Decision Trees and Machine Learning for Cybersecurity: How Model Settings Affect Attack Detection

Konrad Lukiewicz[1][0000−0003−2463−9338], Michal Podpora[2][0000−0002−1080−6767], Grzegorz Dralus[1][0000−0001−7963−8602], Damian Mazur[1][0000−0002−3247−5903], Jakub Dralus[3], Tomasz Kajdanowicz[4][0000−0002−8417−1012], and Aleksandra Kawala-Sterniuk[4][0000−0001−7826−1292]

[1] Department of Electrical and Computer Engineering Fundamentals, Rzeszow University of Technology, 35-959 Rzeszow, Poland
kondluki@wp.pl, {gregor,mazur}@prz.edu.pl
[2] Institute of Computer Science, University of Opole, 45-052 Opole, Poland
michal.podpora@uni.opole.pl
[3] Faculty of Electrical and Computer Engineering, Cracow University of Technology, Warszawska 24, 31-155 Krakow, Poland
jakub.dralus@student.pk.edu.pl
[4] Department of Artificial Intelligence, Faculty of Information and Communication Technology, Wroclaw University of Science and Technology, 50-370 Wroclaw, Poland
{aleksandra.kawala-sterniuk,tomasz.kajdanowicz}@pwr.edu.pl

Abstract. The aim of this research is to evaluate the effectiveness of decision trees in detecting cyberattacks and to compare different learning conditions' impact on classification performance. The paper presents an analysis of the impact of various hyperparameters, including splitting criteria (Gini vs. Entropy), feature selection, and tree depth, on the accuracy, precision, recall, and F-measure of models. A comparative analysis is performed using machine learning classifiers, such as Gradient Boosting, AdaBoost, Support Vector Machines (SVM), Lasso, and Random Forest, to assess their relative performance in cyber-attack classification. The findings demonstrate that decision trees are able to achieve high effectiveness in detecting network intrusions, and feature selection can enhance classification performance. Some of the classifiers among the evaluated models, including Random Forest and Gradient Boosting, offer better performance, showcasing their potential as alternatives to decision trees in cybersecurity applications. The results show the importance of hyperparameter optimization and feature engineering, particularly in improving threat detection model performance and/or accuracy.

Keywords: cyber-attacks · decision tree · classification · feature selection · machine learning.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

1 Introduction

With the rapid advancement of machine learning applications, optimizing algorithms has become a key element to improve the effectiveness of mathematical modeling, also in cybersecurity applications. This research paper examines the impact of learning conditions on the accuracy of a decision tree algorithm. Specifically, it explores how various parameters – such as tree depth, partitioning criteria, minimum samples per leaf, and the number of selected features – affect model performance. Various evaluation measures are considered to assess the efficiency of decision tree models in attack detection.

Understanding how different learning conditions influence the performance of the decision tree is essential to improve the accuracy of classification. By tracing their influence on the model-building process it is possible to find their impact on the quality of classification and prediction. In the context of accurately assessing the performance of the model, this article will closely examine various measures such as precision, precision, recall, and F-measure. In this context, different implementations of these measures in case of specific learning conditions are considered, identifying which ones may better reflect the effectiveness of the decision tree algorithm depending on the specifics of the data analyzed.

The paper is structured as follows. Section 2 provides a literature study and a review of solutions, together with motivation for the study. Section 3 describes the database and the methods applied to build the decision tree. Section 4 describes the research methodology. In Section 5, we present the summary.

2 Review of Solutions Based on Recent Literature

There are many possible benefits to applying Machine Learning to cybersecurity challenges. One of such challenges is the need for a quick synthesization of intelligence gathered during an ongoing attack. Machine Learning is able to provide quick analysis and inference upon large amounts of historical and current incoming data. Moreover, by applying ML to specific tasks, security teams can be released of routine and repetitive tasks. Modern approaches to using AI have shown great promise in reducing error rates and speeding up detection [31, 19]. The use of algorithms can be divided into two main areas: automated threat detection and response, and autonomous threat detection and response. In both categories, Machine Learning can automate manual work with particular attention to processes that require accuracy and real-time response.

Traditional cybersecurity systems, such as Security Information and Event Management (SIEM) and Intrusion Prevention Systems (IPS), are widely used to monitor and analyze network activity. SIEMs collect and process data from various IT resources to identify potential threats, but they typically lack AI-based automation [28]. Similarly, IPSs support network security without leveraging machine learning. While effective, these traditional solutions often fall short in terms of accuracy and adaptability compared to AI-driven approaches [19].

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

Table 1. Summary of various Machine Learning models, used over the decade.

Year	Source	Database	Topic	Learning model	Accuracy	Precision	Recall
2010	[9]	Spambase	Email spam	RF	95.43%	-	-
2011	[6]	Own base	Email spam	RF	95.00%	95.70%	95.70%
2012	[1]	Spambase	Email spam	DT	92.34%	93.50%	93.50%
2013	[29]	Own base	Emails	NB	85.96%	-	-
2014	[22]	SMS base	SMS spam	SVM, DT SVM, DT	SVM 98.61% DT 96.60%	SVM 98.60% DT 96.50%	SVM 98.60% DT 96.50%
2015	[5]	Twitter dataset	Spam tweets	SVM	95.20%	-	93.60%
2016	[30]	Twitter dataset	Spam tweets	ANN	91%	-	-
2018	[3]	Spambase	Email spam	ANN	92.41%	92.40%	92.40%
2019	[7]	Enron	Email spam	SVN	-	98.10%	98.10%
2020	[25]	Twitter dataset	Spam tweets	SVM	98.88%	-	94.47%

2.1 Motivation

Cybersecurity faces increasingly complex threats, demanding ongoing improvements in detection and response. Machine learning has become essential, enabling automated anomaly detection and threat classification. This paper evaluates the effectiveness of decision tree algorithms in identifying network security issues and compares their performance with other machine learning classifiers, with a focus on the impact of learning hyperparameters on accuracy.

The investigation of the deployment of machine learning for the identification of cybersecurity issues in computer networks and the analysis of the impact of mathematical learning conditions on the accuracy of the decision tree algorithm offers a number of advantages:

- **Improving the effectiveness of threat detection** – By understanding the impact of mathematical learning conditions on the accuracy of the decision tree algorithm, detection processes can be optimised, contributing to more effective protection of IT infrastructure against attacks.
- **Resource optimisation** – Finding the optimal conditions for mathematical learning allows for more efficient use of computing and human resources in the process of cybersecurity posture assessment, even in multistage attack scenarios [18].
- **Faster response to threats** – With more effective threat identification methods, organisations can respond quicker to any potential attacks and to take appropriate countermeasures, minimising the risk of major security threats and their consequences.
- **Enhancing IT infrastructure resilience** – By exploring the ML application in the cyber domain, organizations can strengthen their resilience to prepare for increasingly sophisticated cyber threats. Improving the accuracy of decision-making algorithms, such as decision trees, can help to address potential threats more effectively.

2.2 Representative Machine Learning Approaches

Beyes classifiers rely on the Bayesian statistical framework to identify, for instance, spam messages based on keywords, senders, or domains [31, 24]. **Neural Networks**, including Recurrent Neural Networks (RNN) and Convolutional

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

Neural Networks (CNN), demonstrate ability to detect more complex patterns [15]. **Decision-tree-based methods** (e.g., Random Forest, Gradient Boosting) typically split feature space into increasingly homogeneous regions, helping to classify various threats in a straightforward, rule-based manner [12]. **Dataset-driven methods** use publicly available labeled data, enabling supervised algorithms to learn typical threat signatures and patterns [14].

2.3 Intruder Detection in a Computer Network

Commonly used machine learning approaches for intrusion detection include artificial neural networks (ANN), fuzzy association, Support Vector Machines (SVM), and decision trees [23, 11, 8, 16]. Recent work has shown that these methods can significantly improve detection rates for threats such as Probe or Remote-to-Local (R2L) attacks, achieving high accuracy (e.g., up to 99.80% with Naive Bayes or decision trees) [33, 27, 17, 26, 21, 2, 4]. Early detection and identification of emerging attack vectors remain challenging, prompting researchers to try hybrid approaches that combine anomaly-based and misuse-based detection. As a result, these techniques are being researched, causing improvement of learning conditions and the availability and quality of large-scale threat datasets.

3 Database and Methods to Build the Decision Tree

The database collection has been subjected to a series of experiments and scientific studies over the past decade. The entire collection has more than 126 thousand records used for network learning. However, the collection is not perfect, as it has significant amounts of redundant records which affects the performance of the network and prevents the learning algorithms from working correctly, also redundant records in the training set were eliminated. This treatment makes the classifier unbiased and allows it to represent a more realistic environment. The lack of repetitive data also affects the performance of the classifier which results in better rates during scientific studies. The training and test set contains a reasonable number of instances, which is affordable for experiments on the entire set without having to randomly select parts of the set.

The classes and data labels presented in the database collection are divided into four categories that represent the corresponding attack class:

- **Denial of Service** (DoS) attacks – an attack designed to block or restrict a computer system, network resources or services.
- **Probe** – an attack involving the introduction of an intruder to scan and search for information or vulnerabilities in a network or computer system.
- **Remote to Local** (R2L) – an attack involving the introduction of an intruder who remotely gains unattended access to a computer system via a network by sending a data packet to the system.
- **User to Root** (U2R) – an intruder gains access to a user with permissions allowing standard use and then, already inside the system, attempts to gain access to a user with administrative rights.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

3.1 A methodology for constructing trees using the Gini index

The methodology for building a decision tree using a Gini index, known as a Gini tree, is a popular technique used in data analysis and machine learning. The first step in building is the root of the tree – based on the Gini index, a variable and its boundary value are selected to divide the training dataset. The training dataset is then divided into subsets – variables that meet a given condition will go into one subset and those that do not meet the condition will go into another [10]. A Gini index is calculated for each subset, which is a measure of how well the split separates the classes in the data. The lower the value of the Gini index, the greater the homogeneity of the subset. The step of splitting the dataset and calculating the index is repeated recursively until a stop condition is reached. This may include a certain depth of the tree, a minimum number of observations in the leaf or reaching maximum homogeneity [20].

Gini tree leaves are created when the stop condition is met. The leaves are labelled with a target value, which is assigned to the majority of observations in the leaf. Once the tree is built, a pruning process takes place. Its purpose is to reduce overfitting and improve the overall generalisability of the model. It involves removing nodes or subtrees that do not yield significant predictive improvement. The tree is built in a way that minimises the Gini index at each splitting step, resulting in a tree with minimal heterogeneity and well-matched training data. They are often used because of their simplicity, easy interpretability and effectiveness in data classification.

Gini Impurity is one of the measures used to build a decision tree to determine how the features of a data set should divide the nodes to form a tree. We denote Gini Impurity by a measure of how often a randomly selected element from a set is labelled according to the distribution of labels in the subset. Consider a set D containing k class samples. We denote the probability of the samples belonging to class i at a given node as p. The Gini Impurity for D is defined as:

$$Gini(D) = 1 - \sum_{i=1}^{k} *p_i^2 \tag{1}$$

A node with an even distribution of classes has the highest contamination. The minimum is obtained when all records belong to the same class. If the dataset D is divided into two subsets D_1 and D_2 of size n_1 and n_2 then Gini Impurity should be defined as:

$$Gini_A(D) = \frac{n_1}{n} Gini\,(D_1) + \frac{n_2}{n} Gini\,(D_2) \tag{2}$$

To obtain the information gain for the attribute, the weighted branch impurity is subtracted from the original impurity. The best split can also be selected using *Gini gain*, which is calculated using the following formula:

$$\Delta Gini(A) = Gini(D) - Gini_A(D) \tag{3}$$

There are various metrics used to train decision trees. In addition to the Gini Impurity described above, there is also the Entropy measure. This is an

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

information theory metric that measures impurity and distortion in a group of observations. It determines how the decision tree chooses to partition the data. Consider a dataset with N classes. Calculate the entropy using the formula

$$E = -\sum_{i=1}^{N} p_i \log_2 p_i \tag{4}$$

Where p_i defines the probability of randomly selecting an example in class i.

3.2 Entropy-based tree building method

An alternative to Gini is the entropy-based decision tree-building method, known as the Entropy tree. Like the Gini tree, it begins by selecting the root variable and its boundary value to partition the training dataset. The best partitioning is determined by information gain, which measures the reduction in entropy after splitting. Entropy quantifies uncertainty in the data, and information gain represents its reduction after partitioning. The higher the information gain, the more accurate and better the partitioning [13]. The algorithm recursively repeats the steps of partitioning the dataset, calculating entropy, and computing information gain until a stopping condition is met – either at a predefined tree depth, at a minimum number of samples per leaf, or if the subset achieves homogenity. The last subdivisions, before the stop condition is met, form the leaves of the tree, which are labelled with a target value assigned to the most observations in a leaf. As in Gini, pruning can be applied in the final step to reduce overfitting and improve generalization.

The entropy-based tree building method relies on several formulas from information theory. One of the key formulas used is C4.5 algorithm entropy formula, which calculates the uncertainty or randomness of a system. The entropy for a set S is calculated as the sum of the entropy for each class in the set, given their proportions.

$$H(X) = -\sum_{i=1}^{n} p_i * \log_2 (p_i) \tag{5}$$

where S is Entropy, n – number of classes, p_i – proportion of class occurrence in the set.

Information gain $IG\ (S,\ A)$ when splitting a dataset S on attribute A is calculated as:

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H\left(S_V\right) \tag{6}$$

where $H(S)$ is the entropy of the dataset S before the split, $|S|$ is the total numbers of instances in the dataset S, $Values(A)$ is the set of possible values of attribute A, $|S_v|$ is the number of instances in dataset S that have the value v for the attribute A, $H(S_v)$ is the entropy of the dataset S_v for a specific value of attribute A.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

In entropy-based tree building, the goal is to recursively minimize entropy at each node split by selecting the attribute that maximizes information gain. This process continues until a predefined stopping criterion is met, such as a maximum tree depth or subset homogenity, resulting in the final tree structure.

3.3 Comparison of tree building methods

When constructing a decision tree using the Gini index and using Entropy, there are differences in the algorithmic approach. Table 2 shows the comparison made in the construction of the decision tree:

Table 2. Comparison of Gini and Entropy tree construction.

Comparative aspect	**Gini**	**Entropy**
Measure of heterogeneity	The Gini tree uses the Gini index as a measure of heterogeneity. This index measures how well the division separates different classes of data.	Entropy tree uses entropy as a measure of heterogeneity. The index measures disorder and uncertainty in the data.
Data breakdown	Division of data based on the value of the selected variable and the cut-off value, where the selection of the division variable varies.	Partitioning of data on the basis of a variable and a cut-off value that maximises information gain (reducing entropy).
Effectiveness of division	The Gini tree minimises the Gini index to reduce heterogeneity in the data.	Entropy tree maximises information gain.
Interpretability	Both are intuitive methods, easily interpretable, can be presented in a hierarchical form.	

The choice of methodology for constructing a decision tree depends on the specifics of the problem and the user's preferences. In practice, both methods produce similar expected results, which may differ in some cases. In the process of selecting a methodology, other factors such as efficiency, computation time and interpretability should also be taken into account when choosing an appropriate tree construction method.

4 Research Methodology

To evaluate the effectiveness of decision trees in cyber-attack detection, experiments were conducted using machine learning techniques. Since scikit-learn operates on structured numerical data, the dataset was reformatted into a two-dimensional matrix, where rows represented samples and columns corresponded to feature variables. To ensure reliable models' performance it was important to clean the datasets of any duplicate elements (unless the dataset used for testing had already been cleaned). If the dataset contained numeric and non-numeric

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

values, additional preprocessing was required to standardize the data format. A one-of-K and one-hot encoding method was therefore used. This technique consists of converting each categorical feature with m possible inputs into n binary features of which only one is active at a time.

The datasets used in the experiment did not have explicit duplicates but required format standardization. To improve model efficiency, feature selection was applied to remove redundant or irrelevant features that could lead to overfitting. Unnecessary features may decrease model accuracy, while increasing computation time without significant benefits.

The experiments used the *SelectPercentile* method from the *sklearn.feature_selection* module to select features based on their highest scores. Then, once the best subset of features was determined, recursive feature elimination was applied, which affects model building.

For the experiment, a decision tree model was built, which partitions the data using information gain. Each node was split based on a single feature, and if all resulting instances belonged to the same class, then such node was considered complete and clean. The recursive feature splitting continued until the entropy reached zero or a stopping criterion was met. The tree consisted of internal decision nodes and terminal leaves, with test function implemented in each node, determining the branches of final tree. The learning algorithm starts from the root and continues until a leaf node is reached. Each leaf node has a result label, which is a class objective in the case of classification and a numerical value when using regression. The model was evaluated using accuracy, precision, recall, F-measure, and confusion matrices.

4.1 Experiments

A decision tree learning algorithm was used for this experiment, acknowledging its general tendency to overfit. Therefore, exhaustive parameter tuning was performed to assess optimal model parameters. A tree was constructed, but using only one feature at a time to split the node and partition the data. This led to a single-variant use of features. Recursive feature elimination was used, with a number passed as a parameter to identify features. Two measures were implemented to determine (form) the decision trees – Gini Impurity and Entropy.

Feature selection was used for elimination of redundant and irrelevant attributes, selecting a minimal subset representing the problem with minimal performance degradation. Working with fewer features may improve the result, while attributes of similar value usually offer little predictive value.

The classifier is trained in the initial phase using all attributes and the weights assigned to each attribute. In the second stage, it was decided to reduce the smallest attributes to a smaller set. An analysis of the accuracy of the estimator was then carried out once the appropriate set of features had been selected. The analysis was performed for the two measures of decision tree construction. Table 3 provides details of the estimator accuracy analysis by models used for all attributes.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

Table 3. Accuracy analysis details for all attributes.

Model	Class	Accuracy	Precision	Recall	F-meas.
Gini	DoS	99.64%	99.51%	99.67%	99.59%
	Probe	99.51%	99.39%	99.27%	99.32%
	R2L	97.92%	97.15%	96.60%	97.05%
	U2R	99.65%	86.29%	90.96%	88.21%
Entropy	DoS	98.76%	98.50%	99.65%	99.61%
	Probe	99.47%	99.19%	98.95%	98.17%
	R2L	97.88%	97.03%	96.93%	97.98%
	U2R	99.62%	86.44%	87.87%	86.58%

The number of features was then reduced to the optimal quantities and the accuracy of the estimators was again analysed by the decision tree models used. The results are presented in Table 4.

Table 4. Details of accuracy analysis after feature reduction.

Model	Class	Accuracy	Precision	Recall	F-meas.
Gini	DoS	99.74%	99.69%	99.68%	99.70%
	Probe	99.08%	98.97%	99.30%	99.38%
	R2L	97.45%	97.70%	97.11%	98.02%
	U2R	99.65%	88.54%	91.04%	90.82%
Entropy	DoS	99.74%	99.71%	99.71%	99.71%
	Probe	99.07%	98.67%	99.09%	98.43%
	R2L	97.34%	97.40%	97.07%	98.22%
	U2R	99.67%	88.00%	90.26%	88.59%

The first confusion matrix presented in Table 5 juxtaposes Normal and Denial-of-Service (DoS) attacks, delineating the model's predictive efficacy. The matrix reveals a commendable discriminatory prowess, as evidenced by the high True Positive count for Normal instances and the marginal False Negative count for DoS instances. These findings are underscored by the notably high Positive Predictive Value (PPV) of 99.70%, indicative of the model's ability to accurately classify Normal instances. Furthermore, the Negative Predictive Value (NPV) of 99.68% underscores the model's proficiency in correctly identifying DoS instances, thereby highlighting its robustness in mitigating false classifications.

In the context of distinguishing between Normal and Probe attacks, the confusion matrix presented in table 6 illuminates the model's discriminative acumen. A meticulous examination reveals a substantial number of True Positives for both Normal and Probe instances, indicative of the model's adeptness in accurately categorizing these attack types. The high Positive Predictive Value of 99.38% corroborates the model's precision in identifying Normal instances, while the NPV of 99.30% accentuates its efficacy in discerning Probe attacks, thus bolstering confidence in its predictive capabilities.

Transitioning to the comparison of Normal and Remote-to-Local (R2L) attacks, the confusion matrix presented in Table 7 unveils nuanced patterns in the model's classification performance. Despite a substantive True Positive count for Normal instances, a discernible number of False Negatives for R2L attacks

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

Table 5. Confusion matrix for DoS attacks.

Confusion Matrix		Predicted Label: Normal	Predicted Label: DoS	
True values	Normal	9682	29	Positive predictive 99.70%
	DoS	23	7437	Negative predictive 99.68%

Table 6. Confusion matrix for Probe attacks.

Confusion Matrix		Predicted Label: Normal	Predicted Label: Probe	
True values	Normal	9651	60	Positive predictive 99.39%
	Probe	17	2404	Negative predictive 99.30%

underscore potential areas for model refinement. The Positive Predictive Value of 98,02% underscores the model's reliability in identifying Normal instances, albeit the NPV of 97.11% hints at a moderate propensity for misclassifying R2L attacks as Normal. Thus, while the model demonstrates commendable precision in categorizing Normal instances, further optimization may be warranted to enhance its discrimination of R2L attacks.

Table 7. Confusion matrix for R2L attacks.

Confusion Matrix		Predicted Label: Normal	Predicted Label: R2L	
True values	Normal	9519	192	Positive predictive 98.02%
	R2L	92	3099	Negative predictive 97.11%

Lastly, the comparison of Normal and User-to-Root (U2R) attacks unveils nuanced intricacies in the model's classification dynamics. Despite a discernible True Positive count for Normal instances, the presence of False Negatives for U2R attacks suggests potential areas for model refinement. The Positive Predictive Value of 90.82% underscores the model's efficacy in identifying Normal instances, albeit the NPV of 91.04% hints at a moderate propensity for misclassifying U2R attacks as Normal (see Table 8). Hence, while the model exhibits commendable precision in discerning Normal instances, opportunities for optimization may exist to augment its discrimination of U2R attacks.

Table 8. Confusion matrix for U2R attacks.

Confusion Matrix		Predicted Label: Normal	Predicted Label: U2R	
True values	Normal	8820	891	Positive predictive 90.82%
	U2R	6	61	Negative predictive 91.04%

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

4.2 Comparison of the Decision Tree algorithm with other classifiers

The Decision Tree-based classification model was compared with other selected machine learning model classifiers. Parameter tuning of all proposed ML models was performed using the search grid optimization method [32].

As part of this process, different combinations of hyperparameters were systematically tested to find the configuration that maximizes the accuracy of the predictions. The detailed values of the selected hyperparameters for each model are shown in Tables 9-13. The optimal parameters of the classification models were determined using 5-fold cross-validation. This technique gives high confidence in the determination of optimal parameter values and thus high accuracy with the classification task. The search grid method used set learning data.

The parameterisation in the Gradient Boosting method involved selecting the values of three parameters: learning rate, maximum depth and number of estimators. The grids of the parameters were searched and the most optimal parameters selected are shown in Table 9.

A search grid optimization method was used to search for the optimal values of the hyperparameters of the AdaBoost classifier model, in which only two parameters were optimized: learning rate, maximum depth and number of estimators. The searched hyperparameter values and the finally selected parameters are shown in Table 10.

Table 9. Hyperparameters tuning for the Gradient Boosting model.

Parameters	Set of values	Optimal value
Learning rate	0.01; 0.1; 0.2	0.1
Maximum depth	3; 5; 7	5
Number of estimators	50; 100; 200	200

Table 10. Hyperparameters tuning for the AdaBoost model.

Parameters	Set of values	Optimal value
Learning rate	0.01; 0.1; 0.2	0.01
Number of estimators	50; 100; 200	50

As with the other models, the best parameters for the SVM method were selected based on grid search tests as shown in Table 11. During the tests, a linear kernel was used, adjusting the parameters cost and number of iterations.

For the Random Forest method, the search values and selected search parameters are shown in Table 12. Optimal values of three parameters were sought: *number of estimators*, *maximum tree depth*, *minimum number of samples to split*.

The Lasso classifier method, as well as the Ridge Classifier, searched for optimal values of the alpha parameter only. The range of search values and the optimal values are shown in Table 13.

For the best additional classification models (with optimal parameters), the average accuracy was calculated for the test data. Table 14 shows the classification accuracy values of the best additional models for the four classes of attacks

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

Table 11. Hyperparameters tuning for the SVMt model.

Parameters	Set of values	Optimal value
Cost	0.01; 0.1; 1; 10	0.01
Number of iterations	1000; 5000; 10000	5000

Table 12. Hyperparameters tuning for the Random Forest model.

Parameters	Set of values	Optimal value
Number of estimators	50; 100; 200	200
Maximum tree depth	None; 10; 20; 30	None
Minimum number of samples to split	2; 5; 10	2

and with the logs that were not attacks, also the average classification accuracy values of the Decision Tree determined in Section 4. The reported results were calculated for data without feature selection.

Additional classifiers developed using machine learning algorithms perform differently in classifying cyber-attacks. The best classification accuracy is demonstrated by the Gradient Boosting and Random Forest algorithms. These two algorithms classify cyber-attacks with an accuracy of more than 99%. A high classification accuracy of more than 94% is demonstrated by the SVM algorithm. The Lasso classifier and Ridge classifier algorithms classify cyber-attacks with the lowest accuracy, around 80%. This can be explained that they are basically regression algorithms but with a classification option. In the task of classifying cyber-attacks, they did not perform well.

Of the additional algorithms, only two i.e., Gradient Boosting and Random Forest achieved marginally higher classification accuracy than the Decision Tree algorithm. Thus, these two algorithms should be considered as alternatives in future studies. An in-depth study of cyber-attack classification using these two best additional algorithms should be performed.

5 Summary

To summarize the results of the accuracy analysis, both splitting approaches (Gini and Entropy) demonstrated very high performance across most classes and models. The accuracy and precision values for both were very close, maintained at the expected level. Differences between values are minimal and measured values are high. The classification effectiveness confirms that the majority of positively classified cases were indeed correct.

For *DoS* classification, both splitting methods (Gini and Entropy) achieve very high accuracy, precision, recall and F-size, showing their potential in detecting Denial of Service attacks. Similarly, the models performed well in *Probe*

Table 13. Hyperparameters tuning for the Lasso Classifier and Ridge Classifier models.

Model	Parameters	Set of values	Optimal value
Lasso Classifier	Alpha	0.01; 0.1; 1; 10	0.01
Ridge Classifier	Alpha	0.01; 0.1; 1; 10	10

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

Table 14. Average classification accuracy for all models.

Models	Accuracy
Gradient Boosting	99.67%
AdaBoost	84.03%
SVM	94.22%
Lasso	78.69%
Ridge	80.14%
Random Forest	99.56%
Decision Tree – Gini	98.18%
Decision Tree – Entropy	98.93%

attack classification, although with slightly lower precision and recall than for DoS. *R2L* attack classification turned out to be more difficult, with acceptable but less impressive results, suggesting that there is some room for further optimization. The most challenging class in classification was *U2R*, where precision and accuracy were the lowest. Classification of U2R attacks still remains a challenge, especially in terms of precision. Increasing efficiency of U2R classification is important to minimise the risk of privilege escalation for unauthorised users.

Overall, the Gini and Entropy DT-building approaches prove to be useful in cyberattack classification. There is a potential for further optimisation, especially in the area of attacks with more advanced characteristics, such as U2R. The introduction of new training data or parameter refinement could also contribute to the improvement of their ability to detect more advanced forms of attacks.

Comparative analysis of alternative machine learning classifiers showed considerable performance differences. *Gradient Boosting* and *Random Forest* achieved the highest precision (99. 67% and 99. 56%, respectively). Decision tree-based models performed also very well: *Decision Tree based on Entropy*, with an accuracy of 98.93%, and the *Decision Tree based on Gini*, with an accuracy of 98.18%. *SVM* performed well with 94.22% accuracy, making it a valid choice for small or medium-sized data sets. The Lasso and Ridge regression algorithms showed rather poor effectiveness (78.69% and 80.14%, respectively) compared to the aforementioned ones.

These findings indicate that, while decision tree-based approaches are highly effective, Gradient Boosting and Random Forest algorithms could be positioned as strong alternatives, considering their high accuracy and relatively low computational requirements. This conclusion gives a good basis for further exploration and optimization.

However, some limitations remain, particularly in detecting low-frequency or stealthy attacks such as U2R, where the models struggled with precision and recall. Future research could focus on improving detection capabilities through techniques such as anomaly-based learning, data augmentation, or hybrid ensemble models. Including practical case studies or deployment scenarios would help demonstrate the real-world applicability of the proposed models, while the use of clearer graphs and visualizations could further support interpretability and accessibility for a broader audience.

The authors might consider further exploring the practical implications and potential applications of their findings in real-world cybersecurity systems, such

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

as intrusion detection platforms or automated threat response mechanisms. Integrating these models into operational environments could provide valuable information on their robustness and adaptability under realistic conditions.

References

1. Agarwal, R., Dhoot, A., Kant, S., Bisht, V.S., Malik, H., Ansari, M.F., Afthanorhan, A., Hossaini, M.A.: A novel approach for spam detection using natural language processing with amals models. IEEE Access (2024)
2. Akhi, A.B., Kanon, E.J., Kabir, A., Banu, A.: Network intrusion classification employing machine learning: A survey. Ph.D. thesis (2019)
3. Bassiouni, M., Ali, M., El-Dahshan, E.: Ham and spam e-mails classification using machine learning techniques. J of Appl Sec Research **13**(3), 315–331 (2018)
4. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407 (2019)
5. Chen, C., Zhang, J., Xie, Y., Xiang, Y., Zhou, W., Hassan, M.M., AlElaiwi, A., Alrubaian, M.: A performance evaluation of machine learning-based streaming spam tweets detection. IEEE Trans. on Computational Social Systems **2**(3), 65–76 (2015)
6. Das, L., Ahuja, L., Pandey, A.: Analysis of twitter spam detection using machine learning approach. In: 2022 3rd International Conference on Intelligent Engineering and Management (ICIEM). pp. 764–769. IEEE (2022)
7. Diale, M., Celik, T., Van Der Walt, C.: Unsupervised feature learning for spam email filtering. Computers & Electrical Engineering **74**, 89–104 (2019)
8. Elhag, S., et al.: On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems. Expert Systems with Applications **42**(1), 193–202 (2015)
9. Etaiwi, W., Awajan, A.: The effects of features selection methods on spam review detection performance. In: 2017 International Conference on New Trends in Computing Sciences (ICTCS). pp. 116–120. IEEE (2017)
10. García, S., Luengo, J., Herrera, F., et al.: Data preprocessing in data mining, vol. 72. Springer (2015)
11. Gharaee, H., Hosseinvand, H.: A new feature selection ids based on genetic algorithm and svm. In: 2016 8th International Symposium on Telecommunications (IST). pp. 139–144. IEEE (2016)
12. Jain, V., Phophalia, A.: M-ary random forest-a new multidimensional partitioning approach to random forest. Multimedia Tools and Applications **80**(28), 35217–35238 (2021)
13. Kaur, D., Bedi, R., Gupta, S.K., et al.: Review of decision tree data mining algorithms: Id3 and c4. 5. In: International Conference on Information Technology and Computer Science. pp. 5–8 (2015)
14. Kirubai, J.C.D., Priscila, S.S.: MLP-based defense mechanisms against cyber crime: Insights from dataset-driven attack prevention strategies. In: 2024 Int. Conf. on Data Science and Network Security (ICDSNS). pp. 01–06. IEEE (2024)
15. Li, Y., Chen, G., Dong, Z.: Multi-view graph contrastive representative learning for intrusion detection in ev charging station. Applied Energy **385**, 125439 (2025)
16. Lin, W.C., et al.: Cann: An intrusion detection system based on combining cluster centers and nearest neighbors. Knowledge-based systems **78**, 13–21 (2015)
17. Liu, Z., Ghulam, M.U.D., Zhu, Y., Yan, X., Wang, L., Jiang, Z., Luo, J.: Deep learning approach for IDS: using DNN for network anomaly detection. In: 4th Int. Congress on Information and Comm. Tech.: ICICT. pp. 471–479. Springer (2020)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4

18. Machalewski, T., Szymanek, M., Czubak, A., Turba, T.: Expressing impact of vulnerabilities: An expert-filled dataset and vector changer framework for modelling multistage attacks, based on CVE, CVSS and CWE. In: ECMS 2024 Proceedings. pp. 569–578 (2024), https://doi.org/10.7148/2024-0569, accessed: 2025-02-20
19. Mahindru, A., Sangal, A.L.: Fsdroid:-a feature selection technique to detect malware from android using machine learning techniques: Fsdroid. Multimedia Tools and Applications **80**, 13271–13323 (2021)
20. Mining, W.I.D.: Data mining: Concepts and techniques. Morgan Kaufinann **10**(559-569), 4 (2006)
21. Mishra, P., Varadharajan, V., Tupakula, U., Pilli, E.S.: A detailed investigation and analysis of using machine learning techniques for intrusion detection. IEEE communications surveys & tutorials **21**(1), 686–728 (2018)
22. Najadat, H., Abdulla, N., et al.: Mobile sms spam filtering based on mixing classifiers. Int. J of Advanced Computing Research **1**, 1–7 (2014)
23. Ouarda, A.: Image thresholding using type-2 fuzzy c-partition entropy and particle swarm optimization algorithm. In: International Conference on Computer Vision and Image Analysis Applications. pp. 1–7. IEEE (2015)
24. Rusland, N.F., Wahid, N., Kasim, S., Hafit, H.: Analysis of naïve bayes algorithm for email spam filtering across multiple datasets. In: IOP conference series: materials science and engineering. vol. 226, p. 012091. IOP Publishing (2017)
25. Sagar, R., Jhaveri, R., Borrego, C.: Applications in security and evasions in machine learning: a survey. Electronics **9**(1), 97 (2020)
26. Sarnovsky, M., Paralic, J.: Hierarchical intrusion detection using machine learning and knowledge model. Symmetry **12**(2), 203 (2020)
27. Udurume, M., et al.: Comparative evaluation of network-based intrusion detection: Deep learning vs traditional machine learning approach. In: 2024 15th Int. Conf. on Ubiquitous and Future Networks (ICUFN). pp. 520–525. IEEE (2024)
28. Vielberth, M.: Security information and event management (siem). In: Encyclopedia of Cryptography, Security and Privacy, pp. 1–3. Springer (2021)
29. Vineeth, G.V.S.S.K., et al.: Email spam: A new strategy of screening spam emails using natural language processing. In: 2023 3rd Int. Conf. on AI and Smart Energy (ICAIS). pp. 710–715. IEEE (2023)
30. Xu, H., Sun, W., Javaid, A.: Efficient spam detection across online social networks. In: 2016 IEEE Int. Conf. on Big Data Analysis (ICBDA). pp. 1–6. IEEE (2016)
31. Yilmaz, A.B., Taspinar, Y.S., Koklu, M.: Classification of malicious android applications using naive bayes and support vector machine algorithms. International Journal of intelligent systems and applications in engineering **10**(2), 269–274 (2022)
32. Zhang, A., Lipton, Z.C., Li, M., Smola, A.J.: Dive into deep learning. Cambridge University Press (2023)
33. Zhour, R., Khalid, C., Abdellatif, K.: Hybrid intrusion detection system based on random forest, decision tree and multilayer perceptron (mlp) algorithms. In: 2023 10th International Conference on Wireless Networks and Mobile Communications (WINCOM). pp. 1–5. IEEE (2023)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_4