	Kappa
Accuracy	0.8644	0.6567	0.3612	0.9630	0.4661	0.6621	0.8796	0.6098	0.4190	0.3958
Balanced Accuracy	0.7933	0.4279	0.7744	0.7970	0.5512	0.7857	0.8756	0.5975	0.4631	0.4311
F1-score	0.8133	0.4581	0.7617	0.8234	0.5721	0.7925	0.8788	0.5995	0.4864	0.4620
Recall	0.7146	0.3540	0.8986	0.6785	0.5079	0.7886	0.8734	0.5897	0.4334	0.3566
Screening-weighted	0.7563	0.3884	0.8466	0.7386	0.5325	0.7926	0.8751	0.5972	0.4521	0.3970

As shown in Table 5, changing the fitness function substantially altered the decision behavior of the classifier even though the underlying architecture and search space remained unchanged. When accuracy was used as the fitness function, the search process favored candidate configurations that maximized the proportion of correct predictions across the entire dataset. Under class imbalance, this objective does not distinguish between the relative importance of errors in the majority and minority classes. As a result, configurations that reduced false positives could still be favored even when the number of false negatives increased. When balanced accuracy was used, sensitivity and specificity contributed equally to the fitness value. This changed the search direction because candidate configurations with poor performance on one class could no longer be compensated for solely by strong performance on the other class. The F1-based objective produced a different search pressure because it jointly considered precision and recall for the positive class. Therefore, the search did not simply favor configurations with high sensitivity, but also penalized configurations that generated excessive false positives. This behavior explains why the F1-based configuration produced a comparatively balanced distribution of errors. The recall-based objective placed the strongest pressure on reducing false negatives. Under this objective, the optimizer favored configurations that expanded the positive decision region whenever additional hypertension cases could be detected. The resulting increase in true positives was accompanied by an increase in false positives and, consequently, lower specificity. The screening-weighted objective produced intermediate behavior between balanced accuracy and recall. Recall received the largest weight, while specificity, F1-score, and PR-AUC continued to contribute to the objective. Consequently, the search was not driven exclusively by a

single type of classification error.

3.4. Pareto-based multi-objective optimization

A Pareto-based multi-objective experiment was conducted to avoid reducing two competing objectives to a single scalar fitness value. Recall and specificity were optimized simultaneously so that the search could preserve solutions representing different trade-offs between hypertension-case detection and non-hypertension classification. The selected compromise solution is compared with representative scalar-objective configurations in Table 6.

Table 6. Comparison of the Pareto compromise solution with scalar-objective configurations.

Model	Accuracy	Precision	Recall	Specificity	F1	Balanced Accuracy	ROC-AUC	PR-AUC	MCC
LHS-GWO-BBC-DT – Accuracy	0.8644	0.6567	0.3612	0.9630	0.4661	0.6621	0.8796	0.6098	0.4190
LHS-GWO-BBC-DT – F1	0.8133	0.4581	0.7617	0.8234	0.5721	0.7925	0.8788	0.5995	0.4864
LHS-GWO-BBC-DT – Pareto compromise	0.8186	0.4651	0.7136	0.8391	0.5631	0.7763	0.8754	0.5996	0.4716

The Pareto compromise solution did not produce an extreme value for either recall or specificity. Instead, it remained between configurations that strongly favored sensitivity and those that strongly favored specificity. Technically, this behavior is a direct consequence of the dominance mechanism. Candidate solutions were not ranked using a single scalar value. Instead, a solution was retained when no other candidate improved one objective without worsening the other. This mechanism therefore preserved a set of alternative operating points rather than forcing the search toward one predefined balance between both objectives. The Pareto front provides information that cannot be obtained from single-objective optimization because it explicitly represents the trade-off between sensitivity and specificity. The compromise solution reported in Table 6 should therefore be interpreted as a representative point selected from the non-dominated solution set rather than as the only optimal solution.

3.5. Threshold sensitivity analysis

After the model configuration had been obtained, threshold sensitivity analysis was conducted to evaluate how the classification threshold affected the distribution of positive and negative predictions. Threshold selection was performed using out-of-fold probabilities from the training data so that the held-out test set did not influence the selection of the final operating point. The selected threshold was 0.45. The corresponding test-set performance is summarized in Figure 7.

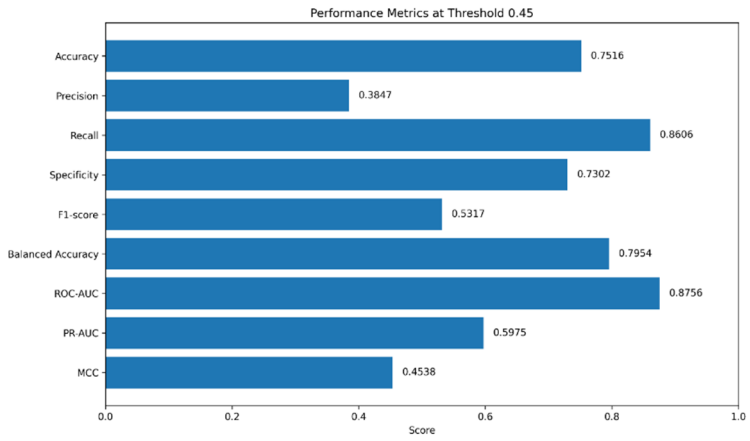

Figure 7. Performance at the selected classification threshold.

Reducing the classification threshold expanded the probability region assigned to the hypertension class. In practical terms, observations with predicted probabilities between 0.45 and the default threshold of 0.50 were reassigned from the negative class to the positive class. The resulting confusion matrix was $\begin{bmatrix} 5878 & 2172 \\ 220 & 1358 \end{bmatrix}$ and the false-negative rate was $FNR = \frac{220}{220+1358} = 0.1394$. Relative to the accuracy-oriented configuration, which produced 966 false negatives, the number of missed hypertension cases decreased by approximately $\frac{966-220}{966} \times 100 \approx 77.2\%$. This reduction resulted from a change in the decision rule rather than from a change in the ranking ability of the model. Accordingly, ROC-AUC and PR-AUC remained unchanged when only the threshold was modified. These metrics summarize probability-ranking behavior across thresholds, whereas recall, specificity, precision, F1-score, and the confusion matrix depend directly on the selected operating threshold.

3.6. False-negative analysis

False-negative errors were examined separately because they correspond to hypertension cases classified as non-hypertension at the selected operating point. For the accuracy-oriented LHS-GWO-BBC-DT configuration, the false-negative rate was FNR = 966/(966 + 612) = 0.6122. This indicates that a substantial proportion of hypertension cases fell on the negative side of the model's decision boundary. In contrast, under the selected threshold of 0.45, FNR = 220/(220 + 1358) = 0.1394. The reduction in false-negative rate indicates that a more positive-class-sensitive decision rule moved a large number of hypertension observations into the correct decision region. Technically, the observed trade-off can be explained through the relationship between false negatives and false positives. Expanding the positive decision region increases the probability that true hypertension cases are detected, but it also causes some non-hypertension observations to be classified as positive.

3.7. Role of PR-AUC, MCC, and Cohen's kappa

PR-AUC evaluates the relationship between precision and recall across multiple thresholds. When the positive class represents a smaller proportion of the dataset, this metric provides a direct

assessment of how effectively the model preserves positive predictive quality while retrieving minority-class observations. As shown in Table 4, class-weighted XGBoost achieved the highest PR-AUC among the principal baseline models, followed by class-weighted LightGBM and EasyEnsemble. This pattern indicates that these models provided relatively strong ranking performance for hypertension cases.

MCC measures the correlation between predicted and observed classes while incorporating all four elements of the confusion matrix. The comparison between accuracy-oriented and F1-oriented LHS-GWO-BBC-DT illustrates the importance of this distinction. The accuracy-oriented configuration achieved higher overall accuracy but lower MCC than the F1-oriented configuration. This indicates that a higher proportion of globally correct predictions does not necessarily correspond to a more balanced agreement across both classes. Cohen's kappa provides an additional perspective by correcting the observed agreement for agreement that may occur by chance. In a dataset dominated by one class, this correction helps distinguish genuine predictive agreement from agreement partly induced by the class distribution.

3.8. Effect of the fitness function on search behavior

When the fitness-function experiments are considered together, the results show that the objective function affects not only the final metric values but also the direction of hyperparameter search during GWO optimization. Under accuracy-based fitness, candidate solutions that increased the total number of correct predictions obtained a selective advantage. Because the non-hypertension class represented the majority of the observations, hyperparameter changes that improved performance on this class had a comparatively large influence on fitness.

Under balanced-accuracy fitness, sensitivity and specificity contributed equally to the optimization objective. Consequently, candidate solutions with substantial loss of sensitivity could no longer be compensated for solely by improved specificity. Under F1-based fitness, the search pressure was concentrated on the precision–recall relationship for the positive class. This objective indirectly penalized two extreme behaviors: excessive false-positive prediction and insufficient recovery of true-positive cases.

Under recall-based fitness, the search process more aggressively expanded the positive decision region. This behavior increased true-positive detection but also increased false-positive classification. The screening-weighted objective introduced a composite search pressure. Recall contributed the largest weight, while specificity, F1-score, and PR-AUC constrained the optimizer from moving entirely toward a sensitivity-only solution.

3.9. Position of LHS-GWO-BBC-DT relative to comparator models

The experimental results do not indicate that LHS-GWO-BBC-DT consistently achieved the highest performance across all evaluation metrics when compared with established ensemble and gradient-boosting methods. Class-weighted XGBoost achieved strong balanced accuracy and PR-AUC, while class-weighted LightGBM also provided a competitive combination of recall and specificity. However, LHS-GWO-BBC-DT provides an analytical advantage because the behavior of the classifier can be explicitly controlled through the fitness function. Using the same classifier and search space, different optimization objectives produced clearly different operating points. This makes it possible to

examine the direct effects of accuracy, balanced accuracy, F1-score, recall, and weighted objectives without changing the underlying classifier architecture. Accordingly, the contribution of the present experiments should not be framed as evidence that LHS-GWO-BBC-DT universally outperforms alternative classifiers. Rather, the results demonstrate that the objective function is a critical component of hyperparameter optimization under class imbalance. The findings also indicate that optimizer performance should not be assessed solely on the basis of the best fitness value. Two model configurations may exhibit similar probability-ranking ability while producing markedly different classification behavior because of differences in optimization objective and decision threshold.

4. Conclusions

This study illustrates the significant influence of the fitness function in guiding the optimization behavior of LHS-GWO-BBC-DT in the context of class imbalance. Optimization focused on accuracy tended to favor configurations with high overall accuracy and specificity, yet it diminished the detection of hypertension cases. In contrast, objectives oriented toward balanced accuracy, F1-score, recall, and screening shifted the search toward solutions with greater sensitivity to the minority class. The Pareto-based formulation further revealed that recall and specificity are competing objectives. Additionally, threshold analysis confirmed that hyperparameter optimization and decision-threshold selection should be addressed as distinct stages. The comparison with other classifiers suggests that LHS-GWO-BBC-DT does not consistently outperform established methods; thus, its primary contribution lies in elucidating how the design of objective functions influences search behavior and the distribution of classification errors.

Future research should assess the robustness of these findings by employing a greater number of independent optimization runs, alongside repeated and nested cross-validation, and external validation using independent hypertension datasets. Additionally, further studies should compare Latin hypercube sampling (LHS) initialization with conventional random initialization under identical computational budgets. Investigating alternative multi-objective formulations that incorporate PR-AUC, MCC, model complexity, or false-negative cost is also recommended. Moreover, exploring adaptive weighting strategies, cost-sensitive learning, probability calibration, and their application to other imbalanced health-classification problems may enhance the understanding of how metaheuristic optimization objectives affect model behavior across various clinical screening contexts.

Use of AI tools declaration

The authors confirm that Artificial Intelligence (AI) was not utilized in the preparation of this article.

Acknowledgments

This work was supported by a research grant from the Ministry of higher education, science and technology Indonesia under the Regular Fundamental Research Scheme (PFR) 2026 [251-11/UN7.D2/PP/IV/2026].

Conflict of interest

The authors declare there is no conflict of interest.

References

1. F. A. Ahmad, The dangers of hypertension threaten young Indonesians, Health Development Policy Agency, Ministry of Health of the Republic of Indonesia, 2024. Available from: https://www.badankebijakan.kemkes.go.id/bahaya-hipertensi-mengintai-anak-muda-indonesia/.
2. D. Arlinta, High blood pressure among Indonesians; many adolescents also have hypertension, Kompas.id, 2026. Available from: https://www.kompas.id/artikel/en-tensi-tinggi-penduduk-indonesia-remaja-juga-banyak-yang-hipertensi.
3. Ministry of Health of the Republic of Indonesia, 2026 CKG Results: 1 million previously healthy residents now detected with hypertension, 2026. Available from: https://www.kemkes.go.id/id/hasil-ckg-2026-1-juta-warga-yang-semula-sehat-kini-terdeteksi-hipertensi.
4. R. Kurniawan, B. Utomo, K. N. Siregar, K. Ramli, B. Besral, R. J. Suhatril, et al., Hypertension prediction using machine learning algorithm among Indonesian adults, *IAES Int. J. Artif. Intell.*, **12** (2023), 776–784. https://doi.org/10.11591/ijai.v12.i2.pp776-784
5. R. I. Estiko, B. Widyantoro, D. A. Juzar, R. M. Yusup, I. F. Rakhmat, E. Rijanto, Screening hypertension using non laboratory risk factors with machine learning: A retrospective cross sectional study in Indonesia, *BMJ Open*, **15** (2025), e092364. https://doi.org/10.1136/bmjopen-2024-092364
6. F. A. Bornee, M. R. K. Chowdhury, M. Z. Islam, Z. Raihana, H. A. Chowdhury, M. Kader, et al., Machine learning approaches in identifying factors associated with hypertension and undiagnosed hypertension in adults in rural areas of Bangladesh, *Arch. Public Health*, **84** (2026), 116. https://doi.org/10.1186/s13690-026-01941-z
7. R. R. Ridoy, M. Mothashin, M. A. Hossain, T. Mahbub, S. C. Barmon, P. K. Saha, et al., Machine learning based identification of key predictors and socioeconomic inequalities in the co existence of diabetes and hypertension among Bangladeshi adults, *Sci. Rep.,* **16** (2026), 8233. https://doi.org/10.1038/s41598-026-37966-1
8. D. Nuryunarsih, L. Herawati, A. Badi'ah, J. D. T. Donsu, Okatiranti, Predicting changes in systolic and diastolic blood pressure of hypertensive patients in Indonesia using machine learning, *Curr. Hypertens. Rep.*, **25** (2023), 377–383. https://doi.org/10.1007/s11906-023-01261-5
9. J. Han, L. Lin, T. Zhao, Y. Zhang, Z. Zhao, W. Wang, et al., An interpretable machine learning model for predicting stroke in patients with hypertension: Insights from the SPRINT, *Sci. Rep.*, **16** (2026), 18891. https://doi.org/10.1038/s41598-026-48322-8
10. S. Dhanushkumar, S. Padmakala, Explainable ML model hypertension prediction: Permutation importance, partial dependence, and lift analysis of logistic regression, random forest, and gradient boosting, in *Proceedings of the 2026 IEEE 2nd International Conference on Quantum Photonics, Artificial Intelligence and Networking*, IEEE, (2026), 1–6. https://doi.org/10.1109/QPAIN69676.2026.11546501

11. V. Muttangi, H. B. Samruddi, A. Deshpande, K. Patel, P. T. Akkasaligar, Hypertension risk prediction: Insights from metabolic profiles, in *Proceedings of the 2026 IEEE International Conference on Convergence in Computing Technology*, IEEE, (2026), 1–6. https://doi.org/10.1109/I3CTCON68242.2026.11507754
12. A. Eldem, An integrated machine learning framework for the early diagnosis of hypertension disease, *Eng. Appl. Artif. Intell.*, **162** (2025), 112564. https://doi.org/10.1016/j.engappai.2025.112564
13. H. Lee, Y. Kim, Machine learning based risk assessment of hypertension among older adults in Korea, *Digit. Health*, **12** (2026), 1–10. https://doi.org/10.1177/20552076261435733
14. W. Liu, I. W. Tsang, Making decision trees feasible in ultrahigh feature and label dimensions, *J. Mach. Learn. Res.*, **18** (2017), 1–36. https://doi.org/10.5555/3122009.3176825
15. H. He, E. A. Garcia, Learning from imbalanced data, *IEEE Trans. Knowl. Data Eng.*, **21** (2009), 1263–1284. https://doi.org/10.1109/TKDE.2008.239
16. G. Lemaître, F. Nogueira, C. K. Aridas, Imbalanced learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning, *J. Mach. Learn. Res.*, **18** (2017), 1–5. https://doi.org/10.48550/arxiv.1609.06570
17. A. B. Hassanat, A. S. Tarawneh, G. A. Altarawneh, A. Almuhaimeed, Stop oversampling for class imbalance learning: A critical review, *IEEE Access*, **10** (2022), 47643–47660. https://doi.org/10.1109/ACCESS.2022.3169512
18. S. Hido, H. Kashima, Y. Takahashi, Roughly balanced bagging for imbalanced data, *Stat. Anal. Data Min.*, **2** (2009), 412–426. https://doi.org/10.1002/sam.10061
19. P. Probst, A. L. Boulesteix, B. Bischl, Tunability: Importance of hyperparameters of machine learning algorithms, *J. Mach. Learn. Res.*, **20** (2019), 1–32.
20. S. E. Roshan, S. Asadi, Improvement of bagging performance for classification of imbalanced datasets using evolutionary multi objective optimization, *Eng. Appl. Artif. Intell.*, **87** (2020), 103319. https://doi.org/10.1016/j.engappai.2019.103319
21. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw.*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
22. J. Kong, W. Kowalczyk, D. A. Nguyen, T. Bäck, S. Menzel, Hyperparameter optimization for improving classification under class imbalance, in *Proceedings of the 2019 IEEE Symposium Series on Computational Intelligence*, IEEE, (2019), 3072–3078. https://doi.org/10.1109/SSCI44817.2019.9002679
23. R. Ahmed, G. P. Rangaiah, S. Mahadzir, S. Mirjalili, M. H. Hassan, S. Kamel, Memory, evolutionary operator, and local search based improved grey wolf optimizer with linear population size reduction technique, *Knowl.-Based Syst.*, **264** (2023), 110297. https://doi.org/10.1016/j.knosys.2023.110297
24. M. Sam'an, Farikhin, M. Munsarif, An improved decision tree model through hyperparameter optimization using a modified gray wolf optimization for diabetes classification, *Comput. Methods Biomech. Biomed. Eng.*, **29** (2026), 1843–1859. https://doi.org/10.1080/10255842.2025.2460178
25. M. Tayebi, S. El Kafhali, Performance analysis of metaheuristics based hyperparameters optimization for fraud transactions detection, *Evol. Intell.*, **17** (2024), 921–939. https://doi.org/10.1007/s12065-022-00764-5

26. S. R. Dip, H. K. Meena, Enhancing brain tumor detection in MRI with feature fusion and SMOTE optimization techniques, in *Proceedings of the 2025 7th International Conference on Signal Processing, Computing and Control*, IEEE, (2025), 574–579. https://doi.org/10.1109/ISPCC66872.2025.11039387
27. A. A. Aouragh, M. Bahaj, F. Toufik, Balancing and metaheuristic techniques for improving machine learning models in brain stroke prediction, *IAES Int. J. Artif. Intell.*, **14** (2025), 473–481. https://doi.org/10.11591/ijai.v14.i1.pp473-481
28. X. Zheng, A. Batool, S. Kumar, N. M. Shahani, Grey wolf optimization optimized ensemble models for predicting the uniaxial compressive strength of rocks, *Appl. Sci.*, **16** (2026), 1130. https://doi.org/10.3390/app16021130
29. M. Sarhani, S. Voß, R. Jovanovic, Initialization of metaheuristics: Comprehensive review, critical analysis, and research directions, *Int. Trans. Oper. Res.*, **30** (2023), 3361–3397. https://doi.org/10.1111/itor.13237
30. B. Tu, F. Wang, Y. Huo, X. Wang, A hybrid algorithm of grey wolf optimizer and Harris hawks optimization for solving global optimization problems with improved convergence performance, *Sci. Rep.*, **13** (2023), 22909. https://doi.org/10.1038/s41598-023-49754-2
31. C. Zhu, J. Zhang, J. Yang, Grain condition inspection: One improved grey wolf algorithm for nodes deployment optimization combining with path planning of multi inspection robots, *Soft Comput.*, **28** (2024), 11971–11986. https://doi.org/10.1007/s00500-024-09935-1
32. M. D. Shields, J. Zhang, The generalization of Latin hypercube sampling, *Reliab. Eng. Syst. Saf.*, **148** (2016), 96–108. https://doi.org/10.1016/j.ress.2015.12.002



AIMS Press

©2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0)