Methods and Approaches for Mathematical Optimization of Hyperparameters of Machine Learning Models

Fakhriddin Shodiev[1], Sharipboy Fayzullaev[1], Yulduz Nematjonova[1], Shaun Billiones[1, 2], Shafoat Eshonkhulova[3], Shakhnoza Samandarova[1], Laylo Ernazarova[1] and Valida Chorieva[1]
[1]*Samarkand State Pedagogical Institute, Spitamen Shah Str. 166, 140100 Samarkand, Uzbekistan*
[2]*Polytechnic University of the Philippines, Anonas Str., 1016 Manila, Philippines*
[3]*Jizzakh State Pedagogical University, Sh. Rashidov Str., 130100 Jizzakh, Uzbekistan*
shodiyevfaxriddin3@gmail.com, sharif_samarkand@icloud.com, yulduznematjonova78@gmail.com, shaun.cs.teacher@gmail.com, shafoateshonkulovaaa@gmail.com, samandarovashkhnoza123@gmail.com, layloernazarova1@gmail.com, validachoriyeva@gmail.com

Keywords: Hyperparameter Optimization, Genetic Algorithm, Ant Colony Optimization, Ensemble Models, Random Forest, XGBoost, Chronic Renal Illness, Medical Diagnostics.

Abstract: This paper investigates the mathematical optimization of machine learning hyperparameters in the context of early chronic kidney disease (CKD) detection. This study compares two high-performing ensemble classifiers - Random Forest and XGBoost - under five tuning strategies: default scikit-learn settings, Grid Search, Random Search, a Genetic Algorithm (GA), and Ant Colony Optimization (ACO). Logistic regression, k-nearest neighbors, support vector machines, decision trees, and Naive Bayes were evaluated as baseline comparators under default settings only; their tuning results are reported in the supplementary material. The main experimental comparison focuses on RF and XGBoost, for which full five-strategy benchmarks, sensitivity analysis, and runtime profiling are available. Grid Search and Random Search are shown to be impractical at scale: for Random Forest, Grid Search evaluated 432 configurations in 4.2 hours and Random Search evaluated 200 in 1.8 hours, neither reliably reaching high-sensitivity solutions. GA-optimized Random Forest and XGBoost reached AUC-ROC values of 0.963 and 0.968, respectively, while ACO-optimized variants achieved 0.969 and 0.975. Critically, sensitivity for the CKD class the clinically most important metric rose from 0.864 to 0.924 (ACO) for Random Forest. Paired Wilcoxon signed-rank tests confirmed that all GA/ACO improvements over Grid Search are statistically significant ($p < 0.01$). The results establish that evolutionary optimization offers a practically favorable accuracy-to-runtime trade-off for medical classification tasks where the cost of false negatives is high.

1 INTRODUCTION

The predictive performance of any machine learning model depends heavily on the optimal tuning of its hyperparameters. Hyperparameters are mathematical parameters that are given before training a model and control its complexity, flexibility, and generalization ability. If they are not chosen correctly, the model will experience problems of overfitting or underfitting. Therefore, mathematical optimization of hyperparameters is one of the central areas of machine learning research today.

Parallel advancements in algorithmic modeling across complex mechanical systems, structural safety, and geospatial infrastructure diagnostics further underscore the necessity of robust computational optimization frameworks [1] - [4]. Classical machine learning models, such as logistic regression, k-nearest neighbors, support vector machines, decision trees, and Bayesian classifiers, also have hyperparameters that indicate the functional form and boundaries of the modeling [5], [6]. The type of kernel and the coefficient in SVM, the value and distance measure in kNN, and the depth level in decision trees all have a significant impact on the final result of the modeling. Although these models are relatively simple, finding their optimal parameters is still considered a mathematical problem with an uncertain search space.

In recent years, the widespread use of ensemble models has complicated the process of optimizing hyperparameters. Ensemble approaches generate

robust models that generalize algorithms such as Bagging, Boosting, Random Forest, Gradient Boosting, XGBoost and CatBoost [7], [8]. Since they consist of a complex set of hyperparameters, the optimization process is computationally intensive. Ensemble models are becoming more and more popular in areas where high accuracy is required, especially in areas such as public administration, economic forecasting, medical diagnosis, and technical diagnostics. Accordingly, mathematical optimization strategies are also improving day by day. Traditional methods for hyperparameter selection, such as Grid Search and Random Search, focus on identifying the best result by examining all or random parameter combinations of the algorithm. However, in large-scale space, this process slows down significantly. To speed up this process and make it more efficient, mathematical methods including Bayesian optimization, gradient-based search, evolutionary algorithms, surrogate models, population optimization, and game theory-based strategies have been created. The factors mentioned above show how important this topic is. When hyperparameters are optimized correctly, the model's accuracy goes up, the cost of computing goes down, the learning process stays stable, and the techniques utilized have a far greater practical value [9], [10]. Against this background, the present work makes several specific contributions that, taken together, distinguish it from prior studies. First, rather than evaluating a single optimiser, we implement both GA and ACO within a unified experimental framework on the same dataset and under identical evaluation conditions, which allows a direct, reproducible comparison of the two evolutionary strategies. Second, the encoding schemes for the two methods are designed to reflect the structural differences between the algorithms: for GA, each hyperparameter configuration is encoded as a fixed-length mixed-type chromosome combining discrete and real-valued genes, with domain-specific mutation operators that respect the feasibility boundaries of each parameter; for ACO, the search graph is layered so that each layer corresponds to one hyperparameter, allowing the pheromone trail to capture dependencies between adjacent hyperparameter choices across iterations. Third, the evaluation is conducted in a medical diagnostic setting early detection of chronic kidney disease - where sensitivity to the positive class carries disproportionate clinical weight, making this application more demanding than standard accuracy-focused benchmarks. Fourth, a systematic runtime analysis is included alongside prediction quality metrics, providing practitioners with a practical cost-benefit basis for choosing between the two evolutionary approaches and conventional search strategies.

2 METHODOLOGY

This study relies on the Chronic Kidney Disease (CKD) dataset obtained from the UCI Machine Learning Repository. The dataset comprises 400 patient records collected from a hospital in Tamil Nadu, India, over a two-year period. Each record includes 24 clinical and laboratory attributes such as blood pressure, serum creatinine, hemoglobin, albumin, and blood glucose - along with a binary class label indicating the presence or absence of CKD. Of the 400 instances, 250 belong to the CKD class and 150 to the non-CKD class, yielding a mild class imbalance that was accounted for in the evaluation protocol. Before modelling, all records containing missing values were imputed using median substitution for continuous attributes and mode substitution for nominal ones. Categorical variables were encoded via label encoding, and all continuous features were normalized to the [0, 1] range using min-max scaling, which maps each attribute x_i to $\frac{(x_i - x_{min})}{(x_{max} - x_{min})}$. Categorical variables encoded via label encoding were left in their integer-coded form and were not rescaled, as tree-based models are invariant to monotone transformations of ordinal predictors. The processed dataset was partitioned into a training set 70%, a validation set 15%, and a held-out test set 15%, following a stratified split to preserve the class ratio across all three subsets. Each experimental configuration including default settings, Grid Search, Random Search, GA, and ACO - was repeated ten times with different random seeds, and mean values together with standard deviations are reported. All experiments were conducted in Python 3.10 using scikit-learn 1.3 on a workstation equipped with an Intel Core i7-12700 processor and 16 GB of RAM. The following types of models are considered during the study. Logistic regression, support vector machines, k-nearest neighbors, decision trees, and Naive Bayes are all classic models. Ensemble models such as Random Forest, Gradient Boosting, XGBoost and other boosting-based approaches are presented. A vector of hyperparameters for each model.

$$\Theta = (\theta_1, \theta_2, \ldots, \theta_d).$$

It is defined in the form of an expression. $\Theta_{SVM} = (C, \gamma, k\ ernel)$ for SVM, Θ_{RF} = (n_estimators,

max_depth, min_samples_split) for Random Forest, etc. Evaluation algorithms search this $\Theta \subseteq R^d$ space and find the objective function

$$J(\Theta) = 1 - \text{Metric}(\Theta).$$

minimize option or directly optimize max M etric(Θ). Here, Matric refers to accuracy, F1, AUC, or other selected metrics.

Two main optimization schemes were used in the methodology, which are as follows:

- Genetic algorithm-based optimization technique,
- Process of optimization based on the Ant Colony Optimization.

The common functional diagram for both optimization architectures consists of the following blocks: Input data and model selection, Hyperparameter encoding, Evolutionary search, Model training and evaluation for each candidate hyperparameter, best solution selection and validation, Final model evaluation on the test set. Below, evolutionary methods are described in detail in sequence [11], [12].

Hyperparameter optimization was performed using a Genetic Algorithm (GA) with a population of 50 chromosomes, tournament selection (size 3), single-point crossover (p_c = 0.85), mutation (p_m = 0.05), and 5% elitism over 30 generations. The optimized search space included Random Forest (n_estimators ∈ [50, 500]$, max_depth ∈ [3, 20], min_split ∈ [2, 10] min_leaf ∈ [1, 5]) and XGBoost (n_estimators ∈ [50, 500], max_depth ∈ [3, 12], learning_rate ∈ [0.01, 0.30]$, subsample/colsample ∈ [0.60, 1.00]$) configurations. Early stopping was triggered if the best fitness value stalled for 10 consecutive generations, indicating that the population had mostly converged [13], [14].

2.1 Chromosome and Population Identification

Each potential solution is a vector of hyperparameters

$$X_i = (\theta_1^{(i)}, \theta_2^{(i)}, \ldots, \theta_d^{(i)}),$$

and is interpreted as a chromosome. The initial population is generated based on random or heuristic rules in $P^{(0)} = \{X_1, X_2, \ldots, X_N\}$.N ∈ Z is the population size.

2.2 Determining the Fitness Function

A suitable model for each chromosome is trained and evaluated on the validation set, resulting in a fitness function as follows.

$$F(X_i) = \text{Metric}(\Theta = X_i).$$

When it is necessary to maximize the F1 criterion, it looks like this:

$$F(X_i) = F1_{\text{score}(X_i)}.$$

2.3 The Selection Operator

Good individuals are selected with high probability to form the next generation. This is done through a tournament selection or roulette method. The roulette method is as follows.

$$P(X_i) = \frac{F(X_i)}{\sum_{j=1}^{N} F(X_j)}.$$

Based on the given probabilities, pairs of paternal chromosomes are formed.

2.4 Crossover Operator

The selected chromosome pairs are shuffled according to hyperparameters.

The probability of crossbreeding is selected in the $P_c \in [0.6, 0.9]$ range. In single-point hybridization, it is indexed randomly.

$$X_{new}^{(1)} = (\theta_1^{(1)}, \ldots, \theta_k^{(1)}, \theta_{k+1}^{(2)}, \ldots, \theta_d^{(2)}),$$

$$X_{new}^{(2)} = (\theta_1^{(2)}, \ldots, \theta_k^{(2)}, \theta_{k+1}^{(1)}, \ldots, \theta_d^{(1)}).$$

This method allows for a variety of hyperparameter combinations to be reshuffled.

2.5 Mutation Operator

Mutation is used to enhance local search and prevent the population from becoming monotonous. The probability of mutation is usually small and is as follows.

$$X_i^{(mut)} = X_i \left(\theta_1^{(1)}, \ldots, \theta_j^{(new)}, \theta_{k+1}^{(2)}, \ldots, \theta_d^{(2)}\right).$$

For each gene, a change can be made in the following form:

- by re-selecting values for discrete parameters from the admissible set
- and for continuous parameters it is as follows

$$\theta_j^{new} = \theta_j^{old} + e, e \sim U(-\delta, \delta),\ \delta > 0.$$

2.6 Criteria for Forming and Stopping a New Population

Based on the principle of elitism, the best solutions from the generation resulting from crossing and mutation are preserved and a new population is formed in the form of $P^{(t+1)} = \text{Elite}(p^{(t)}) \cup \text{offspring}(p^{(t)})$. The algorithm iterates until one of the following conditions is met:

- the number of iterations must reach $t < T_{max}$;
- the value of the best fitness function should remain almost unchanged over a certain number of iterations;
- A specified quality threshold is required to be achieved (e.g., F1 > 0.90).

The $\Theta^*_{GA} = \text{argmax}\{X_i \in p\}$ obtained as a result of GA is recorded as the optimal set of hyperparameters.

2.7 Hyperparameter Optimization Based on Ant Colony Optimization

Ant Colony Optimization (ACO) was executed for 40 iterations using 30 artificial ants navigating a layered parameter graph.

Paths were constructed probabilistically using a pheromone weight of α=1.0 and a heuristic weight of β=2.0. The search concluded automatically when the global F1-score stabilized for five successive iterations, signifying that the search had converted to a limited array of promising paths [15], [16].

2.8 Represent the Problem in Graph Form

Hyperparameter selection is approached as a sequence of step-by-step decisions. The set of allowed values for each parameter is defined as nodes. So, the graph will look like the one below.

$$G = (V, E).$$

where V are the values of the hyperparameters, E are the transitions between them. The initial pheromone amount $\tau_{ij}(0) = \tau_0$is assigned to each edge.

2.9 Ant Placement and Road Construction

In each iteration, m ants are placed at the starting node. Each ant selects parameters, forming a complete vector of hyperparameters. The probability of moving from node to node based on pheromone and heuristic information is as follows.

$$P_{ij}^k = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{l \in \Omega_i} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}},$$

where η_{ij} is the heuristic indicator, the proximity of a given parameter to the desirable range are the hyperparameters that regulate the effect of the pheromone α and β heuristics, respectively, and Ω_i denotes the set of allowed transitions from the current node. A model is trained on the Θ_k hyperparameters formed for each ant and the fitness function $F(\Theta_k)$ is calculated [17], [18].

2.10 Pheromone Upgrade

After the iteration is complete, the pheromone is updated and evaporated along all edges, and the best path is promoted by the following expression:

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k,$$

where $\rho \in (0,1)$ represents the evaporation coefficient.

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k}, & \text{if an ant used edge (i,j)} \\ 0, & \text{otherwise} \end{cases}$$

$L_k = \frac{1}{F(\Theta_k)}$ is the "path quality" obtained by the ant $k \in \{1,2,\ldots,m\}$, and $Q > 0$is the normalization constant. As a result, edges located in high-quality hyperparameter combinations collect more pheromones and are prioritized in subsequent iterations.

2.11 Stopping Criteria and Solution

The algorithm is iteratively repeated and at each iteration the best ant is found and the global best solution is expressed as follows.

$$\Theta^*_{ACO} = \arg\max_{\Theta_k} F(\Theta_k).$$

The algorithm stops when the number of iterations reaches a specified limit or when the fitness value stabilizes [19].

Table 1: Results of Random Forest and XGBoost models under different optimization methods are presented.

Method	Accuracy	Sensitivity	Specificity	F1	AUC-ROC	Runtime
RF - Default	0.912±0.009	0.864±0.012	0.941±0.008	0.889±0.011	0.931±0.009	-
RF - Grid Search	0.927±0.007	0.879±0.010	0.958±0.006	0.904±0.009	0.944±0.007	4.2±0.3 h
RF - Random Search	0.923±0.008	0.871±0.011	0.954±0.007	0.896±0.010	0.939±0.008	1.8±0.2 h
RF - GA	0.948±0.006	0.917±0.008	0.967±0.005	0.931±0.007	0.963±0.005	2.1±0.2 h
RF - ACO	0.955±0.005	0.924±0.007	0.971±0.004	0.938±0.006	0.969±0.004	2.4±0.2 h
XGB - Default	0.921±0.008	0.877±0.011	0.948±0.007	0.899±0.010	0.940±0.008	-
XGB - Grid Search	0.935±0.006	0.891±0.009	0.961±0.005	0.913±0.008	0.952±0.006	3.7±0.3 h
XGB - Random Search	0.930±0.007	0.884±0.010	0.957±0.006	0.906±0.009	0.947±0.007	1.6±0.2 h
XGB - GA	0.956±0.005	0.928±0.007	0.974±0.004	0.941±0.006	0.968±0.004	1.9±0.1 h
XGB - ACO	0.961±0.004	0.935±0.006	0.978±0.003	0.947±0.005	0.975±0.003	2.2±0.2 h

3 RESULTS

Across all ten repeated runs, the mean performance values and their standard deviations are summarized in Table 1. When both classifiers were trained under default scikit-learn hyperparameter settings, Random Forest reached a test accuracy of 0.912 (±0.009) and XGBoost reached 0.921 (±0.008). While these baseline figures are reasonable, sensitivity for the CKD class arguably the most consequential metric in a screening context stood at only 0.864 and 0.877, respectively, leaving a meaningful fraction of true cases undetected. Conventional search methods offered partial improvement. Grid Search over 432 candidate configurations for Random Forest raised sensitivity to 0.879 and F1 to 0.904, but required an average wall-clock time of 4.2 hours per run, which is impractical for iterative model development. Random Search over 200 sampled configurations reduced runtime to approximately 1.8 hours while delivering only marginally lower performance sensitivity 0.871, F1 0.896, confirming that neither approach resolves the efficiency-accuracy trade-off at scale. Evolutionary optimization yielded substantially stronger results. GA-tuned Random Forest attained accuracy of 0.948 (±0.006), sensitivity of 0.917 (±0.008), F1 of 0.931 (±0.007), and AUC-ROC of 0.963 (±0.005). The corresponding GA-tuned XGBoost values were 0.956 (±0.005), 0.928 (±0.007), 0.941 (±0.006), and 0.968 (±0.004). ACO-optimized variants pushed performance a notch further: Random Forest reached sensitivity 0.924 and AUC-ROC 0.969, while XGBoost reached sensitivity 0.935 and AUC-ROC 0.975. Paired Wilcoxon signed-rank tests, applied across ten repeated runs with Bonferroni correction for multiple comparisons (family-wise $\alpha = 0.05$), confirmed that GA- and ACO-tuned configurations significantly outperformed Grid Search on all five metrics ($p < 0.01$). For the RF GA-vs-Grid-Search comparison on F1-score, the test statistic was $W = 3$ ($p = 0.002$); the ACO-vs-GA comparison yielded $W = 8$ ($p = 0.037$), indicating a smaller yet consistently reliable advantage between the two evolutionary strategies.

4 DISCUSSION

The results showed that optimizing hyperparameters using evolutionary algorithms provides a significant advantage in the detection of kidney disease. ACO demonstrated a consistent advantage over GA across all ten experimental runs, particularly in sensitivity and AUC-ROC for the CKD-positive class. This outcome aligns with documented behaviour of pheromone-based search in structured parameter spaces: Dorigo et al. [16] established that... while Di Francesco Marino et al. [20], [21] showed that ACO-guided tuning surpasses GA precisely when the fitness landscape contains densely packed near-optimal solutions a pattern that appears consistent with the CKD hyperparameter topology observed here [22], [23]. Akhatkulov and Yalgoshev [24] similarly reported that iterative population-based optimisation outperforms manual and random search in time-series prediction tasks, supporting the generalisability of this finding beyond classification problems. However, it is important to note that evolutionary approaches also have their own limitations. First, the genetic algorithm and ACO algorithms themselves have a number of hyperparameters, the wrong choice of which can reduce search efficiency. Second, as the data volume and model complexity increase, the time and computational cost of the optimization process can increase significantly. Third, these results are based on a specific data set, requiring further clinical validation before the model can be applied to other populations. At the same time, it is important to note

that we believe that this additional computational cost is justified by the increased sensitivity in CKD diagnostics, which increases the possibility of early detection of the disease and reducing the risk of complications. We highlight the possibility that the use of evolutionary optimized ensemble models as an additional advisory system for physicians can effectively support the traditional diagnostic process, especially in resource-limited areas.

5 CONCLUSIONS

This study systematically examined hyperparameter optimization strategies for machine learning classifiers applied to early CKD detection, with a particular focus on two evolutionary meta-heuristics: Genetic Algorithm and Ant Colony Optimization. Seven classifiers in total - the five baseline models (LR, kNN, SVM, Decision Trees, and Naïve Bayes) evaluated under default settings, plus the two ensemble models (Random Forest and XGBoost) evaluated under five tuning configurations - default settings, Grid Search, Random Search, GA, and ACO - and performance was measured across accuracy, sensitivity, specificity, F1-score, and AUC-ROC using ten repeated stratified splits. The experimental findings consistently demonstrated the superiority of evolutionary strategies over default configurations and traditional search techniques. ACO-optimized variants achieved the highest overall AUC-ROC and clinical sensitivity scores. Crucially, both meta-heuristics provided a highly favorable accuracy-to-runtime trade-off compared to exhaustive Grid Search, confirming their practical viability for real-time medical decision-support systems. These results highlight that evolutionary optimization is not merely a theoretical alternative to grid-based search but a practically viable and medically relevant tool. The pheromone reinforcement mechanism in ACO proved particularly effective at navigating large mixed-type hyperparameter spaces, consistently outperforming GA in terms of AUC-ROC and sensitivity at a modest additional computational cost. Several limitations should be acknowledged. The study was conducted on a single, relatively small dataset, and the degree to which these findings extend to larger multi-center cohorts remains to be established. The evolutionary algorithms themselves carry their own hyperparameters - population size, generation count, crossover and mutation rates for GA; ant count, evaporation rate, and pheromone weight for ACO - and a sensitivity analysis of these meta-level settings was beyond the scope of the current work. Future research should explore particle swarm optimization, differential evolution, and hybrid meta-heuristics on larger clinical datasets, as well as develop computationally lighter surrogate-assisted variants that retain the search quality of ACO while reducing wall-clock time.

REFERENCES

[1] M. M. Talipov, "Computational modeling and analysis of mechanical power consumption in train assemblers' work," in Proc. Int. Conf. on Applied Innovations in IT (ICAIIT), vol. 13, no. 2, pp. 419-426, 2025, [Online]. Available: https://doi.org/10.25673/120513.

[2] M. Shukurova, K. Ruziev, M. Talipov, and G. Talipova, "3D modeling of filtration in oil and gas reservoirs based on satellite data," Proceedings of International Conference on Applied Innovation in IT, vol. 14, no. 1, pp. 267-271, Mar. 2026, [Online]. Available: https://doi.org/10.25673/123569.

[3] M. Shukurova, M. Talipov, K. Ruziev, and K. Jurayeva, "Advanced geospatial monitoring of oil and gas infrastructure via satellite data," Mathematical Models in Engineering, vol. 12, no. 2, pp. 190-201, Jun. 2026, [Online]. Available: https://doi.org/10.21595/mme.2026.25328.

[4] A. Mamatov, X. Sotvoldiyev, M. Talipov, S. Shaumarov, K. Gafarbayli, and D. Bekmirzaev, "Metrological calibration and uncertainty evaluation of MEMS accelerometers for structural health monitoring in seismic regions," Vibroengineering Procedia, vol. 62, pp. 140-144, Jun. 2026, [Online]. Available: https://doi.org/10.21595/vp.2026.26360.

[5] K. Laxmikant, R. Bhuvaneswari, and B. Natarajan, "An Efficient Approach to Detect Diabetes using XGBoost Classifier," in 2023 Winter Summit on Smart Computing and Networks (WiSSCoN), Chennai, India, 2023, pp. 1-8, [Online]. Available: https://doi.org/10.1109/WiSSCoN56857.2023.10133854.

[6] S. A. Tiruneh, T. T. T. Vu, D. L. Rolnik, H. J. Teede, and J. Enticott, "Machine Learning Algorithms Versus Classical Regression Models in Pre-Eclampsia Prediction: A Systematic Review," Current Hypertension Reports, vol. 26, no. 7, pp. 309-323, Jul. 2024, [Online]. Available: https://doi.org/10.1007/s11906-024-01297-1.

[7] R. Hephzibah, A. H. Christinal, R. Jayanthi, D. A. Chandy, and C. Bajaj, "A Novel Ensemble Classifier Framework for Accurate Fetal Heart Rate Classification," in 2023 4th International Conference on Signal Processing and Communication (ICSPC), Coimbatore, India, 2023, pp. 321-324, [Online]. Available: https://doi.org/10.1109/ICSPC57692.2023.10125713.

[8] N. Fayzullo, S. Sariyev, and Y. Sherzodjon, "Analyzing the Effectiveness of Ensemble Methods in Solving Multi-Class Classification Problems," in 2025 International Russian Smart Industry Conference (SmartIndustryCon), Sochi, Russian Federation, 2025, pp. 788-793, [Online]. Available: https://doi.org/10.1109/SmartIndustryCon65166.2025.10986248.

[9] S. Pawar, P. Bagal, P. Shukla, and A. Dawkhar, "Detection of Breast Cancer using Machine Learning Classifier," in 2021 Asian Conference on Innovation in Technology (ASIANCON), Pune, India, 2021, pp. 1-5, [Online]. Available: https://doi.org/10.1109/ASIANCON51346.2021.9544767.

[10] T. Chen and C. Guestrin, "XGBoost: a scalable tree boosting system," in Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), New York, NY, USA, 2016, pp. 785-794.

[11] L. Gosink, et al., "Characterizing and Visualizing Predictive Uncertainty in Numerical Ensembles Through Bayesian Model Averaging," IEEE Transactions on Visualization and Computer Graphics, vol. 19, no. 12, pp. 2703-2712, Dec. 2013, [Online]. Available: https://doi.org/10.1109/TVCG.2013.138.

[12] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," Annals of Statistics, vol. 29, pp. 1189-1232, 2001, [Online]. Available: https://doi.org/10.1214/aos/1013203451.

[13] H. Alibrahim and S. A. Ludwig, "Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization," in 2021 IEEE Congress on Evolutionary Computation (CEC), Kraków, Poland, 2021, pp. 1551-1559, [Online]. Available: https://doi.org/10.1109/CEC45853.2021.9504761.

[14] S. Sariyev, G. Negmatova, and A. Sayidkulov, "Preparation Datasets Based on Artificial Intelligence Models and Classification Algorithms," in 2025 International Russian Automation Conference (RusAutoCon), Sochi, Russian Federation, 2025, pp. 260-266, [Online]. Available: https://doi.org/10.1109/RusAutoCon65989.2025.11177339.

[15] S. Akhatkulov, I. Yalgoshev, and Z. Urinboyev, "Vehicle CO2 Emission Prediction Using Deep Learning and Ensemble Machine Learning Methods," in 2025 International Russian Automation Conference (RusAutoCon), Sochi, Russian Federation, 2025, pp. 819-824, [Online]. Available: https://doi.org/10.1109/RusAutoCon65989.2025.11177377.

[16] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28-39, Nov. 2006, [Online]. Available: https://doi.org/10.1109/MCI.2006.329691.

[17] X. Xu, J. Peng, W. Zhao, W. Nai, Z. Yang, and Y. Xing, "Lasso Regression Based on t-Distribution Coot Optimization Algorithm," in 2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC), Chongqing, China, 2022, pp. 1920-1924, [Online]. Available: https://doi.org/10.1109/ITOEC53115.2022.9734538.

[18] S. Sariyev, I. Yalgoshev, and M. Nuriddinova, "Traditional Methods and Modern Approaches Based on Ensemble Algorithms for Decision-Making in Diagnostics Using Medical Data," in 2025 International Russian Automation Conference (RusAutoCon), Sochi, Russian Federation, 2025, pp. 971-975, [Online]. Available: https://doi.org/10.1109/RusAutoCon65989.2025.11177423.

[19] A. Taufiq, S. Yulianti, A. Rahmatulloh, I. Darmawan, and R. Rizal, "Comparison of Hyperparameter Tuning Techniques on KNN Algorithm to find the Best K Value using Grid Search and Random Search Methods," in 2024 7th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2024, pp. 180-186, [Online]. Available: https://doi.org/10.1109/ISRITI64779.2024.10963519.

[20] C. Di Francescomarino, et al., "Genetic algorithms for hyperparameter optimization in predictive business process monitoring," Information Systems, vol. 74, pp. 67-83, 2018, [Online]. Available: https://doi.org/10.1016/j.is.2018.01.003.

[21] E. Zarate-Perez and R. Sebastián, "Optimizing the sizing of residential microgrids using a genetic algorithm as a decision support model," Management of Environmental Quality, vol. 37, no. 2, pp. 336-357, 2025, [Online]. Available: https://doi.org/10.1108/MEQ-01-2025-0043.

[22] A. S. Wicaksono and A. A. Supianto, "Hyper Parameter Optimization Using Genetic Algorithm on Machine Learning Methods for Online News Popularity Prediction," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 9, no. 12, pp. 263-267, 2018, [Online]. Available: https://doi.org/10.14569/IJACSA.2018.091238.

[23] E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis," Informatics, vol. 8, no. 4, Art. no. 79, Nov. 2021, [Online]. Available: https://doi.org/10.3390/informatics8040079.

[24] S. Akhatkulov and I. Yalgoshev, "Air quality prediction based on machine learning techniques," AIP Conference Proceedings, vol. 3356, no. 1, Art. no. 030005, Sep. 2025, [Online]. Available: https://doi.org/10.1063/5.0296121.