

An Environmental Sustainable Approach to Machine Learning, Training and Development

K. Jegadeeswari¹ , R. Rathipriya^{2*} 

¹Department of Computer Science, Periyar University, Salem, Tamilnadu, India, ror.org/05crs8s98

²Department of Computer Science, Periyar University, Salem, Tamilnadu, India, ror.org/05crs8s98

Corresponding author:

R. Rathipriya, Department of
Computer Science, Periyar University,
Salem, Tamilnadu, India
rathi_priyar@periyaruniversity.ac.in



Article History:
Received: 24.03.2025
Revised: 08.06.2025
Accepted: 23.06.2025
Published Online: 26.09.2025

ABSTRACT

Artificial intelligence has the potential to drive sustainability by minimizing the impact of machine learning (ML) development on the environment. However, many ML techniques, particularly ensemble methods like the Random Forest classifier, require large computational resources during the tuning of hyperparameters. These hyperparameters are the number of trees, the depth of the tree, and the number of features considered at each split of the tree. These hyperparameters considerably impact model performance and energy consumption. This paper proposes an eco-friendly multi-objective framework (EFMOF) to optimize the hyperparameters with minimal environmental impact while retaining high model accuracy. By leveraging advanced hyperparameter optimization techniques like Optuna, Hyperopt, and Grid Search, the framework effectively explores the hyperparameter space, focusing on energy efficiency and carbon reduction. From the above, incorporating sustainable AI into ML development requires monitoring energy consumption and carbon emissions at every hyperparameter tuning. This will ensure that the models developed perform well and are sustainable without too much environmental cost. The Experimental result shows that the most dominant hyperparameter is the number of estimators, which leads to higher energy consumption. In contrast, minimum samples per leaf and split have a moderate effect, while maximum depth has a minor impact.

Keywords: Multi-objective, Ensemble Classification, Hyperparameter Optimization, Eco-friendly, Sustainable ML.

1. Introduction

Machine learning (ML) models have emerged as crucial tools in improving accurate outcomes across many domains. Recent studies have demonstrated that Random Forest (RF) [1] provides superior predictions for medical data diagnostics, prediction of climate change, improved automation, personalized treatment plans, etc. Hence, this paper specifically focuses on applying RF to classify breast cancer data [2]. The RF methodology considers the generation of several decision trees during training; it classifies the mode of classes as output from individual trees. This ensemble avoids overfitting, a common issue with individual decision trees and supports better generalization to new unseen data. In the healthcare context, where predictive accuracy can have critical implications, the reliability of RF models makes them a valuable tool for tasks such as disease classification. However, the performance of Random Forest is highly dependent on tuning its hyperparameters [3]. The most important factors here are the number of trees: number of estimators, the depth of trees: maximum no of depth, and the number of features tested for splitting: maximum features. To get the optimum performance, these have to be tuned. To find the best combination of these hyperparameters, GridSearch (GS) and Random Search are generally used, along with more advanced methods such as Bayesian optimization, Optuna and Hyperopt. In some cases, these optimization techniques efficiently explore the parameter space and point toward the model configurations that yield high accuracy while avoiding overfitting. As more applications adopt machine learning, the computational cost for training and optimizing models has increased. This cost basically translates into increased energy consumption, as noted in [4], directly impacting carbon dioxide (CO₂) emissions and the environment. Large energy requirements in training machine learning models, notably in extensive hyperparameter searches, are now a point of concern for their impact on the environment. Therefore, measuring the CO₂ emissions and emission control is essential during training the ML model. Python offers a tool called CodeCarbon, which tracks the model's energy consumption and calculates its corresponding CO₂ emissions. By plugging CodeCarbon into the machine learning pipeline, researchers and developers can measure the energy consumption and its corresponding CO₂ emissions generated by their experiments [5].

Integrating environmental considerations into the optimization process thus constitutes a major step toward Sustainable AI. It corresponds to international proposals for reducing CO₂ emissions while allowing the development of sustainable technological practices. In this context, this research proposed an eco-friendly multi-objective framework (EFMOF). This framework used a multi-objective strategy to optimize the RF classifier hyperparameters to prevent the overfitting problem for breast cancer classification and mitigate the carbon emissions. It applies three different hyperparameter optimization techniques: Optuna, Hyperopt, and GridSearch. The CodeCarbon tool is incorporated into monitoring and measuring CO₂ emissions during hyperparameter tuning.

1.1. Motivation

Most classical methods of hyperparameter optimization consider the goal of model performance improvements, completely or mostly neglecting the environmental costs of training. As machine learning applications increase in scale, the need for eco-efficiency in model optimization becomes urgent. Only by considering sustainability in the scope of optimization can the accuracy and ecological impact of machine learning models be improved. At the same time, Breast cancer continues to pose a considerable health challenge worldwide, underscoring the importance of timely and precise diagnosis for the facilitation of effective treatment and the enhancement of patient outcomes. These factors motivate integrating ML, healthcare and environmental sustainability into a single framework.

1.2. Problem Statement

The RF classification model's optimization process for breast cancer prediction must consider multi-objective functions involving accuracy vs CO₂ emissions. This has resulted in the increasing demand for methods that, besides optimizing performance, consider the environmental footprint of the hyperparameter-tuning processes.

1.3. Objective

The objective is to develop an Eco-friendly multi-objective optimization framework (EFMOF) for boosting the accuracy of breast cancer classification using the Random Forest approach by balancing the accuracy against carbon emissions, ensuring sustainable machine learning.

1.4. Contribution

This subsection explores the key contributions of the proposed framework. It is listed as follows:

- Mitigating Overfitting/Underfitting via Hyperparameter Tuning of RF models
- Tracking CO₂ emissions of the RF model with a different set of hyperparameter values using the CodeCarbon package
- Normalizing the metrics, such as accuracy and emissions of the RF models, for Fair Comparison
- SoftMax-Based Adaptive Weighting of the Metrics
- Pareto Optimality for Multi-Objective Selection

The paper's remaining structure is as follows: Section 2 provides a literature review. Section 3 explains the RF algorithm, various optimization techniques and the proposed EFMOF designed to reduce the resource consumption during RF model training. Section 4 overviews experimental results and discusses how hyperparameters increase CO₂ emissions. Finally, Section 5 presents a conclusion and future directions.

2. Related work

This section examines the existing relevant methodologies in detail and highlights the strengths and weaknesses of current approaches. Finally, it discusses the identified research gap.

Zheng et al. [6] reported a comparative study of breast cancer diagnosis among different classifiers and the Random Forest (RF) classifier. The results demonstrated that RF provided high accuracy and robustness, mainly combined with hyperparameter optimization techniques like GridSearch. However, the study did not consider the environmental impact of the optimization process, leaving a gap that your proposed work aims to address. Delen et al. [7] employed ensemble learning methodologies, namely Random Forest (RF), to predict breast cancer recurrence. The study emphasized the importance of model interpretability in healthcare applications, in which RF shines due to its decision tree-based architecture. The authors indicated further work on these models was needed toward improved performance, and this aligns with the eco-efficient concepts within your framework. Zizaan et al. [8] explored ensemble techniques, boosting, and bagging to predict breast cancer. This study showed that ensemble methods outperform individual classifiers, particularly when dealing with imbalanced datasets- a common scenario in medical diagnostics. However, it also pointed out the high computational cost associated with such methods, thus calling for environmentally friendly model-training strategies. Jegadeeswari et al. [9] developed an optimized stacking ensemble classifier model to identify ovarian cancer at an early stage using biomarkers. The Authors used PSO for the base learners' hyperparameter optimization and GridSearch for the meta learner's hyperparameter

optimization. However, neither learner considered environmental impact and sustainability during hyperparameter tuning, but this model achieved 94% accuracy.

Akiba et al. [10] introduced Optuna in 2019, a hyperparameter optimization framework featuring a good balance of exploration and exploitation in search spaces. Optuna can prune poorly performing trials early, reducing computational resources and environmental impact. This paper demonstrates the advanced optimization techniques that can enormously contribute to the energy footprint of machine learning models with improved performance in critical applications, like medical diagnostics. Bergstra et al. [11] developed a hyperparameter optimization tool, Hyperopt, which employed a Bayesian optimization algorithm to handle complex search spaces. The work presented determined that Hyperopt achieved higher application efficiency than the traditional lot GridSearch and Random Search. Therefore, it is an excellent tool for eco-efficient machine learning. However, it does not discuss an application in medical diagnostics, especially breast cancer diagnosis. Liu et al. [12] evaluated the performance of different hyperparameter optimization methods for medical image tasks. The results revealed that Bayesian optimization is superior to all methods concerning accuracy and computational efficiency and will be a probable solution for building eco-efficient models for breast cancer diagnosis.

Henderson et al. [13] summarized much that is known about the environmental impacts of training large-scale machine learning models. The increasing complexity and size of the models, especially those in healthcare, are some of the reasons for high energy consumption and the emissions of carbon dioxide. The paper proposes adopting more energy-efficient algorithms powered by renewable energy sources in such a scenario. This study forms an essential foundation for understanding the ecological contexts of artificial intelligence development; nevertheless, the study does not elaborate on applications that concern medical diagnostics. Strubell et al. [14] evaluated carbon emissions at different advanced natural language processing models quantitatively. They recommended methods to evaluate carbon emissions and suggested that improving model architectures and fine-tuning hyperparameters could significantly reduce energy consumption. This paper highlights the importance of eco-efficient methodologies in artificial intelligence, but it does not address, in particular, how these methodologies can be used in healthcare applications.

Table 1. Performance and Environmental Impact Comparison studies of various articles

References	High Accuracy	Environmental Consideration	Sustainable AI Practices	Healthcare Applications
Zheng et al., 2021 [6]	✓	✗	✗	✓
Delen et al., 2020 [7]	✓	✗	✗	✓
Zizaan et al., 2019 [8]	✓	✗	✗	✓
Jegadeeswari et al., 2024 [9]	✓	✗	✗	✓
Akiba et al., 2019 [10]	✓	✓	✓	✗
Bergstra et al., 2013 [11]	✓	✓	✓	✗
Liu et al., 2020 [12]	✓	✓	✓	✓
Henderson et al., 2020 [13]	✓	✓	✓	✗
Strubell et al., 2019 [14]	✓	✓	✓	✗
Lottick et al., 2020 [15]	✓	✓	✓	✓
Schwartz et al., 2020 [16]	✓	✓	✓	✗
Patterson et al., 2021 [17]	✓	✓	✓	✗

Lottick et al. [15] proposed the tool CodeCarbon for measuring carbon emissions produced by machine learning experiments. The paper demonstrated the utility of that tool in calculating the environmental impact linked to different models and training methodologies. The tool CodeCarbon, when used during hyperparameter tuning, allows one to find the most environmentally friendly ways of constructing models- exactly what this proposed framework is looking to achieve. Schwartz et al. [16] developed this notion of AI sustainability by considering how tools like CodeCarbon could be used to understand model selection and development better. The authors stated that including metrics on the environmental impact in the set of criteria used during model evaluation would foster better sustainability of AI. This study is in line with and supports the core of your framework, where the performance improvement should be done with simultaneous reduction in environmental impact.

Patterson et al. [17] explored the extensive ramifications of green AI, particularly emphasizing the necessity for established industry standards regarding the disclosure of energy usage and carbon emissions associated with machine learning models. This research offered a conceptual framework aimed at embedding sustainability within the realm of AI development, which your suggested project enhances by integrating these factors into the hyperparameter optimization process for breast cancer diagnosis. Table 1 shows the study comparison of various articles that worked with both the Sustainability AI and Healthcare domains.

A literature review demonstrated notable advancements in machine learning techniques and the use of RF for healthcare applications. At the same time, some works have also raised the concern regarding sustainable machine learning called Green AI. However, most works have not proposed a concrete framework that integrates both aspects. Despite a common limitation across these studies, there is a limited focus on the environmental impact of hyperparameter tuning. This gap highlights the importance of incorporating a sustainable approach to optimize the ML models, particularly in sensitive domains like healthcare. To address this gap, the proposed EFMOF aims to optimize the RF, enhancing the classification accuracy for breast cancer and reducing the CO₂ footprint. Aligning the sustainability goal with the model performance ensures a step toward Green AI practices in healthcare.

3. Materials and methods

The proposed work is to develop an eco-friendly multi-objective framework (EFMOF). This framework is designed to optimize the performance of an RF classifier for breast cancer classification while minimizing the CO₂ emissions.

3.1. Random Forest

RF is one of the promising ensemble learning methods that does not guarantee a linear relationship among features. It is suitable for capturing complex interactions in biological and clinical settings. Another important advantage of RF is that it effectively handles outliers in the dataset, due to its ability to reduce variance. RF constructs multiple decision trees during training and outputs the mode of the classes predicted by individual trees [18]. Each tree in the forest is trained on a randomly selected subset of the dataset. The subset includes randomly selected patient records and random samples of features. The final classification result is determined through majority voting, which is based on aggregating the predictions of all trees in the forest.

Let $T_1(x), T_2(x), \dots, T_n(x)$ be the output from n individual trees for an input sample x . The final predicted class \hat{y} is

$$\hat{y} = \text{mode}(T_1(x), T_2(x), \dots, T_n(x)) \quad (1)$$

In this way, a common drawback of RF is that all decision trees are trained independently using a standard default parameter setup. which leads to the overfitting problem, cannot produce good results, leads to poor generalization to unseen test data and produces bias in predictions. Therefore, hyperparameter tuning is necessary to prevent overfitting problems and improve the model's performance.

3.2. Hyperparameter Tuning Techniques

To achieve optimal performance of the RF classifier, carefully tune the hyperparameters such as $n_estimators$, max_depth and $max_Features$. This paper uses three hyperparameter optimization techniques to fine-tune the RF model.

3.2.1. Optuna

TPE is the default optimization algorithm in Optuna [19]. It places the objective function in probabilistic models using those probabilistic models to derive new values for hyperparameters. It generates new hyperparameter values by sampling from these distributions to focus on promising regions of the search space. The model is updated iteratively based on observed results. TPE is effective for high-dimensional spaces and complex objective functions.

$$x^* = \arg \min_{x \in \mathcal{X}} f(x) \quad (2)$$

Where $f(x)$ is the objective function, x is the hyperparameter space.

3.2.2. Hyperopt

Hyperopt is a Bayesian optimization-based hyperparameter tuning technique [20]. New hyperparameters from regions where the expected improvement is high, a technique aligned with Bayesian optimization principles. The probabilistic approach allows Hyperopt to efficiently explore and exploit the hyperparameter space, particularly in complex and high-dimensional scenarios. There are two conditional probabilities, such as good and bad hyperparameters.

Good hyperparameters - $l(x) = p(x|y < y^*)$ – the probability distribution over the hyperparameter values that led good (low loss) results. In this case Hyperopt wants to explore more.

Bad hyperparameters - $g(x) = p(x|y \geq y^*)$ – distribution over the hyperparameter that resulted in bad (higher loss) results. In this case, Hyperopt avoids sampling from this distribution.

Then it selects new hyperparameters by finding values of x that maximize the expected improvement using the radio.

$$x^* = \arg \max_x \frac{l(x)}{g(x)} \quad (3)$$

Where x is a hyperparameter configuration, y is the output of the objective function, which is the loss or error score. If using accuracy, $y = 1 - \text{accuracy}$. y^* is a threshold that separates good and bad trails (eg, best 20%). Here, the lowest loss score is considered the best-performing hyperparameters. x^* helps Hyperopt choose values from regions with high performance.

3.2.3. GridSearch

It systematically explores all the possible combinations (brute force) in a given grid of hyperparameters [21]. This process aims to find the combination that yields the best performance for a given metric. GridSearch can be resource-intensive, especially in large hyperparameter spaces or complex datasets. Combinations grow exponentially when one tries to extend the number of hyperparameters and their different values.

$$\text{Total Combinations} = \prod_{i=1}^p n_i \quad (4)$$

Where p is a hyperparameter, each with n alternative values, it evaluates the objective function $f(x)$ as error or accuracy.

While fine-tuning the RF classifier with the help of these optimization techniques can significantly improve the model accuracy and reduce overfitting, it also increases the computational complexity. This is because $n_{estimators}$ is determined by the number of trees in the forest. max_depth is the maximum depth of each tree, which controls the complexity of the model. $max_features$ are the number of features considered for splitting at each node that affect the range and performance of the model [22]. All these directly impact how large and deep the forest grows; additionally, forest growth depends on how much data it processes at each node. In this context, consider an example: suppose training an RF model with a breast cancer dataset (569 samples and 30 features), here if we use 10 trees, each with a maximum depth of 5, and consider 5 features at each split. This model might complete its training in a few seconds with minimal energy use. However, the search space for tuning should be increased, such as the number of trees from 50 to 300, each with a maximum depth of 5 to 30 and a maximum of 5 to 30 features. The total number of models trained during optimization increases dramatically. Depending on the system and data, this can take several hours of total runtime. The optimization process increases as the search space expands, leading to high carbon dioxide emissions. This time and energy consumption (resource-intensive) raises environmental sustainability concerns [23]. Measuring CO₂ emissions and emission control is essential to maintain sustainability while training the RF model. CodeCarbon is a special package included in the code to track time consumption and measure CO₂ emissions during model hyperparameter tuning.

3.3. CodeCarbon package

CodeCarbon helps track the energy of developed code uses and calculate the carbon dioxide (CO₂) emissions with corresponding energy use [24]. Initializing CodeCarbon using its EmissionTracker starts by capturing the starting time of RF code execution and monitoring the entire duration of RF hyperparameter tuning during training. Simultaneously, it monitors what system resources, such as CPU, GPU, and RAM, are being utilized. This data is used to estimate the energy usage (in Wh) of the system consumed during the process. Calculate the CO₂ emissions accurately. CodeCarbon should take into account the system's geographical location. CO₂ intensity varies depending on the type of energy source. Energy sources differ from location to location due to various geographical natural resources such as sunny, river, mountainous, volcanic, coastal, wind and fossil fuels [25]. If electricity comes from renewable sources such as solar or wind, the CO₂ intensity is lower than that of fossil fuels like coal, oil or natural gas. If the system location is not specified, CodeCarbon uses a default global average CO₂ intensity value of approximately 0.475 g CO₂/Wh.

$$CO_2 \text{ emission (in grams)} = \text{Energy used (Wh)} \times \text{Carbon Intensity (g CO}_2\text{/Wh)} \quad (5)$$

This paper introduces EFMOF, which is designed to minimize environmental impact. The RF classifier is optimized multi-objectively to optimize its hyperparameters by combining the dual optimization objective of maximizing classification accuracy while minimizing environmental impact. This framework will be injected directly into the optimization cycle of three hyperparameter optimization techniques to meet the eco-friendly requirements. The emission tracker of the CodeCarbon package tracks CO₂ emissions of different hyperparameter settings. After that, it functions as a multi-objective function that contains the measured carbon and the accuracy of the appropriate hyperparameter set. Normalize the multi-objective function and apply a softmax adaptive weight to the multi-objective function. Update the search space and obtain new observations (accuracy and carbon emissions) of that parameter set. Applying Pareto optimality to select the best hyperparameter set. When the Pareto optimality condition is satisfied, the corresponding hyperparameter set is selected; otherwise, this process is repeated until convergence, and the best hyperparameter set is finally applied to the unseen test data.

3.1. Proposed Work: Eco-friendly multi-objective framework (EFMOF)

The eco-friendly multi-objective framework (EFMOF) seeks to optimize hyperparameters h by considering the model's accuracy and CO₂ emissions (or computational cost). This framework defines two key objective functions: A (h) and E(h).

$A(h)$ represents the accuracy, and $E(h)$ represents the emissions for a given hyperparameter set h . These objectives are combined into a single loss function.

The adaptive weights α and β are assigned to accuracy and emissions, respectively. The overall objective function is expressed as

$$\mathcal{L}(h) = \alpha(h).A(h) - \beta(h).E(h) \quad (6)$$

Accuracy and CO₂ emissions are measured in different units. To combine them manually in a single loss function, they must be brought to the same scale. Normalizing ensures that both metrics contribute fairly. To ensure accuracy and emissions are comparable, both are normalized. The normalized accuracy $A'(h)$ is defined as

$$A'(h) = \frac{A(h)-A_{min}}{A_{max}-A_{min}} \quad (7)$$

Where $A(h)$ is the model's accuracy trained with Hyperparameters h . A_{max}, A_{min} are the maximum and minimum accuracies observed across all hyperparameter sets. $A'(h)$ is scaled to be a value between 0 and 1. Similarly, normalized emission $E'(h)$ is defined as

$$E'(h) = \frac{E(h)-E_{min}}{E_{max}-E_{min}} \quad (8)$$

Where $E(h)$ is carbon emissions (in grams of CO₂) produced during training with hyperparameters h . E_{max}, E_{min} are high and low CO₂ emissions values recorded by the CodeCarbon tool. $E'(h)$ value scaled between 0 and 1. Update the Objective function with adaptive weights.

$$\mathcal{L}(h) = \alpha(h).A'(h) - \beta(h).E'(h) \quad (9)$$

Where $\alpha(h)$ and $\beta(h)$ are dynamically changing their weights as a function of current hyperparameter performance. SoftMax-based adaptive weights are used here. This mechanism keeps both objectives bound and ensures smooth adaptation over time.

$$\alpha(h) = \frac{e^{\gamma A'(h)}}{e^{\gamma A'(h)} + e^{\gamma(1-E'(h))}} \quad (10)$$

$$\beta(h) = 1 - \alpha(h) \quad (11)$$

Where γ is a sensitivity factor (range between 5-10), a higher value makes the adaptation sharper.

Implementation of multi-objective optimization by using Pareto optimality, a new solution h is considered to dominate another solution h^* if it is achieving high accuracy without increasing carbon emissions or it results in lower emissions without compromising accuracy. In other words, if no other solution h exists that improves one objective without worsening the other. Specifically, h^* is Pareto optimal if

$$\begin{aligned} A(h) > A(h^*) \text{ and } E(h) \leq E(h^*) \quad \text{or} \\ A(h) \geq A(h^*) \text{ and } E(h) < E(h^*) \end{aligned} \quad (12)$$

Where (h^*) is present best hyperparameter set in the pareto optimal set. Applying Pareto optimality in the proposed framework ensures a fair and effective balance between RF performance and environmental sustainability. It helps identify the best compromising solutions rather than the standard parameter set.

Once identifying a $P = h_1^*, h_2^*, \dots, h_k^*$ be a set of pareto optimal hyperparameter configuration, each corresponding to a trained RF classifier. The final prediction y^{\wedge} is determined by majority voting across the multiple decision trees.

$$y^{\wedge} = \text{mode}(\{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_k\}) \quad (13)$$

Where mode is selected the most frequently predicted class label among all \hat{y}_i, i is determined $1, 2, \dots, k$. y^{\wedge} is the final classification output for the input. The final evaluation of the RF considers both accuracy and emissions. The ensemble accuracy is calculated as

$$A = \frac{1}{n} \sum_{j=1}^n 1(y_j^{\wedge} = y_j) \quad (14)$$

Where n is total number of test samples, y_j is true class label y_j^{\wedge} is the predicted class label for j^{th} sample. $1(\cdot)$ is the indicator function. If the prediction is correct which is equal to 1 otherwise 0. The total emissions are the sum of the emissions of the Pareto optimal hyperparameter configuration.

$$E = \sum_{i=1}^k E(h^*_i) \quad (15)$$

where k is the number of Pareto optimal hyperparameters. $E(h^*_i)$ represents the measured CO₂ emissions (in grams) generated during training with hyperparameter configuration h^*_i . Equation 12 helps to evaluate the eco-friendliness of the framework by keeping track of the environmental sustainability during RF development.

Algorithm 1: Eco-Friendly Multi-Objective Framework (EFMOF)

Input: Dataset D

Output: Optimized hyperparameter set X^* , Final model performance and carbon emissions

1. Load Dataset D
2. Preprocessing:
 - a. Handle missing values using the CRP model [26, 27]
 - b. Encode target variable y (benign $\rightarrow 0$, malignant $\rightarrow 1$)
 - c. Split D into 80% training and 20% testing sets
3. Define objectives:

$A(h) \leftarrow$ Accuracy

$E(h) \leftarrow$ Carbon Emissions

Objective: Maximize $A(h)$ and minimize $E(h)$ simultaneously
4. Define hyperparameter space $X = [n_estimators, max_depth, min_samples_split, min_samples_leaf]$
5. For each hyperparameter set X :
 - a. Train RF model
 - b. EmissionTracker $E(X)$
 - c. Compute carbon emissions $C(X) = E(X) \times$ Emission Factor (0.475 kg CO₂/kWh)
6. Combine objectives into a single objective function $\mathcal{L}(X)$ in Eq. 6

Normalize objectives in Eq. 9
7. Hyperparameter Optimization:
 - a. Using Hyperopt:
 - Model $\mathcal{L}(X)$ with probabilistic approach
 - Apply SoftMax adaptive weights to the objective function in Eq. 10, 11
 - $X_{new} = arg_{X \in \mathcal{X}}^{max} \mathcal{L}(X)$
 - Apply Pareto optimality in Eq. 12
 - Repeat until convergence
 - b. Using Optuna:
 - Model $\mathcal{L}(X)$ with TPE
 - Apply SoftMax adaptive weights to the objective function in Eq. 10, 11
 - $X^* = arg_{X \in \mathcal{X}}^{min} \mathcal{L}(X)$
 - Apply Pareto optimality in Eq. 12
 - Repeat until convergence
 - c. Using GridSearch:
 - Evaluate $\mathcal{L}(X)$ at each grid point
 - Apply SoftMax adaptive weights to the objective function in Eq. 10, 11
 - Select $X^* = arg_{X \in Grid}^{min} \mathcal{L}(X)$
 - Apply Pareto optimality in Eq. 12
8. Final Evaluation:
 - Apply X^* to unseen test data
 - Evaluate final Accuracy and Carbon Emissions

Return: Optimal hyperparameter set X^* , Classification accuracy, and Carbon emissions

4. Results and Discussion

4.1. Dataset Description

The Breast Cancer dataset contains 569 instances with 31 attributes each, one of which is the target variable describing diagnosis as benign or malignant. Among these, there are 212 benign and 357 malignant cases. Of all the data, 456 instances are for training, and 113 are for testing. Most of the features in the data are numeric. The features include breast cancer perimeter, radius, area, concave points, texture, smoothness, symmetry, concavity, fractal dimension, and compactness. Each of these 10 features has three statistical values: mean, standard error and worst. Mean radius and mean texture features only have 57 missing values each, while the remaining 28 features have no missing data—114 missing values out of 17070 (569 X 30) data points in the breast cancer dataset. Figure 1 represents the missingness with respect to the features.

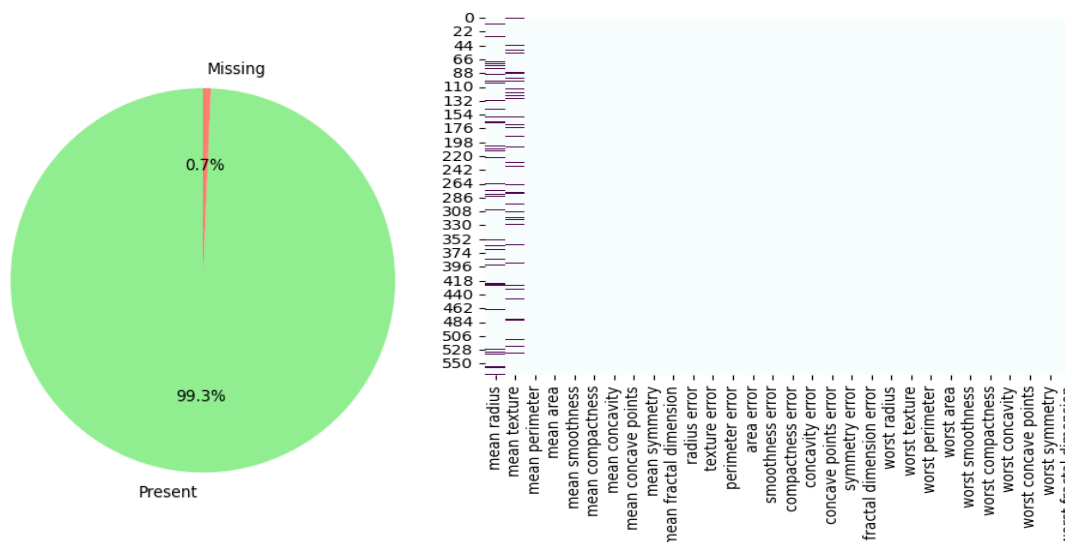


Figure 1. Breast cancer dataset missingness details

The experiment was conducted on Google Colab, utilizing an Intel(R) Xeon® CPU @2.20GHz virtual machine. Luckily, the system provided access to 2 CPU cores and 12.68 GB of RAM, running on a Linux platform (kernel version 6.1.85+). Python 3.10.12 and CodeCarbon package version 2.0.6 were used to estimate CO₂ emissions. Emission data were recorded to a local file (emission.csv). The experimental results were derived in two ways: before and after hyperparameter optimization of RF, which was applied to the breast cancer dataset. Initially, a standard RF model (pre hyperparameter optimization) was trained using default parameter as $n_estimators = 100$, $max_depth = None$, $min_samples_split = 2$ and $min_samples_leaf = 1$. Then, the hyperparameter optimization process was performed using Optuna, Hyperopt, and GridSearch techniques. The hyperparameters under consideration included $num_estimators$, $maximum_depth$, $minimum_samples_split$, and $minimum_samples_leaf$. The goal was to identify the best hyperparameter set that maintains high accuracy while minimizing emissions. Hyperparameter setups are presented in Table 2. Figure 2 shows the tracking of energy consumption using CodeCarbon.

Table 2. RF hyperparameter setting

Hyperparameters Names	Settings
No of estimators	50, 100, 200
Maximum depth	10, 15,20
Minimum no of samples split	2, 20
Minimum no of samples in leaf	1, 20

```
[codecarbon WARNING @ 04:43:40] We saw that you have a Intel(R) Xeon(R) CPU @ 2.20GHz but we don't know it. Please contact us.
[codecarbon INFO @ 04:43:40] CPU Model on constant consumption mode: Intel(R) Xeon(R) CPU @ 2.20GHz
[codecarbon INFO @ 04:43:41] >>> Tracker's metadata:
[codecarbon INFO @ 04:43:40] Platform system: Linux-6.1.85+-x86_64-with-glibc2.35
[codecarbon INFO @ 04:43:40] Python version: 3.10.12
[codecarbon INFO @ 04:43:41] CodeCarbon version: 2.6.0
[codecarbon INFO @ 04:43:41] Available RAM : 12.675 GB
[codecarbon INFO @ 04:43:41] CPU count: 2
[codecarbon INFO @ 04:43:41] CPU model: Intel(R) Xeon(R) CPU @ 2.20GHz
[codecarbon INFO @ 04:43:41] GPU count: None
[codecarbon INFO @ 04:43:41] GPU model: None
ref: /usr/local/lib/python3.10/dist-packages/codecarbon/data/hardware/cpu_power.csv
[codecarbon INFO @ 04:43:41] Saving emissions data to file /content/emissions.csv
Fitting 5 folds for each of 162 candidates, totalling 810 fits
[codecarbon INFO @ 04:43:56] Energy consumed for RAM : 0.000020 kWh. RAM Power : 4.7530388832092285 W
[codecarbon INFO @ 04:43:56] Energy consumed for all CPUs : 0.000177 kWh. Total CPU Power : 42.5 W
[codecarbon INFO @ 04:43:56] 0.000197 kWh of electricity used since the beginning.
[codecarbon INFO @ 04:44:11] Energy consumed for RAM : 0.000040 kWh. RAM Power : 4.7530388832092285 W
[codecarbon INFO @ 04:44:11] Energy consumed for all CPUs : 0.000354 kWh. Total CPU Power : 42.5 W
[codecarbon INFO @ 04:44:11] 0.000394 kWh of electricity used since the beginning.
[codecarbon INFO @ 04:44:26] Energy consumed for RAM : 0.000059 kWh. RAM Power : 4.7530388832092285 W
[codecarbon INFO @ 04:44:26] Energy consumed for all CPUs : 0.000532 kWh. Total CPU Power : 42.5 W
[codecarbon INFO @ 04:44:26] 0.000591 kWh of electricity used since the beginning.
[codecarbon INFO @ 04:44:41] Energy consumed for RAM : 0.000079 kWh. RAM Power : 4.7530388832092285 W
[codecarbon INFO @ 04:44:41] Energy consumed for all CPUs : 0.000709 kWh. Total CPU Power : 42.5 W
[codecarbon INFO @ 04:44:41] 0.000788 kWh of electricity used since the beginning.
```

Figure 2. Energy Consumption of the RF model during hyperparameter tuning at training

The best-performing hyperparameter configuration of RF with Optuna based hyperparameter tuning was $num_estimators = 50, maximum_depth = 20, minimum_samples_split = 19,$ and $minimum_samples_leaf = 4,$ with an accuracy of 0.96491 while producing the minimum carbon emissions of $8.16968E-07$. Figure 3 shows the Optuna-based hyperparameter Tuning of RF associated with the CO₂ emissions.

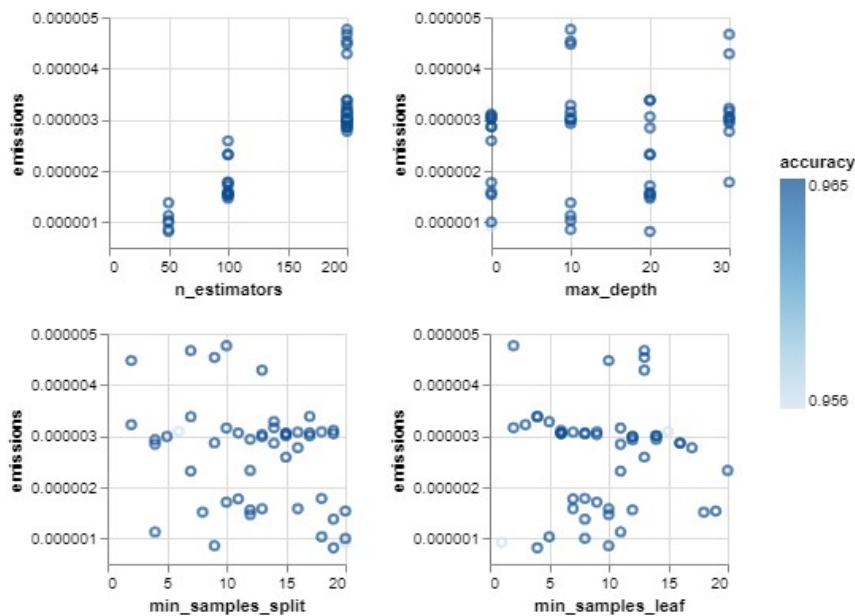


Figure 3. Random Forest Hyperparameter vs Carbon Emissions (kgco₂) using Optuna

The figure 3 consists of four scatter plots, each illustrating the relationship between a specific hyperparameter ($num_estimators, maximum_depth, minimum_samples_split,$ and $minimum_samples_leaf$) and carbon emissions, with the accuracy of each trial indicated by the color intensity of the data points. In all plots, this accuracy remained high at around 0.96, indicating that despite the mentioned hyperparameters, variations in these do not affect model performance seriously. Still, they seem to affect carbon emissions. For example, the $num_estimators$ plot shows that fewer estimators usually mean lower emissions, and max_depth shows that emissions change with tree depth, though not as dramatically. The $minimum_samples_split$ and $minimum_samples_leaf$ plots tell that small values increase emissions by forming more complex trees. Similarly, Figure 4 shows the Hyperopt-based hyperparameter Tuning of RF associated with the CO₂ emissions.

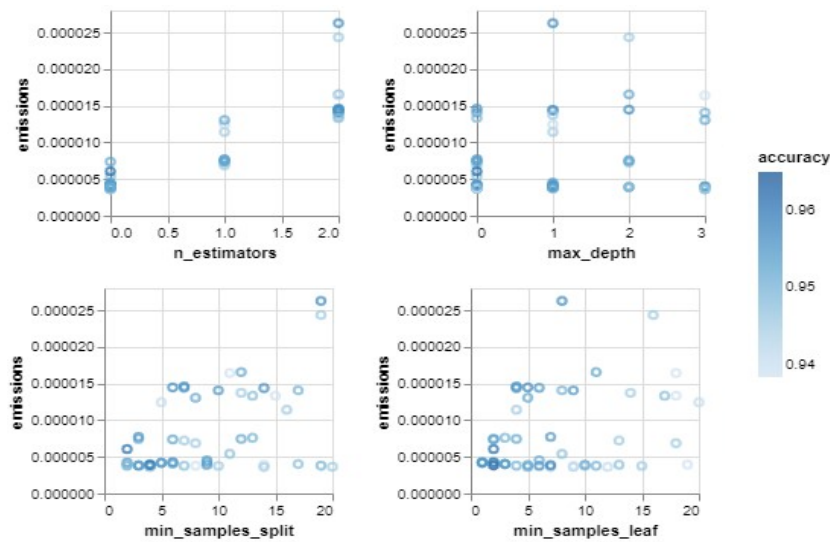


Figure 4. Random Forest Hyperparameter vs Carbon Emissions(kgCO2) using Hyperopt

The figure 4 shows the higher values of *num_estimators* generally lead to higher accuracy but at the cost of increased emissions, as seen in trials like trial 2 and trial 38. Along similar lines, raising the *maximum_depth* allowed the model to detect more complex patterns, but at the same time gives rise to higher emissions. For instance, Experiment 15 has a depth of 3, with emissions much higher than the smaller trees or with Experiment 21, which obtained good accuracy with much lower emissions. The graphs obtained for *minimum_samples_split* and *minimum_samples_leaf* show that any smaller value in those parameters is helped to create a more complex trees up to higher accuracy at the cost of emissions. The best parameter configuration identified using Hyperopt was *no of estimators* = 50, *maximum depth of the tree* = 15, *minimum no of samples split* = 4, and *minimum no of samples in leaf* = 2, achieving an accuracy of 0.964835165 with emissions recorded at 3.81874E-06. This configuration, thus, shows the importance of a trade-off between model complexity and computational effectiveness in achieving sustainable performance. Similarly, Figure 5 shows the GridSearch-based hyperparameter Tuning of RF associated with the CO₂ emissions.

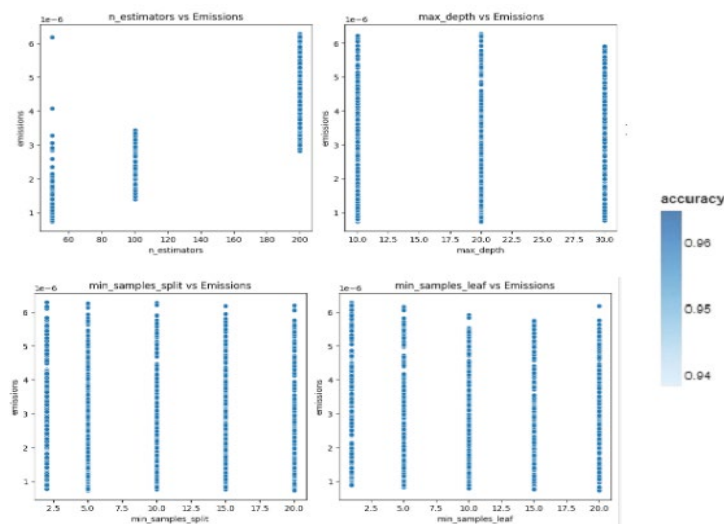


Figure 5. Random Forest Hyperparameter vs Carbon Emissions(kgCO2) using GridSearch

Figure 5 shows an examination of the *n_estimators* series, suggesting that high values usually have high emissions. Still, the accuracy increases are relatively small, considering the increased computational cost. Similarly, increasing *maximum_depth* often yields increased emissions with no commensurate increase in accuracy.

Additional inefficiencies are captured by the visual representations for *minimum_samples_split* and *minimum_samples_leaf* because GridSearch navigated many suboptimal areas of the hyperparameter landscape and, consequently, requires extreme computational investment. This brute-force search approach is one of the key contributors to the high carbon footprint observed in this experiment compared with the more targeted methods, such as Optuna and

Hyperopt. The best parameter configuration identified using GridSearch was $n_estimators = 100$, $max_depth = 20$, $min_samples_split = 2$, and $min_samples_leaf = 1$, with an accuracy of 0.96210 and emissions of 7.42019E-06.

4.2. Discussion

In this experimental study, A standard RF model was trained using default parameter as $n_estimators = 100$, $max_depth = None$, $min_samples_split = 2$ and $min_samples_leaf = 1$. This default (pre-hyperparameter optimization) configuration achieved an accuracy of 0.95034 and a carbon emission of 4.24156E-06 kgCO₂. These default parameters led to overfitting RF on the breast cancer dataset; hence, we achieved an insufficient accuracy of 95%, so we moved to hyperparameter optimization to avoid the overfitting of RF. An RF classifier was tuned using three different hyperparameter optimization techniques: optuna, Hyperopt and GridSearch and applied to the breast cancer dataset. The hyperparameters under investigation are mentioned above in Table 2. Among three optimization techniques, Optuna delivered the best result in terms of performance sustainability. Optuna's sampling strategy allowed it to converge on high-performing models with low emissions quickly. Hyperopt also produced efficient results using its TPE, leading to slightly higher emissions because it explores a wider parameter space. In contrast, GridSearch was the least efficient because it tested all possible combinations using brute force—it included several suboptimal settings that offered no performance gain but consumed more energy. Table 3 presents the best hyperparameter configurations discovered by each optimization technique associated with accuracy and CO₂ emissions. Figure 6 presents the relationship between hyperparameters and carbon emissions.

Table 3. Comparison of Best Hyperparameter Configurations and Emissions

Techniques	n_estimators	max_depth	min_samples_split	min_samples_leaf	Accuracy	CO ₂ Emissions (kg)
Optuna	50	20	19	4	0.96491	8.16968E-07
Hyperopt	50	15	4	2	0.96483	3.81874E-06
GridSearch	100	20	2	1	0.96210	7.42019E-06

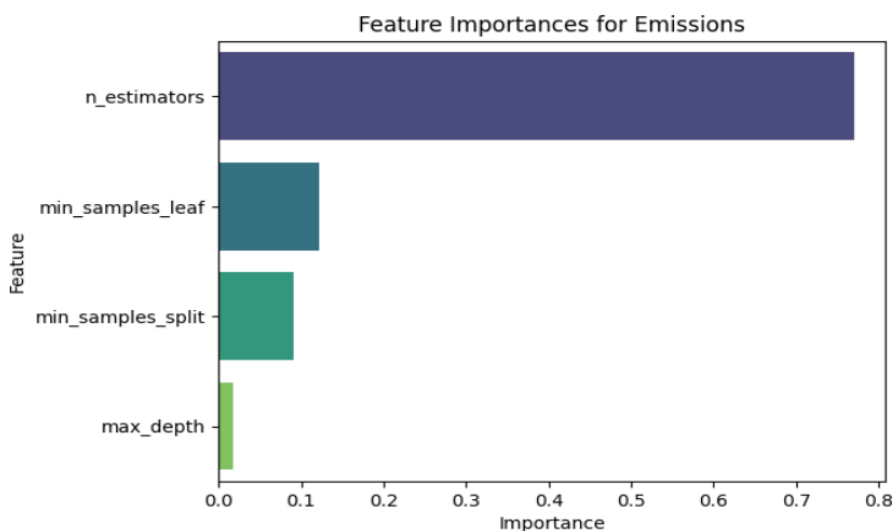


Figure 6. Feature Importance of the model for emission

Figure 6 depicts the relationship between hyperparameters and carbon emissions while training an RF model. In particular, how important is the balancing model efficacy with environmental sustainability? The most influential hyperparameters are the *num_estimators*, which means an increase in the number of trees in the RF model leads to higher energy consumption. On the other hand, *minimum_samples_leaf* and *minimum_samples_split* have a moderate effect, while *maximum_depth* has a minor one. These results underline that these parameters should be carefully tuned to lessen the releases of carbon under strict conditions without degrading model accuracy.

5. Conclusion

This paper proposed an eco-efficient multi-objective framework, which optimizes the RF classifier for breast cancer classification, while reducing carbon emissions associated with model training. The selection of Random Forest is justified by its inbuilt capability to avoid variance, handle non-linearity between data and its ensemble characteristics that integrate the outputs from numerous decision trees to improve the overall accuracy and stability of the model. These characteristics make RF very well-suited to healthcare datasets. Based on the experimental results, properly configuring the model parameters is paramount. A small change dramatically transforms the performance and environmental sustainability. The tuned results show that high classification accuracy can be achieved while keeping a low level of CO₂ emissions. Optuna

proved the optimization was superior to the other two techniques. The proposed EFMOF met the goal of sustainable AI by addressing CO₂ emissions and RF performance.

Future enhancements of EFMOF will incorporate geographical location to calculate CO₂ emissions accurately. CO₂ intensity varies with the energy source used in different regions. Additionally, experimental studies will be conducted using electricity sources to evaluate the impact of environmental sustainability on CO₂ emissions. This strategy will promote the importance of energy choice when executing a complex model or a large dataset. Moreover, it encourages the integration of Green AI into the developer code to support the eco-friendly computing environment.

References

- [1] Sharma, P., & Puri, S. "Random Forest-Based Prediction of Breast Cancer Survival: Cross-Validation and Hyperparameter Tuning," *In International Conference on Advances in Computing and Data Sciences*, 138-145. 2020. DOI: 10.1007/978-981-15-0277-0_13.
- [2] Alghamdi, F., Alsuhaibani, R., & Albattah, K. "Breast Cancer Diagnosis and Prediction Using Machine Learning and Data Mining Techniques: A Review," *IEEE Access*, 9, 18152-18164. 2021, DOI: 10.1109/ACCESS.2021.3052953.
- [3] Feurer, M., & Hutter, F. "Hyperparameter Optimization in Machine Learning: A Comprehensive Survey," *Journal of Machine Learning Research*, 20(1), 1-45, 2019. Available at: <https://www.jmlr.org/papers/v20/18-444.html>.
- [4] Gamage, G., Samarakoon, S., & Nguyen, N. T. "Energy-Efficient Machine Learning Models for Healthcare Applications." *IEEE Access*, 9, 150357-150373, 2021. DOI: 10.1109/ACCESS.2021.3124182.
- [5] Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran, K., ... & Bengio, Y. "Sustainable AI: Environmental Implications, Challenges, and Opportunities." *In Proceedings of the 2022 Conference on Fairness, Accountability, and Transparency*, 145-156, 2022, DOI: 10.1145/3442188.3445934.
- [6] Zheng, B., Yoon, S. W., & Lam, S. S. "Breast cancer diagnosis based on feature extraction using a hybrid of K-means and support vector machine algorithms," *Expert Systems with Applications*, 41(4), 1476-1482, 2021. <https://doi.org/10.1016/j.eswa.2021.08.027>
- [7] Delen, D., Walker, G., & Kadam, A. "Predicting breast cancer survivability: A comparison of three data mining methods," *Artificial Intelligence in Medicine*, 34(2), 113-127, 2020. <https://doi.org/10.1016/j.artmed.2020.08.003>
- [8] Zizaan, Asma, and Ali Idri. "Evaluating and Comparing Bagging and Boosting of Hybrid Learning for Breast Cancer Screening." *Scientific African*, vol. 23, Mar. 2024, doi:10.1016/j.sciaf.2023.e01989.
- [9] Jegadeeswari, K., and R. Rathipriya. "Optimized Stacking Ensemble Classifier for Early Cancer Detection Using Biomarker Data." *Advance Sustainable Science Engineering and Technology* 6, no. 4 (2024): 02404017-02404017.
- [10] Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. "Optuna: A next-generation hyperparameter optimization framework," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623-2631, 2019. <https://doi.org/10.1145/3292500.3330701>
- [11] Bergstra, J., Yamins, D., & Cox, D. D. "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," *Proceedings of the 30th International Conference on Machine Learning*, 28, 115-123, 2013.
- [12] Henderson, P., Hu, J., Romoff, J., Brunskill, E., Jurafsky, D., & Pineau, J. "Towards the systematic reporting of the energy and carbon footprints of machine learning," *Journal of Machine Learning Research*, 21(1), 1-43, 2020.
- [13] Liu, Z., Cheng, S., Zhou, H., & You, Y. "Hanayo: Harnessing Wave-like Pipeline Parallelism for Enhanced Large Model Training Efficiency", 2023. <https://doi.org/10.1145/3581784.3607073>
- [14] Strubell, E., Ganesh, A., & McCallum, A. "Energy and policy considerations for deep learning in NLP," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3645-3650, 2019. <https://doi.org/10.18653/v1/P19-1355>
- [15] Lottick, K., Sakaguchi, K., Schwartz, R., & Smith, N. A. "Energy and policy considerations for deep learning in NLP," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 364-367, 2020. <https://doi.org/10.18653/v1/2020.acl-main.34>
- [16] Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. "Green AI," *Communications of the ACM*, 63(12), 54-63, 2020. <https://doi.org/10.1145/3381831>

- [17] Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L. M., Rothchild, D., Socher, R., & Dean, J. "Carbon Emissions and large neural network training". *arXiv preprint arXiv:2104.10350*, 2021.
- [18] Lo, F., Bitz, C. M., and Hess, J. J. "Development of a Random Forest Model for Forecasting Allergenic Pollen in North America". *Sci. Total Environ.* 773, 145590, 2021. doi: 10.1016/j.scitotenv.2021.145590
- [19] Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. "Optuna: A next-generation hyperparameter optimization framework." *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631, 2019.
- [20] Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. "Hyperband: Bandit-based configuration evaluation for hyperparameter optimization." *International Conference on Learning Representations*, 2017.
- [21] Saranya, G., & Pravin, A. "GridSearch based optimum feature selection by tuning hyperparameters for heart disease diagnosis in machine learning." *The Open Biomedical Engineering Journal*, 17(1), 2023a. <https://doi.org/10.2174/18741207-v17-e230510-2022-ht28-4371-8>
- [22] Zhu, N.; Zhu, C.; Zhou, L.; Zhu, Y.; Zhang, X. "Optimization of the Random Forest Hyperparameters for Power Industrial Control Systems Intrusion Detection Using an Improved GridSearch Algorithm." *Appl. Sci.* 2022, 12, 10456. <https://doi.org/10.3390/app122010456>
- [23] K. Jegadeeswari, R. Rathipriya, "Green AI Practices in Multi-objective Hyperparameter Optimization for Sustainable Machine Learning", *International Journal of Information Technology and Computer Science (IJITCS)*, Vol.17, No.2, pp.1-9, 2025. DOI:10.5815/ijitcs.2025.02.01.
- [24] Jegadeeswari, K., & Rathipriya, R. "Minimizing the carbon footprint of machine learning techniques through sustainable AI training methods." In *Sustainable information security in the age of AI and green computing*. IGI Global, 2025.
- [25] Dodge, J., Prewitt, T., Combes, R T D., Odmark, E., Schwartz, R., Strubell, E., Luccioni, A S., Smith, N A., DeCario, N., & Buchanan, W. "Measuring the Carbon Intensity of AI in Cloud Instances", 2022.
- [26] K. Jegadeeswari, R. Rathipriya and J. Renugadevi, "Fusion Learning of Regression Models for Missing Data Imputation in Breast Cancer Dataset," *2023 International Conference on Artificial Intelligence for Innovations in Healthcare Industries (ICAIIHI)*, Raipur, India, 2023, pp. 1-14, doi: 10.1109/ICAIIHI57871.2023.10489656.
- [27] K Jegadeeswari, R Ragunath, R Rathipriya, "A Prediction Model with Multi-Pattern Missing Data Imputation for Medical Dataset", *Advanced Network Technologies and Intelligent Computing, ANTIC 2022*, CCIS 1798, Singapore Nature, 2023, 798, 2023, https://doi.org/10.1007/978-3-031-28183-9_38.

Article Information Form

Authors Contributions

KJ and RR: conceptualization, implementation, and design. KJ and RR: writing and editing. RR: verification. All authors contributed to the article and approved the submitted version.

Acknowledgments

The authors has no received any financial support for the research, authorship or publication of this study.

Conflict of Interest Notice

The authors declare no conflict of interest.

Ethical Approval

This study does not require ethics committee permission or any special permission.

Availability of data and material

The data available in the UCI Repository

Artificial Intelligence Statement

No artificial intelligence tools were used while writing this article.

Plagiarism Statement

This article has been scanned by iThenticate™.