Analysis of Machine Learning Models Based on Predictive Performance, Energy Consumption, and Carbon Emissions

Willy Permana Putra[1,2,*], Eko Marpanaji[3], Septafiansyah Dwi Putra[4]

[1,3]*Doctoral Program in Engineering Science, Faculty of Engineering, Universitas Negeri Yogyakarta, Yogyakarta, 55281, Indonesia*

[2]*Informatics Engineering, Politeknik Negeri Indramayu, Indramayu, 45252, Indonesia*

[4]*Internet Engineering Technology, Politeknik Negeri Lampung, Lampung, 35141, Indonesia*

(Received: February 25, 2026; Revised: May 1, 2026; Accepted: July 22, 2026; Available online: August 9, 2026)

Abstract

This study aims to evaluate intrusion detection models by jointly considering predictive performance and computational sustainability. The main problem addressed is that many intrusion detection studies emphasize classification accuracy while providing limited evidence about execution time, energy use, and carbon dioxide-equivalent emissions, even though these factors affect repeated training and practical deployment. The contribution of this work is a comparative assessment of four supervised learning models, Random Forest, Histogram-based Gradient Boosting, Support Vector Machine, and Extreme Gradient Boosting, under a unified experimental workflow. The methodology uses the Wednesday working-hours subset of the Canadian Institute for Cybersecurity intrusion detection dataset released in 2017, which contains benign traffic and several denial-of-service attack classes. The procedure includes dataset selection, data cleaning, preprocessing, stratified training and testing, model fitting, predictive evaluation, sustainability measurement, and comparative interpretation. The evaluation is supported by one workflow figure, tables describing predictive results and sustainability measurements, and comparative visualizations of classification and resource-efficiency outcomes. The results show that Extreme Gradient Boosting achieved the strongest overall classification performance, with an accuracy of 0.9994, macro recall of 0.9966, macro F1-score of 0.9956, macro precision of 0.9946, and one-versus-rest area under the curve of 0.9999, while requiring 0.002559 kilowatt-hours of energy and 11.83 seconds of execution time. Random Forest produced highly comparable predictive results with similarly low resource consumption. Histogram-based Gradient Boosting was the most efficient model in terms of time, energy use, and emissions, but its macro-level performance was substantially lower. Support Vector Machine produced acceptable predictive results but required substantially higher computational resources. These findings imply that sustainable intrusion detection should select models through a balanced evaluation of detection capability and computational cost rather than accuracy alone.

Keywords: Machine Learning, Comparative Evaluation, Energy Efficiency, Carbon Emissions, Computing Time

1. Introduction

Digital transformation has increased organizational dependence on continuously available network services and web-based applications. As this dependence grows, the cyberattack surface also expands, particularly for systems that must remain reliable, responsive, and accessible. Intrusion detection systems based on machine learning have become important because they can learn complex traffic patterns that are difficult to represent using static rules. However, model deployment in operational environments also creates computational cost through repeated training, tuning, and evaluation. Therefore, intrusion detection research should not only examine predictive capability but also consider energy consumption, execution time, and carbon emissions in line with the Green AI perspective [1], [2].

The problem addressed in this study is the lack of balanced evidence for selecting intrusion detection models when both detection performance and sustainability are important. A model with excellent accuracy may be unsuitable if it requires excessive computational time or energy, whereas a highly efficient model may be risky if it performs poorly on minority attack classes. This issue is particularly relevant in multiclass intrusion detection because overall accuracy can mask weak class-wise detection when the dataset is imbalanced. Consequently, macro-level metrics and

***Corresponding author: Willy Permana Putra (willy_p@polindra.ac.id)**
DOI: https://doi.org/10.47738/jads.v7i3.1482

sustainability indicators need to be assessed together to support responsible model selection. The CICIDS2017 dataset is widely used in intrusion detection research because it provides labeled benign and attack traffic generated from realistic network scenarios[3]. This study focuses on the Wednesday-workingHours subset because it provides a focused denial-of-service classification setting consisting of BENIGN, DoS Hulk, DoS GoldenEye, DoS Slowloris, and DoS Slowhttptest traffic. The subset was selected to isolate the evaluation to denial-of-service behavior rather than combining heterogeneous attack families from different days of CICIDS2017. This design improves internal consistency for the present comparison, although it also limits the generalizability of the findings to other attack categories, such as brute-force, botnet, infiltration, and web attacks. This limitation is explicitly acknowledged in the discussion and future work sections.

Previous intrusion detection studies have demonstrated the effectiveness of tree-based ensembles, boosting methods, and margin-based classifiers. Random Forest has been widely adopted because ensemble decision trees can handle high-dimensional tabular features and reduce variance [4]. Extreme Gradient Boosting has also achieved strong performance in intrusion and denial-of-service detection, owing to its sequential error correction and regularization [5]. Support Vector Machine remains a relevant baseline because it can produce discriminative decision boundaries, although its computational behavior can become expensive on large datasets [6]. Histogram-based Gradient Boosting is attractive because feature binning can accelerate training on tabular data [7]. Nevertheless, these findings are often reported separately, and many studies still emphasize predictive results without systematically reporting energy and emission measurements.

Based on this background, the study is guided by the following research questions: RQ1: How do Random Forest, Histogram-based Gradient Boosting, Support Vector Machine, and Extreme Gradient Boosting differ in predictive performance for multiclass denial-of-service intrusion detection? RQ2: How do these models differ in execution time, energy consumption, and carbon emissions under the same experimental environment? RQ3: Which model provides the most balanced trade-off between predictive performance and computational sustainability? The main contribution of this study is a reproducible comparative evaluation that combines macro-level classification metrics, sustainability measurements, and a simple multi-criteria scoring procedure to support Green AI-oriented model selection for intrusion detection.

2. Literature Review

2.1. Machine Learning-Based Intrusion Detection

Machine learning-based intrusion detection has been widely investigated because network traffic data often contain nonlinear and high-dimensional patterns. Prior studies using CICIDS2017 and related datasets generally show that ensemble and boosting algorithms achieve strong performance for attack classification [3]. However, these studies differ in feature processing, class composition, validation strategy, and reported metrics, making direct comparison difficult. Some works emphasize high accuracy, while others highlight the importance of recall and F1-score for minority attack classes. This indicates that the research gap is not merely the absence of machine learning methods, but the lack of consistent and balanced evaluation criteria for practical intrusion detection deployment.

The reviewed literature suggests three main methodological tendencies. First, Random Forest and Extreme Gradient Boosting are frequently effective on tabular intrusion datasets because tree ensembles can model nonlinear feature interactions [4], [5], [8], [9], [10]. Second, Support Vector Machine remains useful as a comparative classifier, but its runtime can increase substantially when the number of samples is large or when nonlinear kernels are used [6]. Third, histogram-based boosting methods are designed for computational efficiency, yet their class-wise behavior in imbalanced multiclass intrusion detection requires careful examination [7], [11], [12]. These differences motivate a comparison that evaluates not only global accuracy, but also macro precision, macro recall, macro F1-score, discriminative ability, execution time, energy usage, and emissions [13].

2.2. Computational Sustainability and Green AI

Although predictive metrics such as accuracy, precision, recall, F1-score, and area under the curve remain central in intrusion detection evaluation, they do not fully describe operational feasibility. The Green AI perspective argues that model development should account for computational cost, energy consumption, and environmental impact [1]. This

issue is important for intrusion detection because detection models may be retrained periodically, executed repeatedly, or deployed in resource-constrained environments. Consequently, a model with very high predictive performance is not automatically the best deployment option if it requires excessive computation.

Recent sustainable artificial intelligence studies emphasize transparent reporting of computing resources and environmental impact during machine learning experimentation [13]. CodeCarbon supports this direction by estimating energy consumption and carbon dioxide-equivalent emissions during model execution. In the intrusion detection context, such measurements complement conventional classification metrics and help identify whether a model is suitable for repeated or operational deployment. Therefore, sustainability-oriented evaluation extends predictive evaluation rather than replacing it.

2.3. Research Gap

The literature indicates that Random Forest, Extreme Gradient Boosting, Support Vector Machine, and boosting-based methods can achieve strong results on intrusion datasets [4]-[7], [10], [14], [15], [16]. However, three gaps remain. First, many studies report accuracy without sufficient attention to macro-level behavior in the presence of class imbalance. Second, computational sustainability indicators such as execution time, energy consumption, and carbon emissions are still underreported. Third, model selection is often discussed qualitatively without an explicit trade-off procedure. These limitations motivate the present study, which evaluates four machine learning models using both predictive and sustainability metrics and introduces a simple normalized scoring procedure to support balanced interpretation.

Although energy- and emission-aware reporting has been advocated within the Green AI literature, it is still rarely applied to intrusion detection in practice. Most comparative studies on CICIDS2017 and related datasets report predictive metrics in isolation, and the few that mention efficiency typically limit their discussion to training or inference times without measuring energy consumption or carbon dioxide-equivalent emissions with dedicated tracking tools. Consequently, there is limited empirical evidence on how established intrusion detection models compare when predictive quality and measured computational sustainability are jointly considered within a single controlled workflow. The present study addresses this gap by pairing macro-level classification metrics with tool-based energy and emission measurements for Random Forest, Histogram-based Gradient Boosting, Support Vector Machine, and Extreme Gradient Boosting, and by summarizing the outcome through a transparent normalized trade-off procedure.

Accordingly, this study extends prior work by positioning model comparison as a dual-objective evaluation problem. The analysis seeks to determine whether the most accurate model is also the most sustainable, whether the most energy-efficient model maintains acceptable macro-level performance, and whether a balanced alternative can be identified for Green AI-oriented intrusion detection.

3. Methodology

This study adopted a comparative experimental design to assess the performance of four supervised machine learning algorithms, namely Random Forest (RF), Histogram-based Gradient Boosting (HGB), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB), for multiclass classification. The experiment was designed to examine two main aspects, namely predictive performance and computational efficiency, in order to provide a more comprehensive evaluation of each model. The overall research workflow is presented in figure 1.

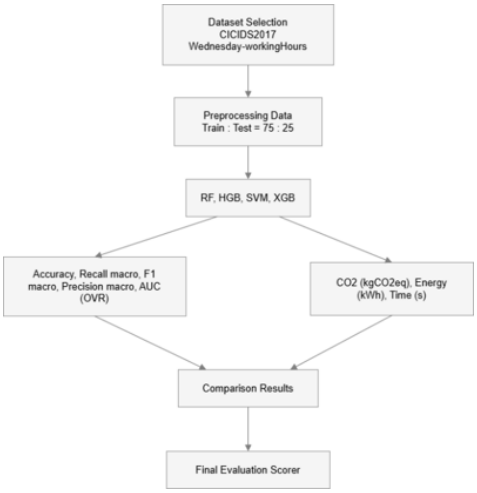

Figure 1. Research process flow.

3.1. Dataset and Experimental Environment

The dataset used in this study was Wednesday-workingHours.pcap_ISCX.csv from the CICIDS2017 dataset. This subset was selected because it provides a focused, multiclass denial-of-service scenario that includes BENIGN, DoS Hulk, DoS GoldenEye, DoS Slowloris, and DoS Slowhttptest traffic. Other CICIDS2017 days were not included because they represent different attack families and would introduce a broader heterogeneous classification problem beyond the present research objective. Therefore, the results should be interpreted as evidence for denial-of-service-oriented intrusion detection rather than as a complete evaluation across all CICIDS2017 attack categories.

The experimental procedure consisted of dataset loading, data cleaning, preprocessing, stratified data splitting, model training, predictive evaluation, sustainability measurement, and comparative analysis. Data cleaning included removal of non-informative identifier attributes, treatment of missing and infinite values, and removal of duplicate records when present. Categorical class labels were encoded into numerical labels. Numerical features were normalized for algorithms sensitive to feature scale, particularly Support Vector Machine. No feature selection method was applied in the reported experiment so that all models were evaluated using the same cleaned feature space. The dataset was divided using a stratified 75:25 train-test split to preserve class proportions between the training and test sets. It should be noted that the evaluated implementations differ in their parallelization and calibration behavior, which affects the interpretation of the runtime and energy results. Random Forest, Histogram-based Gradient Boosting, and Extreme Gradient Boosting were executed with multithreaded tree construction (n_jobs=-1 where applicable) and could therefore utilize the available logical processors, whereas the Support Vector Machine solver is effectively single-threaded and could not exploit the same degree of parallelism. The Support Vector Machine was additionally configured with probability=True, which enables internal cross-validated probability calibration and substantially increases its training cost. Consequently, the runtime and energy figures reported for the Support Vector Machine reflect both its algorithmic scaling on large sample sizes and these implementation-specific settings, and should be interpreted accordingly rather than as a purely algorithmic property. Table 1 summarizes the class distribution of the Wednesday-workingHours subset after preprocessing, comprising 610,481 samples across five classes.

Table 1. Class distribution after preprocessing

Class Label	Number of Samples	Percentages
BENIGN	416736	68.2623%
DoS Hulk	172846	28.3126%
DoS GoldenEye	10286	1.6849%
DoS slowloris	5385	0.8821%
DoS Slowhttptest	5228	0.8564%
Total	610481	100.0000%

Table 2 lists the main hyperparameters used for each model. To keep the comparison consistent and reproducible, the configurations largely follow the standard library defaults, and any parameters not listed were retained at their default values; no hyperparameter search or tuning was performed, so the results reflect the out-of-the-box behavior of each model under a common workflow. Random Forest was trained with 100 trees using the Gini criterion, unrestricted depth, and square-root feature sampling, with multithreaded execution enabled (n_jobs=-1). Histogram-based Gradient Boosting used the log-loss objective with a learning rate of 0.1, up to 100 boosting iterations, a maximum of 31 leaf nodes, a minimum of 20 samples per leaf, and 255 feature bins. Extreme Gradient Boosting was configured with 100 estimators, a maximum depth of 6, a learning rate of 0.3, full row and column sampling, the multi:softprob objective, and multithreaded execution (n_jobs=-1). The Support Vector Machine used a radial basis function kernel with C=1.0 and gamma set to scale, with probability estimation enabled (probability=True) and one-versus-rest decision shape. A common random seed (random_state=42) was fixed for all models to ensure reproducible partitioning and training, and no class weighting was applied (for example, class_weight=None for the Support Vector Machine), so all models were evaluated under identical class-balance conditions.

Table 2. Hyperparameter configuration of evaluated models

Model	Main Hyperparameters
Random Forest (RF)	n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, max_features='sqrt', bootstrap=True, random_state=42, n_jobs=-1
Histogram-based Gradient Boosting (HGB)	loss='log_loss', max_iter=100, learning_rate=0.1, max_leaf_nodes=31, max_depth=None, min_samples_leaf=20, l2_regularization=0.0, max_bins=255, random_state=42
Support Vector Machine (SVM)	kernel='rbf', C=1.0, gamma='scale', shrinking=True, probability=True, tol=0.001, cache_size=200, class_weight=None, decision_function_shape='ovr', random_state=42
Extreme Gradient Boosting (XGB)	n_estimators=100, max_depth=6, learning_rate=0.3, subsample=1.0, colsample_bytree=1.0, objective='multi:softprob', eval_metric='mlogloss', random_state=42, n_jobs=-1

3.2. Machine Learning Models

3.2.1. Random Forest (RF)

Random Forest is an ensemble learning algorithm that trains multiple decision trees and combines their class predictions using majority voting. It is well-suited to high-dimensional tabular intrusion data, as it can capture nonlinear feature interactions and reduce variance across individual trees [17]. The formulation of Random Forest is given in (1).

$$\hat{y} = \text{mode}\{h_b(x)\} \quad \text{for } b = 1,2,\dots,B \tag{1}$$

In (1), $\hat{y}$ denotes the final predicted class, $h_b(x)$ is the prediction of the b-th tree for input x, and B is the number of trees. The mode operator returns the class most frequently predicted by the ensemble.

3.2.2. HistGradientBoosting (HGB)

Histogram-based Gradient Boosting is a gradient boosting approach that discretizes continuous features into bins before split search, thereby reducing computational complexity and improving training efficiency on tabular datasets [18]. The formulation is given in (2).

$$F_m(x) = F_{m-1}(x) + \nu \cdot \gamma_m \cdot h_m(x) \tag{2}$$

In (2), $F_m(x)$ is the additive model at boosting iteration m, $F_{m-1}(x)$ is the model from the previous iteration, ν is the learning rate, γ_m is the step size, and $h_m(x)$ is the weak learner added at iteration m.

3.2.3. Support Vector Machine (SVM)

Support Vector Machine is a supervised learning algorithm that searches for a separating hyperplane with maximum margin between classes. Through kernel functions, it can model nonlinear class boundaries, but its computational cost may increase substantially on large-scale datasets depending on the kernel and parameter settings [19], [20], [21]. The formulation is given in (3).

$$f(x) = \text{sign}(w^T \phi(x) + b) \tag{3}$$

In (3), w denotes the weight vector, $\phi(x)$ is the transformed feature representation of input x, b is the bias term, and the sign function assigns the class according to the side of the decision boundary on which the sample lies.

3.2.4. Extreme Gradient Boosting (XGB)

Extreme Gradient Boosting is a scalable gradient boosting framework that builds additive trees sequentially while optimizing a regularized objective function. Its regularization and efficient tree construction make it competitive for tabular intrusion detection tasks [22], [23], [24], [25]. The formulation is given in (4).

$$\text{Obj}^{(t)} = \sum_i l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \tag{4}$$

In (4), $\text{Obj}^{(t)}$ denotes the objective at iteration t, l is the loss function, y_i is the true label, $\hat{y}_i^{(t-1)}$ is the prediction from the previous iteration, f_t is the newly added tree, and $\Omega(f_t)$ is the regularization term used to control model complexity.

3.3. Evaluation Metrics

3.3.1. Predictive Evaluation Metrics

The predictive performance of the evaluated models was assessed using five principal metrics, namely Accuracy, macro Recall, macro F1-score, macro Precision, and AUC (OVR). Macro-averaged metrics were adopted to ensure that each class contributed equally to the overall evaluation, thereby minimizing bias toward the majority class. In addition, AUC (OVR) was used to measure the discriminative capability of the model for each class against the remaining classes in the multiclass classification setting. The formulations for Accuracy, macro Recall, macro F1-score, macro Precision, and AUC (OVR) are given in (5).

$$\begin{aligned} Accuracy &= \frac{TP + TN}{TP + TN + FP + FN} \\ Precision &= \frac{TP}{TP + FP} \\ Recall &= \frac{TP}{TP + FN} \\ F1 &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \end{aligned} \tag{5}$$

In these formulations, the terms true positive (TP), true negative (TN), false positive (FP), and false negative (FN) are defined in a one-versus-rest manner for each class in the multiclass setting. For a given class c, TP denotes the number of samples of class c that are correctly predicted as c, FP denotes the number of samples from the remaining classes incorrectly predicted as c, FN denotes the number of samples of class c predicted as any other class, and TN denotes the number of samples from the remaining classes correctly not assigned to c. Accuracy is computed globally as the proportion of correctly classified samples across all five classes, whereas macro precision, macro recall, and macro F1-score are obtained by computing each metric independently for BENIGN, DoS Hulk, DoS GoldenEye, DoS Slowloris, and DoS Slowhttptest and then taking the unweighted average, so that every class contributes equally regardless of its size.

3.3.2. Sustainability Evaluation Metrics

Beyond conventional classification metrics, this study also assessed computational sustainability through a carbon and energy tracking approach. The recorded sustainability indicators include CO_2 emissions ($kgCO_2eq$), energy consumption (kWh), execution time (s), and detailed energy usage (kWh) for CPU and RAM. Such measurements are essential because high predictive performance does not necessarily imply computational efficiency. Within the Green AI paradigm, models that consume less energy and produce lower carbon emissions may offer a more responsible alternative, particularly in repeated, large-scale, or continuously running operational environments. The formulations for carbon emission and energy-related measurements are given in (6) – (10). In Equation (7), the carbon intensity (CI)

ISSN 2723-6471

represents the amount of carbon dioxide-equivalent emissions associated with generating one kilowatt-hour of electricity in the region where the experiment was executed. In this study, the carbon intensity used by the tracking tool in the experimental environment was approximately 0.676 kgCO_2eq/kWh, and the same factor was applied uniformly to all evaluated models. Because emissions are obtained by multiplying the measured energy consumption by this factor, the reported carbon dioxide-equivalent values are directly proportional to energy use and should be interpreted as region-specific estimates rather than hardware- or grid-independent properties of the algorithms.

$$E_{total} = E_{CPU} + E_{RAM} \quad (6)$$

$$CO_{2eq} = E_{total} \times CI \quad (7)$$

$$T_{exec} = t_{end} - t_{start} \quad (8)$$

$$E_{CPU} = P_{CPU} \times T_{CPU} \quad (9)$$

$$E_{RAM} = P_{RAM} \times T_{RAM} \quad (10)$$

The notation in these equations can be described as follows: E_{total} represents the total system energy consumption, whereas E_{CPU} end E_{RAM} denote the energy consumption of the CPU and RAM, respectively. The term CO_{2eq} refers to the equivalent carbon emissions, while CI indicates the carbon intensity of the electricity source. The variable T_{exec} denotes the total execution time, with t_{start} and t_{end} representing the start and end times of the computational process. Moreover, P_{CPU} and P_{RAM} correspond to the average power of the CPU and RAM, while T_{CPU} and T_{RAM} indicate the duration of their respective usage.

4. Results and Discussion

4.1. Experimental Results

This section reports experimental results from Random Forest, Histogram-based Gradient Boosting, Support Vector Machine, and Extreme Gradient Boosting using two complementary dimensions: predictive performance and computational sustainability. Table 3 presents the predictive performance of the evaluated models, whereas table 4 summarizes the sustainability-related results. Because the experiment used a single stratified train-test split, very small differences between models are interpreted descriptively rather than as statistically significant differences.

Table 3. Comparison of Predictive Performance

Metric	Machine Learning			
	RF	HGB	SVM	XGB
Accuracy	0.9989	0.9976	0.9826	0.9994
Recall macro	0.9964	0.8203	0.9731	0.9966
F1 macro	0.9955	0.8186	0.9757	0.9956
Precision macro	0.9945	0.8169	0.9789	0.9946
AUC (OVR)	0.9992	0.8259	0.9991	0.9999

Table 3 shows that Extreme Gradient Boosting and Random Forest achieved the strongest predictive performance among the evaluated models. Extreme Gradient Boosting obtained the highest values for accuracy, macro recall, macro F1-score, macro precision, and one-versus-rest area under the curve. Random Forest followed very closely across nearly all metrics. However, because the numerical differences between these two models are very small, the results should be interpreted as descriptive evidence from the present experimental split rather than as proof of statistical superiority.

The most important contrast appears when accuracy is compared with macro-level metrics. Histogram-based Gradient Boosting obtained high accuracy, but its macro recall, macro F1-score, macro precision, and discriminative performance were substantially lower. This pattern suggests that the model may have classified majority classes well while performing less consistently on minority attack classes. Support Vector Machine achieved a high one-versus-rest

ISSN 2723-6471

area under the curve, indicating strong class separability, but its accuracy and macro metrics remained lower than those of Extreme Gradient Boosting and Random Forest. This result suggests that separability alone does not guarantee the strongest multiclass decision performance, especially when the final class assignment is affected by model calibration, class boundaries, and class distribution.

Table 4. Comparison of Carbon Emissions, Energy, Time

Metric	Machine Learning			
	RF	HGB	SVM	XGB
CO2 ($kgCO_2eq$)	0.001747	0.000711	0.931370	0.001730
Energy (kWh)	0.002584	0.001051	1.377907	0.002559
Time (s)	15.04	5.49	38295.39	11.83
Energy (kWh)	CPU 0.002431	CPU 0.000999	CPU 0.987179	CPU 0.002440
	RAM 0.000153	RAM 0.000053	RAM 0.390728	RAM 0.000120

Figure 2 visualizes the sustainability-related comparison using logarithmic scaling because the Support Vector Machine values are substantially larger than those of the other models. This scaling improves readability while preserving the relative differences in carbon emissions, energy consumption, execution time, and component-level energy usage.

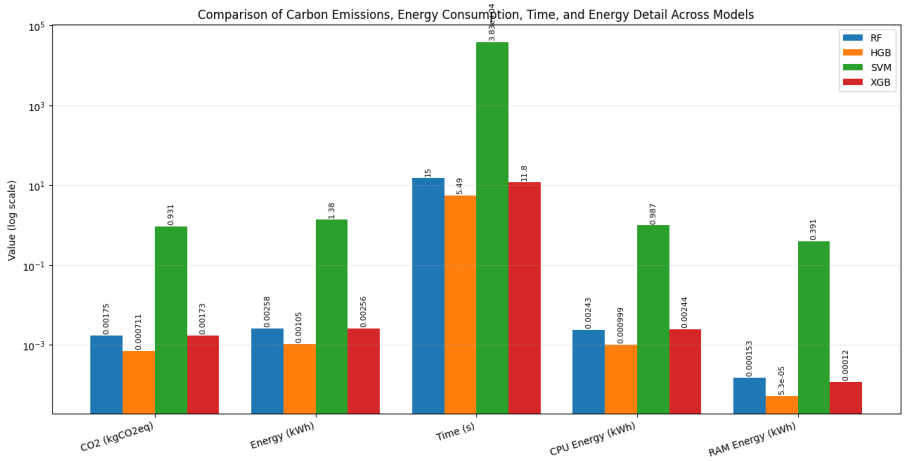

Figure 2. Comparison of Carbon Emission, Energy, and Time.

4.2. Predictive Performance Analysis

Consistent with table 3, Extreme Gradient Boosting and Random Forest formed the leading performance tier and were practically comparable across the evaluated metrics. The small observed advantage of Extreme Gradient Boosting is treated descriptively rather than as evidence of statistical superiority. Support Vector Machine retained strong discriminative ability, as indicated by its high one-versus-rest area under the curve, but lagged behind the two tree-based ensembles in accuracy and macro-level performance.

The principal anomaly concerns Histogram-based Gradient Boosting: its high overall accuracy contrasts sharply with substantially lower macro precision, macro recall, macro F1-score, and one-versus-rest area under the curve. This accuracy-macro discrepancy suggests uneven class-wise behavior, potentially involving minority-class misclassification under the default configuration. Because per-class results and a normalized confusion matrix are not yet available, this explanation remains tentative and is treated as a limitation rather than an intrinsic weakness of the algorithm.

4.3. Carbon Emissions, Energy, and Time Analysis

Table 4 reveals a clear sustainability pattern rather than merely small differences in isolated measurements. Histogram-based Gradient Boosting was the most resource-efficient model, whereas Extreme Gradient Boosting and Random

Forest formed a second, closely matched efficiency tier while also providing the strongest predictive results. Support Vector Machine was the clear outlier, with substantially greater execution time, energy consumption, and carbon emissions than the tree-based models. This contrast should not be interpreted solely as an inherent weakness of the algorithm, because the observed cost also reflects the radial basis function kernel, probability calibration, large sample size, and limited parallelization under the evaluated implementation. Overall, the results show that short multithreaded execution can be more sustainable than a lower-power computation that runs for an extended period.

The magnitude of the Support Vector Machine cost is better understood by considering the average power implied by table 4, obtained by dividing the reported energy by the corresponding execution time. Under this view, the tree-based models draw comparatively high average power while running only briefly—approximately 619 W for Random Forest over 15.04 s, 779 W for Extreme Gradient Boosting over 11.83 s, and 689 W for Histogram-based Gradient Boosting over 5.49 s—which is consistent with their multithreaded execution across many logical processors. The Support Vector Machine, in contrast, draws a much lower average power of approximately 130 W, but sustains it for 38,295 s. Its large energy and emission totals therefore arise primarily from a long, effectively single-threaded run combined with internal probability calibration, rather than from a high instantaneous power demand. This distinction is important because it shows that the observed difference reflects execution duration and implementation-specific settings as much as any intrinsic algorithmic inefficiency, and that the sustainability values should be interpreted with that qualification in mind.

The energy-emission values for Random Forest, Histogram-based Gradient Boosting, and Extreme Gradient Boosting are relatively small compared with the Support Vector Machine result in this experiment. However, their practical meaning should be discussed cautiously because carbon emissions depend on hardware utilization, regional carbon intensity, and CodeCarbon configuration.

4.4. Discussion from a Green AI Perspective

A key finding of this study is that predictive performance and computational sustainability do not always lead to the same model choice. Histogram-based Gradient Boosting was the most efficient model, but it showed weak macro-level performance. Extreme Gradient Boosting and Random Forest achieved the strongest predictive performance while maintaining low time, energy, and emission values. Support Vector Machine showed that acceptable predictive performance may still be impractical when computational cost is excessive. These findings support the Green AI argument that model assessment should include both performance and resource-efficiency dimensions [1], [13].

From a deployment standpoint, these findings are relevant for intrusion detection systems that require repeated execution, periodic retraining, or use in resource-constrained environments. Under such conditions, execution time and energy efficiency should be treated as design criteria rather than secondary observations. However, the present comparison should not be generalized beyond the selected subset without further validation on additional attack categories and external datasets.

4.4.1. Multi-Criteria Trade-off Analysis

To provide a clearer basis for the conclusion that Extreme Gradient Boosting offers the most balanced trade-off, this study applies a simple min-max normalized scoring procedure (table 5). Accuracy, macro recall, macro F1-score, macro precision, and one-versus-rest area under the curve are treated as benefit criteria, whereas carbon emissions, energy consumption, and execution time are treated as cost criteria. Each normalized criterion is assigned equal weight, and the final score is computed as the arithmetic mean of all normalized indicators. This scoring procedure is not intended to replace statistical testing, but it provides a transparent quantitative basis for ranking models under the selected criteria.

Table 5. Normalized multi-criteria trade-off score

Metric	Normalized score	Interpretation
Extreme Gradient Boosting (XGB)	0.9997	Highest observed balance between predictive and sustainability criteria

Random Forest (RF)	0.9952	Very close alternative with strong predictive performance and low resource usage
Histogram-based Gradient Boosting (HGB)	0.4866	Highly efficient, but penalized by low macro-level predictive performance
Support Vector Machine (SVM)	0.4577	Predictively acceptable, but heavily penalized by execution time and energy use

The score indicates that Extreme Gradient Boosting and Random Forest are effectively co-leading, with values of 0.9997 and 0.9952 respectively; this difference falls within the range that this study already characterizes as not statistically significant, so neither model can be claimed as decisively superior on the present single-split evidence.

4.4.2. Limitations

This study has several limitations. First, the experiment used only the Wednesday-workingHours subset of CICIDS2017, so the findings are most relevant to denial-of-service attack classification and should not be generalized to all intrusion categories. Second, the evaluation used a single stratified 75:25 train-test split; therefore, repeated runs, k-fold cross-validation, confidence intervals, and statistical hypothesis tests are needed to assess the stability and significance of the results, and the small differences between the two leading models should be read as descriptive rather than conclusive. Third, the current analysis does not include external validation on other intrusion detection datasets. Fourth, the low macro-level performance of Histogram-based Gradient Boosting is reported as an observation under the default configuration used here and has not yet been verified through a normalized confusion matrix and per-class precision, recall, and F1-score; in particular, its one-versus-rest area under the curve of 0.8259 is atypically low for a gradient boosting classifier at this accuracy level and is therefore treated as an anomaly that requires confirmation before it can be attributed to any intrinsic property of the algorithm. Fifth, the Histogram-based Gradient Boosting model was trained with the library's default early-stopping behavior, which is automatically enabled for large sample sizes; because this behavior was not explicitly controlled and relies on an internal validation split, it may have terminated training before the minority attack classes were fully learned, and this uncontrolled factor is a plausible contributor to the observed macro-level gap. Sixth, no class-imbalance handling such as class weighting or resampling was applied to any model, so the reported behavior on minority classes reflects the unadjusted class distribution and may differ if imbalance-aware training is used. Seventh, the runtime and energy comparison is not fully commensurable across models, because the Support Vector Machine was effectively single-threaded and was configured with internal probability calibration (probability=True), whereas the tree-based models used multithreaded execution; consequently, its energy and emission totals partly reflect execution duration and implementation settings rather than algorithmic efficiency alone, and all sustainability values remain dependent on hardware, software versions, regional carbon-intensity assumptions, and parallel-processing behavior. Future work should address these limitations through broader datasets, repeated validation, statistical testing, per-class confusion matrix analysis, controlled early stopping and handling of class imbalance, and energy-aware hyperparameter optimization.

5. Conclusion

This study evaluated Random Forest, Histogram-based Gradient Boosting, Support Vector Machine, and Extreme Gradient Boosting for multiclass denial-of-service intrusion classification using the Wednesday-workingHours subset of the CICIDS2017 dataset. The evaluation considered both predictive performance and computational sustainability, including accuracy, macro-level metrics, one-versus-rest area under the curve, execution time, energy consumption, and carbon dioxide-equivalent emissions.

The results show that Extreme Gradient Boosting achieved the strongest observed predictive performance, while Random Forest produced nearly equivalent results with similarly low resource usage. Histogram-based Gradient Boosting was the most efficient model in terms of execution time, energy consumption, and emissions, but its low macro-level metrics indicate weaker class-balanced performance. The Support Vector Machine achieved acceptable predictive performance but required substantially more computation time and energy, making it less suitable for Green AI-oriented deployment under the present configuration.

Using an equal-weight normalized trade-off score, Extreme Gradient Boosting ranked first, followed closely by Random Forest. This supports the conclusion that Extreme Gradient Boosting provided the best observed balance between detection effectiveness and computational sustainability in this experimental setting. Nevertheless, the difference between Extreme Gradient Boosting and Random Forest should be interpreted cautiously because the study did not include repeated runs, confidence intervals, or statistical significance testing.

Future work should extend the evaluation to additional CICIDS2017 subsets and other intrusion detection datasets, incorporate repeated validation and statistical testing, report per-class confusion matrices, and develop energy-aware hyperparameter optimization. Further research should also evaluate real-time deployment scenarios and multi-objective decision frameworks that jointly consider accuracy, latency, energy consumption, interpretability, and carbon emissions.

6. Declarations

6.1. Author Contributions

Conceptualization: W.P.P., E.M., and S.D.P.; Methodology: W.P.P., E.M., and S.D.P.; Software: W.P.P., E.M., and S.D.P.; Validation: W.P.P., E.M., and S.D.P.; Formal Analysis: W.P.P., E.M., and S.D.P.; Investigation: W.P.P., E.M., and S.D.P.; Resources: W.P.P., E.M., and S.D.P.; Data Curation: W.P.P., E.M., and S.D.P.; Writing Original Draft Preparation: W.P.P., E.M., and S.D.P.; Writing Review and Editing: W.P.P., E.M., and S.D.P.; Visualization: W.P.P., E.M., and S.D.P.; All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6.3. Funding

The authors would like to thank Politeknik Negeri Indramayu and Universitas Negeri Yogyakarta for supporting this research through academic collaboration and laboratory facilities.

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

This research received no external funding. The authors acknowledge the institutional and laboratory support provided by Politeknik Negeri Indramayu and Universitas Negeri Yogyakarta.

References

[1] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green AI,” *arXiv (Cornell University) / CoRR*, vol. abs/1907.10597, 2019, doi: 10.48550/arxiv.1907.10597.

[2] C. Slimani, L. Morge-Rollet, L. Lemarchand, D. Espes, F. Le Roy, and J. Boukhobza, “A study on characterizing energy, latency and security for Intrusion Detection Systems on heterogeneous embedded platforms,” *Future Generation Computer Systems*, vol. 162, no. Jan., pp. 1-13, Jan. 2025, doi: 10.1016/j.future.2024.07.051.

[3] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” in *Proceedings of the 4th International Conference on Information Systems Security and Privacy, SCITEPRESS - Science and Technology Publications*, vol. 2018, no. 1, pp. 108–116, 2018. doi: 10.5220/0006639801080116.

[4] P. V. Chavan and N. V. Alone, “Optimizing Intrusion Detection with Random Forest: A High-Accuracy Approach Using CIC-IDS 2017,” *Int. J. Comput. Appl.*, vol. 187, no. 3, pp. 17–22, May 2025, doi: 10.5120/ijca2025924816.

[5] M. Nassef, “Boosting Intrusion Detection Against DDoS Attacks Using a Feature Engineering-Based Fine-Tuned XGBoost Model,” *Int. J. Semant. Web Inf. Syst.*, vol. 21, no. 1, pp. 1–39, Jul. 2025, doi: 10.4018/IJSWIS.383062.

[6] H. GÜNEY, "Preprocessing Impact Analysis for Machine Learning-Based Network Intrusion Detection," *Sakarya University Journal of Computer and Information Sciences*, vol. 6, no. 1, pp. 67–79, Apr. 2023, doi: 10.35377/saucis...1223054.

[7] Md. A. Talukder, M. Khalid, and N. Sultana, "A hybrid machine learning model for intrusion detection in wireless sensor networks leveraging data balancing and dimensionality reduction," *Sci. Rep*., vol. 15, no. 1, pp. 4617-4629, Feb. 2025, doi: 10.1038/s41598-025-87028-1.

[8] A. Dey, "Advanced Threat Recognition: Harnessing Machine Learning for Intrusion Detection," *Int. J. Res. Appl. Sci. Eng. Technol*., vol. 12, no. 8, pp. 1248–1259, Aug. 2024, doi: 10.22214/ijraset.2024.64096.

[9] H. A. Salman, A. Kalakech, and A. Steiti, "Random Forest Algorithm Overview," *Babylonian Journal of Machine Learning*, vol. 2024, no. Jun., pp. 69–79, Jun. 2024, doi: 10.58496/BJML/2024/007.

[10] W. Chimphlee and S. Chimphlee, "Hyperparameters optimization XGBoost for network intrusion detection using CSE-CIC-IDS 2018 dataset," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 13, no. 1, pp. 817-829, Mar. 2024, doi: 10.11591/ijai.v13.i1.pp817-826.

[11] mina ibrahim, H. AbdelRaouf, K. Amin, and N. Semary, "Keystroke dynamics based user authentication using Histogram Gradient Boosting," *IJCI. International Journal of Computers and Information,* vol. 2022, no. Sep, pp. 1-12, Sep. 2022, doi: 10.21608/ijci.2022.155605.1081.

[12] L. Iosipoi and A. Vakhrushev, "SketchBoost: Fast Gradient Boosted Decision Tree for Multioutput Problems," *ArXiv*, vol. 2023, no. Nov., pp. 1-15, Nov. 2023, doi: 10.48550/arXiv.2211.12858.

[13] R. Verdecchia, J. Sallou, and L. Cruz, "A systematic review of Green AI," *WIREs Data Mining and Knowledge Discovery*, vol. 13, no. 4, pp. 1-12, Jul. 2023, doi: 10.1002/widm.1507.

[14] N. S. Abd, N. W. Khalid, and B. H. Ali, "The importance of the clustering model to detect new types of intrusion in data traffic," *arXiv,* vol. 2024, no. Nov., pp. 1-12, Nov. 2024, doi: 10.48550/arXiv.2411.14550.

[15] N. Kumar and U. Kumar, "Diverse Analysis of Data Mining and Machine Learning Algorithms to Secure Computer Network," *Wireless Personal Communications*, vol. 2021, no. Mar., pp. 1-12, Mar. 17, 2021. doi: 10.21203/rs.3.rs-305354/v1.

[16] B. Almaslukh, "Deep Learning and Entity Embedding-Based Intrusion Detection Model for Wireless Sensor Networks," *Computers, Materials & Continua*, vol. 69, no. 1, pp. 1343–1360, 2021, doi: 10.32604/cmc.2021.017914.

[17] L. Breiman, "Random Forests," *Mach. Learn*., vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.

[18] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011, [Online]. Available: http://jmlr.org/papers/v12/pedregosa11a.html

[19] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn*., vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[20] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset," *IEEE Access*, vol. 9, no. 1, pp. 22351–22370, 2021, doi: 10.1109/ACCESS.2021.3056614.

[21] R. A. Disha and S. Waheed, "Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique," *Cybersecurity*, vol. 5, no. 1, pp. 1-12, Dec. 2022, doi: 10.1186/s42400-021-00103-8.

[22] T. Chen and C. Guestrin, "XGBoost," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA: ACM*, Aug. vol. 2016, no. 1, pp. 785–794, 2016. doi: 10.1145/2939672.2939785.

[23] G.Divya, M.Naga, T.JayaDharani, B.Pavan, and K.Praveen, "Deciphering Cardiac Destiny: Unveiling Future Risks Through Cutting-Edge Machine Learning Approac," *arXiv*, vol., 2024, no. Sep., pp. 1-12, Sep. 2024, doi: 10.48550/arXiv.2409.15287.

[24] S. A. Ajagbe, J. B. Awotunde, and H. Florez, "Intrusion Detection: A Comparison Study of Machine Learning Models Using Unbalanced Dataset," *SN Comput. Sci*., vol. 5, no. 8, pp. 1028-1045, Nov. 2024, doi: 10.1007/s42979-024-03369-0.

[25] Md. A. Talukder et al., "Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction," *J. Big Data*, vol. 11, no. 1, pp. 1-12, 2024, doi: 10.1186/s40537-024-00886-w.