Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871

Vol. 6, No. 5, October 2025, Page. 3405-3418
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.5.3817

Accelerating Classification For Iot Attack Detection Using Decision Tree Model With Gini Impurity Tree-Based Feature Selection Technique

Muhammad Hafizh Dzaki*[1], Adhitya Nugraha[2], Ardytha Luthfiarta[3], Azizu Ahmad Rozaki Riyanto[4], Yohanes Deny Novandian[5]

[1,2,3,4,5]Informatics Engineering, Universitas Dian Nuswantoro, Indonesia

Email: [1]muhammadhafizhdzaki@gmail.com

Received : Sep 27, 2024; Revised : Nov 3, 2024; Accepted : Nov 11, 2024; Published : Oct 16, 2025

Abstract

The Internet of Things (IoT) continues to expand rapidly, with the number of connected devices expected to reach billions in the near future. However, it makes IoT devices prime target for cyber-attack. Therefore, an effective Intrusion Detection System (IDS) is required to detect these attacks swiftly and accurately. This study aims to build a machine learning-based IDS to effectively detect attack on IoT network using the CIC IoT 2023 dataset. The dataset contains over 46 million data rows with 48 features, covering 33 attack types and 1 benign class. To address the dataset's complexity and enhance processing efficiency, feature selection technique was applied. Six feature selection techniques from three categories – Filter-based, Wrapper-based, and Hybrid methods – were evaluated to produce the best feature subset. Each subset was tested using a Decision Tree algorithm. Then, the model performance calculated based on accuracy, computational time, as well as macro-precision, -recall, and -F1-score. The results demonstrate that the three best feature selection from each category – Mutual Information, Genetic Algorithm, and Gini Impurity Tree-based – improved training time by average different 55 seconds from 148 seconds, which speed up by 63.06% without sacrificing accuracy. The Gini Impurity Tree-based algorithm proved to be the most efficient, producing the smallest feature subset, which is 10 features, faster processing times, which is 40 seconds, and shallower tree's depth, which is 64 level from 73 level. In conclusion, feature selection not only enhances computational efficiency but also simplifies tree's shape without sacrificing the accuracy of detection.

Keywords : *Decision Tree, Feature Selection, Internet of Things, IoT Attack.*

1. INTRODUCTION

Internet of Things (IoT) refers to a network of physical or virtual devices that can connect to the internet and communicate with each other [1]. This concept has been implemented in several sectors, such as agriculture, household appliances, and wearable goods. By the end of 2020, there were 11,3 billion connected IoT devices. This number is expected to continue to grow, reaching approximately 27,1 billion by 2025 [2]. This growth occurs because the ease of data collection and process automation are the advantages offered by IoT devices [3], [4].

Despite all its advantages, the interconnected IoT network presents vulnerabilities for malicious actors to launch attack. To address this issue, the demand for an Intrusion Detection System (IDS) that capable of detecting suspicious activities that may lead to IoT attack within a network is very high. An IDS is a network security tool that can be embed into IoT device to monitor network traffics and devices for known malicious activity, suspicious behavior, or security policy violation [5]. IDS systems that can detect network anomaly accurately and swiftly are preferred.

As cited from Splunk [6], a company acquired by CISCO, Intrusion Detection Systems (IDS) are divided into several types. One of them is Anomaly-based Detection (AD IDS), where the core technology of this type of IDS is built using statistical functions, knowledge-based method, and machine learning application. Implementing this IDS involves a model that responds to all network traffics, allowing safe traffic and rejecting traffic with attack pattern. The main advantage of AD IDS is its ability to operate in flexible operating system environment and its capability to detect new attack pattern, provided that the model was well-trained [6], [7]. However, there is one thing that needs to be noted. According to [8], IoT device has limited resource and computational power. So, it is a necessary to make sure that AD IDS can be deployed and embed into IoT device.

With the increasing complexity of network attack threats, AI-supported AD IDS offers improved accuracy, speed, and comprehensive approach to recognizing new attack pattern. Therefore, the application of AI-based IDS is increasingly prioritized to meet the need for more adaptive and responsive security [6], [7]. One of the key pillars in the development of AI-based IDS is machine learning. Machine learning is highly suitable for implementation in IDS systems due to its ability to continuously adapt and improve its performance as more attack data is processed, making it an ideal solution to address the ever-evolving type of attack [6], [9].

From 20 previous journals on IDS using machine learning technology, a satisfactory accuracy was achieved, with the lowest accuracy being 71% and the highest reaching 99% [10]. Therefore, machine learning plays a crucial role in detecting IoT attack in IDS [11]. To build an effective machine learning model for detecting attack in IoT networks, a comprehensive dataset is required, covering a wide range of scenarios and relevant attack variations, as well as being representative of real-world production condition. There are several publicly available IoT attack datasets, such as the TON IoT and BoT-IoT dataset, which have been widely used in previous studies [12], [13]. However, this study utilizes the CIC IoT 2023 dataset, developed by the Canadian Institute for Cybersecurity and released by the University of New Brunswick [14]. This choice is based on several advantages offered by CIC IoT 2023, such as a broader range of attack types and variations, more up-to-date, and modern attack data. Also, the large dataset size that is relevant for testing various machine learning-based attack detection techniques [14].

The dataset comprises over 46 million records and includes 48 features. If a machine learning model immediately created from this huge size dataset, it will also create a more complex and bigger machine learning model [15]. Recalling previous statement that said "IoT device has limited resource and computational power", this concept presents a challenge in developing a machine learning-based AD IDS for detecting IoT attack that also be able to deployed into IoT device. This issue is due to the size and data processing complexity, requiring significant computational resources.

To address this, feature selection method must be applied to mitigate these challenges [16]. The primary objective of this study is to detect IoT attack using machine learning algorithm to ensure accurate and effective detection. The IoT attack detection process will involve a feature selection stage using various techniques, ultimately producing the best feature subset for detecting IoT attack in CIC IoT 2023 dataset.

2. METHOD

2.1. Flowchart Workflow

Figure 1 shows the flowchart for the machine learning classification task in this study, detailing step-by-step process from data preprocessing to model evaluation. Here the key steps this study have done:

a. The process begins with loading the CIC IoT 2023 dataset.
b. In the preprocessing step, unnecessary columns, missing or empty values, and duplicates data are removed. Then, the dataset is split into training and testing sets.
c. Resampling technique are applied to the training dataset only to address class imbalance.

d. Various feature selection techniques are used to create feature subset, which are stored in a list. Both of training and testing datasets then transformed using the selected features.
e. Before classification, both datasets are normalized using a normalization technique.
f. A classification model is built using the dataset by applying selected features
g. Finally, the model is evaluated based on performance metrics.

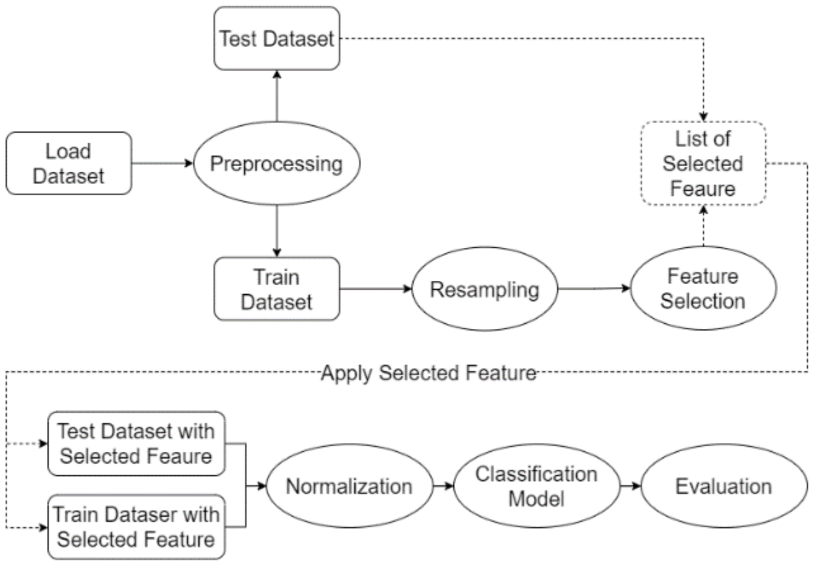

Figure 1. Research Work Steps

2.2. Dataset

This study utilizes the CIC IoT 2023 dataset, developed by Canadian Institute for Cybersecurity and released by the University of Brunswick, Canada [14]. The dataset covers over 46 million rows of data and 47 features, along with 1 target column. It includes 33 types of attack and 1 type of non-malicious network traffic (Benign). Class distribution can be seen in the Figure 2. By following the documentation provided, 33 types of attack can be grouped into 7 categories of attack. Thus, a total of 8 targets are formed, as visualized in Figure 3.

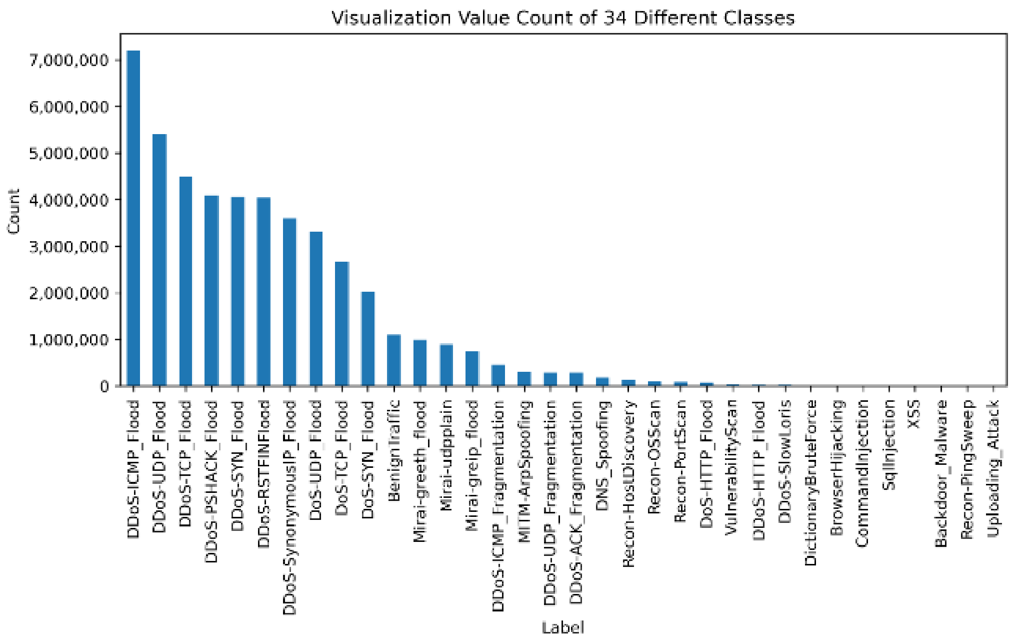

Figure 2. Class Distribution of CIC IoT 2023 with 34 Classes

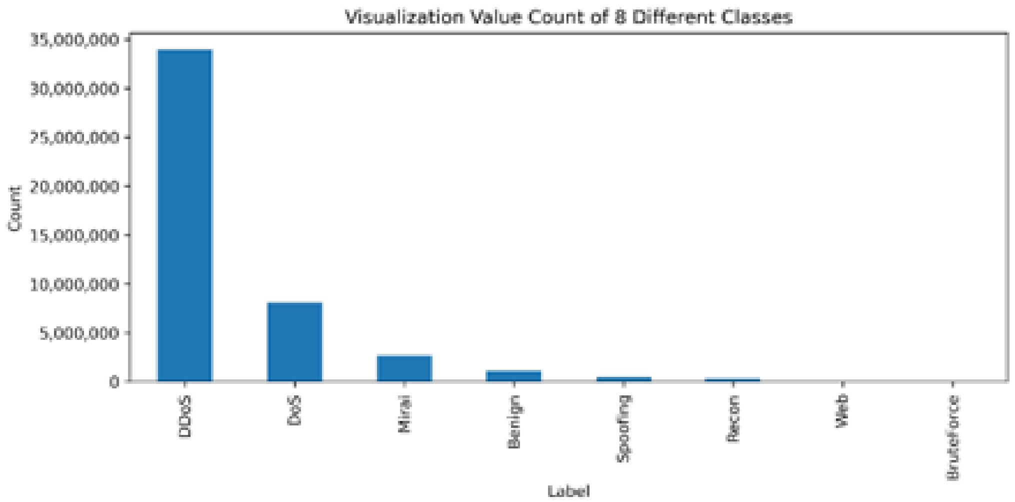

Figure 3. Class Distribution of CIC IoT 2023 with 8 Classes

2.3. Preprocessing and Splitting

Preprocessing is a crucial step in machine learning. Proper preprocessing will result in a more efficient dataset for model training process [13], [17], [18]. In this study, several preprocessing steps were carried out to prepare the dataset. The first step is to remove unnecessary columns, such as the incremental id. Next, missing or empty values will be removed from the dataset. Additionally, duplicated values will be eliminated from the dataset. As a result, it can be ensured that all rows of data have unique values. Finally, the dataset will be split into training and testing sets. The training-testing split ratio is 9:1. For more details, the Table 1 shows the amount of data in each split.

Table 1. The Amount of Data from Train and Test Dataset

Category	Label	Data Train Value Count	Data Test Value Count
DDoS	0	30586079	3398454
DoS	1	7281663	809074
Mirai	2	2370712	263412
Benign	3	988375	109820
Spoofing	4	437853	48650
Recon	5	319104	35456
Web	6	22346	2483
Brute Force	7	11758	1306

2.4. Resampling

The resampling process is conducted to balance the class distribution so that the model does not become biased toward the majority class during the training process. Resampling is also performed to ensure that the minority class can be better generalized by the model [19]. In this study, the resampling process was done using a combination of two algorithms: Random Undersampling, which is used to reduce the amount of data in the majority class, and Random Oversampling, which is used to increase the amount of data in the minority class. These two techniques were selected based on the research from [20], which stated that Random Under- and Over-sampling yielding a good result and suitable for dataset with extreme class imbalance. It can be seen from the Table 1 that the class distribution in the CIC IoT 2023 dataset is highly imbalanced. Therefore, the combination of these two resampling methods is appropriate for implementation in this study.

The resampling process was applied only to the training data. The goal is for the model to learn from balanced data during the training process, avoiding the risk of overfitting and improving performance on the test data [21]. Meanwhile, the test data was on purpose left imbalanced to reflect a real-world scenario, where the type of attack vary and are not consistent every day.

2.5. Feature Selection

The process of feature selection in dataset aims to reduce dimensionality. As a result, the required computational resources are minimized, and the time needed for training is also decreased. In this study, the feature selection was performed using six different methods to find the optimal subset of feature. This step is crucial for reducing the dataset's complex dimension by discarding less informative feature and keeping those most relevant to the dataset's label [21], [22].

In ”A Survey On Feature Selection Method For Product Review” paper, there were 21 papers comparing feature selection algorithms, divided into three categories: filter-based, wrapper-based and hybrid-method [23]. Filter-based utilize predetermined criteria to evaluate the relevance of features based on statistical calculations from the data without involving any machine learning algorithms [22]. In this study, the Fisher Score and Mutual Information techniques are included in this category.

The second category is the wrapper-based, which iteratively evaluates features by involving machine learning algorithm to assess the performance of feature subset from the dataset [22]. This study used the Recursive Feature Elimination algorithm with Logistic Regression as the estimator, as well as the Genetic Algorithm. Generally, wrapper methods have advantages in terms of accuracy because they directly consider the performance of the wrapped model, although they tend to be more time-consuming due to repeatedly performing the evaluation process [22].

The last category is hybrid method that combine the advantages of both filter and wrapper technique [22]. In this study, a Gini Impurity Tree-based technique employed to measure the extent to which a feature separates the target based on Gini Impurity. This method is considered hybrid because the computation of Gini Impurity is already embedded within the tree algorithm itself. A similar approach is present in the Lasso algorithm, which inherently applies L1 regularization during the training process. L1 regularization effectively reduces some coefficients to zero, thus retaining only the most important features within the model [24].

After applying all six feature selection techniques, the subset produced by each algorithm were stored in separate variable. The selected features then applied to the training and testing data. Consequently, new six pairs of training and testing datasets were generated, each corresponding to one of the feature selection methods. By employing various feature selection technique from the three existing categories, this study aims to identify method that offer the best combination of features, thereby enhancing the accuracy and efficiency of the classification process of the applied machine learning models within each category.

2.6. Normalization

Before moving to the classification model training stage, all generated datasets will be normalized first. Normalization is utilized to ensure that each feature has value on the same scale, preventing the potential dominance of certain feature that have a larger value range than other [25].

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad (1)$$

In this study, the MinMaxScaler method was utilized as a normalization technique, mapping feature into the range between 0 and 1. The formula is formulated in Point 1. This process is crucial because certain machine learning algorithm, such as Logistic Regression, XGBoost, and LightGBM, are susceptible to data scaling [25]. By consistently applying normalization to both the training and testing

Vol. 6, No. 5, October 2025, Page. 3405-3418
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.5.3817

data, expected that the resulting model will exhibit improved performance and stability in classification tasks.

2.7. Model Classification and Evaluation

This study utilizes one machine learning algorithm, namely Decision Tree algorithm. This algorithm was selected because it has its own approach to handling tabular data by partitioning the data based on statistical measures of uncertainty using specific criteria. The Decision Tree algorithm constructs hierarchical rules by creating nodes at each decision branch [26]. As observed in previous research, the Decision Tree algorithm also demonstrates good performance in many studies in the literature [11], [13], [21], [27]. Furthermore, the Decision Tree algorithm yields a tree which structure, shape, and level of the tree are easily identify. After the model is formed, the prediction process using the test data can be done. From the prediction result, performance of model is evaluated using several metrics commonly used in classification task [20].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

$$Macro\ Precision = \frac{1}{n}\sum_{i=1}^{n}\frac{TP_i}{TP_i+FP_i} \quad (3)$$

$$Macro\ Recall = \frac{1}{n}\sum_{i=1}^{n}\frac{TP_i}{TP_i+FN_i} \quad (4)$$

$$Macro\ F1\text{–}Score = \frac{1}{n}\sum_{i=1}^{n} 2 \times \frac{Precision_i \times Recall_i}{Precision_i+Recall_i} \quad (5)$$

3. RESULT

A series of experiments were conducted in a server environment equipped with an AMD EPYC 7742, 64-Core Processor, with a total memory capacity 150 GB. This study utilized the Python programming language, specifically version 3.10.10.

3.1. Result of Resampling Technique

The results of the resampling process using a combination of the Random Undersampling and Random Oversampling algorithms can be observed in the Table 2. To assure a balanced effect of the Undersampling and Oversampling techniques, the value of 437,853 from Spoofing class was selected. Using this value, the data size from four dataset classes was decreased, while the rest classes was increased.

Table 1. The Amount of Data Before and After Resampling

Category	Label	Before Resampling	After Resampling
DDoS	0	30586079	437853
DoS	1	7281663	437853
Mirai	2	2370712	437853
Benign	3	988375	437853
Spoofing	4	437853	437853
Recon	5	319104	437853
Web	6	22346	437853
Brute Force	7	11758	437853

3.2. Result of Feature Selection

Table 3. Information Related to Feature Selection Technique

Feature Selection Techniques	Time Consumed	Top Feature Used
Fisher Score	< 3 minutes	10
Mutual Information	> 20 minutes	20
Recursive Feature Elimination	> 20 minutes	22
Genetic Algorithm	> 15 hours	22
Gini Impurity Tree-based	< 5 minutes	10
Lasso-based	< 3 minutes	10

Each feature selection technique will return the number of columns required. Table 3 presents the number of columns used in this study and other information related to the techniques. Wrapper-based feature selection directly returns the name of the features selected by the algorithm. That's why Wrapper-based feature selection technique returns the longest subset than the other technique. Meanwhile, the Filter-based and Hybrid-method feature selection only return the ranking of feature importance values for each feature. Therefore, a user-predefined value is required to determine the number of features to be used. In this study, the value of the user-predefined value was based on the analysis of images.

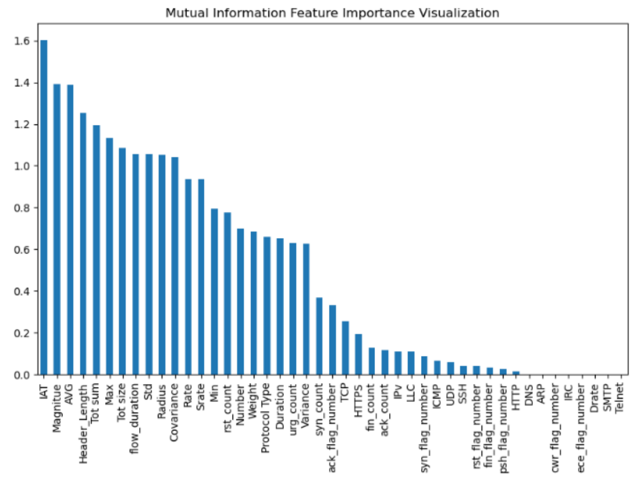

Figure 4. Mutual Information Feature Importance Visualization

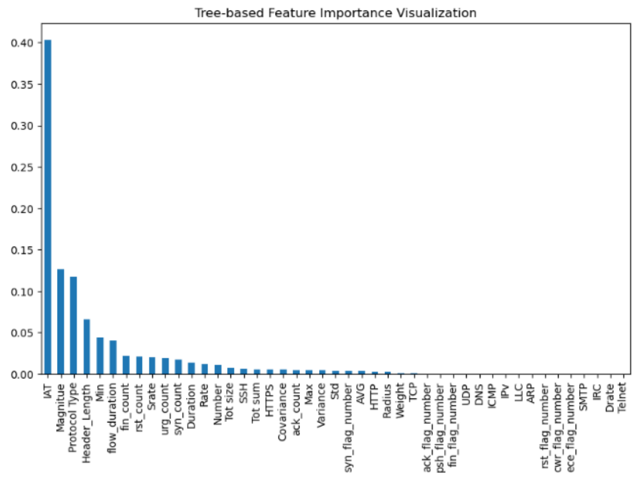

Figure 5. Gini Impurity Tree-based Feature Importance Visualization

To interpret the visualization, the taller the bar chart meaning respective feature has higher influential value on determining the classification label, and vice versa. Based on Figure 4, Mutual Information technique used the top 20 features due to a more gradual distribution, while the Gini Impurity Tree-based technique on Figure 5 used top 10 features due to a steeper distribution. For more detailed selected feature, Table 4 will display briefly by showing only top 10 most influential feature importance from each technique.

Table 4.1. Top 10 Feature Importance for Each Technique

	Fisher Score	Mutual Information	RFE-LR
Features Selected	Variance	IAT	Duration
	Protocol Type	Magnitue	syn_count
	ack_flag_number	AVG	fin_count
	Magnitue	Header_Length	Std
	HTTPS	Tot sum	Tot size
	TCP	Max	syn_flag_number
	Min	Tot size	rst_flag_number
	Std	flow_duration	psh_flag_number
	Radius	Std	ack_flag_number
	rst_count	Radius	HTTP

Table 4.2. Top 10 Feature Importance for Each Technique

	Genetic Algorithm	Gini Impurity Tree-based	Lasso-based
Features Selected	flow_duration	IAT	Protocol Type
	Header_Length	Magnitue	fin_flag_number
	Protocol Type	Protocol Type	syn_count
	Srate	Header_Length	urg_count
	Drate	Min	HTTP
	rst_flag_number	flow_duration	HTTPS
	psh_flag_number	fin_count	SSH
	ack_flag_number	rst_count	TCP
	cwr_flag_number	Srate	ICMP
	ack_count	urg_count	Min

3.3. Result of Classification Model

In this study, the classification model used Decision Tree algorithm. Each training dataset from the respective feature selection method was trained and then tested using the corresponding test data. The result of the experiments are shown in Table 5. Each Decision Tree model creates a tree structure that can be identified. The description of the tree's structure generated by each feature selection technique are outlined in Table 6. For more clearly and easier insight, figure 6, figure 7, figure 8, and figure 9 are the visualization from table 5 and table 6.

Figure 6 visualizes the duration of Decision Tree model accross different feature selection techniques. The model took longest time to train when no feature selection technique applied (148 seconds). While, Lasso-method resulted in the shortesest training time (24 seconds).

Figure 7 visualizes the comparison between 4 metrics of Decision Tree model accross different feature selection techniques. Accuracy (blue bar) seems relatively consistent accross all techniques with slight variations, as Mutal Information, Genetic Algorithm, and Gini Impurity Tree-based are the highest among other techniques. Macro Precision (red bar), Macro Recall (yellow bar), and Macro F1-Score (green bar) varies more noticeably, as Fisher Score and Recursive Feature Elimination are lower among others and Mutual Information, Genetic Algorithm, and Gini Impurity Tree-based are higher among other feature selection techniques.

Figure 8 depicts the depth of tree generated using different feature selection techniques. Overall, by applying feature selection technique, It results lower tree depth. Recursive Feature Elimination tends to have deepest tree depth amongs the other techniques. While Genetics Algorithm tends to have the shallowest depth among the other techniques.

Table 5. Decision Tree Model Experiment Results

Feature Selection Techniques	Decision Tree Metrics	Model Value and Score
None (Raw Dataset)	Duration (Sec)	148
	Accuracy	0.994
	Macro Precision	0.839
	Macro Recall	0.867
	Macro F1-Score	0.852
Fisher Score	Duration (Sec)	36
	Accuracy	0.742
	Macro Precision	0.495
	Macro Recall	0.541
	Macro F1-Score	0.509
Mutual Information	Duration (Sec)	81
	Accuracy	0.994
	Macro Precision	0.828
	Macro Recall	0.854
	Macro F1-Score	0.840
Recursive Feature Elimination	Duration (Sec)	30
	Accuracy	0.740
	Macro Precision	0.533
	Macro Recall	0.613
	Macro F1-Score	0.553
Genetic Algorithm	Duration (Sec)	43
	Accuracy	0.994
	Macro Precision	0.839
	Macro Recall	0.861
	Macro F1-Score	0.850
Gini Impurity Tree-based	Duration (Sec)	40
	Accuracy	0.994
	Macro Precision	0.828
	Macro Recall	0.852
	Macro F1-Score	0.839
Lasso-based	Duration (Sec)	24
	Accuracy	0.730
	Macro Precision	0.529
	Macro Recall	0.597
	Macro F1-Score	0.543

Table 6. Shape of the Tree Model from Each Technique

Feature Selection Techniques	Tree Depth	Number of Leaves	Number of Nodes
None (Raw Dataset)	73	91745	183489
Fisher Score	71	396108	792215
Mutual Information	65	100406	200811
Recursive Feature Elimination	73	389932	779863
Genetic Algorithm	63	100882	201763
Gini Impurity Tree-based	64	107643	215285
Lasso-based	65	397637	795273

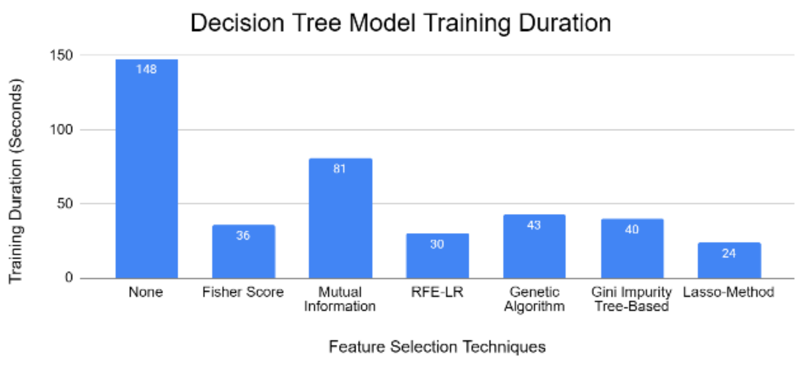

Figure 2. Decision Tree Model Training Duration

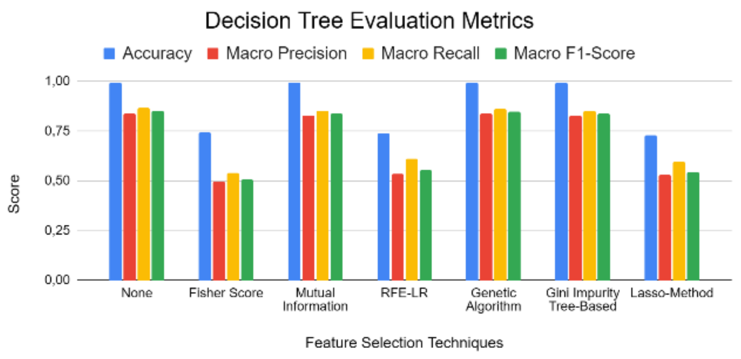

Figure 3. Decision Tree Evaluation Metrics

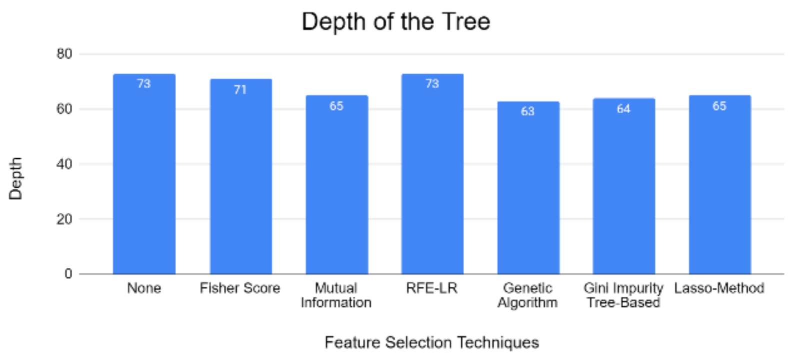

Figure 4. Depth of Created Tree

Figure 9 shows the overall shape of the tree created. Overall, feature selection technique creating broader tree . While, without applying feature selection techniques creating deeper tree. It shows Fisher

Score, Recursive Feature Elimination, and Lasso-method have broader tree than Mutual Information, Genetic Algorithm, and Gini Impurity Tree-based techniques.

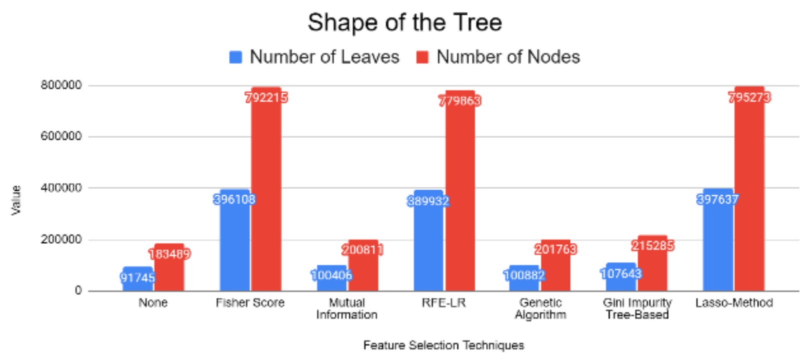

Figure 5. Shape of The Tree

4. DISCUSSIONS

Experiment uses Decision Tree algorithm as a classification model. Based on the Table 5, each category of feature selection has one technique that gives the best results. The Filter-based category has a Mutual Information algorithm, the Wrapper-based category has a Genetic Algorithm, and the Hybrid-method category has a Gini Impurity Tree-based. From this point onwards, these three algorithms will be called the “3 Best”. Meanwhile, datasets without the usage of selection features will be called “None”.

The comparative results between 3 Best and None do not show significant differences across several metrics. Both Decision Tree models achieve an accuracy of 0.994. Additionally, the macro-precision, macro-recall, and macro-F1-score values show average differences of 0.87%, 1.31%, and 1.06%, respectively. However, it is evident that 3 Best offers significant advantages in model training time, with an average processing duration 63.06% faster than None. Furthermore, the shape of tree created by the model, in terms of the average difference in the depth by 3 Best, is 12.33% lower than None. However, the reduced depth results in 3 Best having an average difference in the number of leaves and nodes that is 12.24% higher than None.

Based on data from Table 6, It was found that tree produced through the feature selection process creates a broader structure, while those without feature selection have a deeper structure. Deeper trees naturally result in slower final decision-making. Moreover, increased tree depth indicates a more complex decision hierarchy and decision boundaries, where the model tends to overemphasize differences within the data. Consequently, this makes the model more susceptible to overfitting.

Among the 3 best techniques, the Gini Impurity Tree-based feature selection is the most effective in identifying the most important subset of features. It has been demonstrated that the Gini Impurity Tree-based algorithm produces the smallest dataset subset, consisting of only 10 features, achieves the fastest processing time of less than 5 minute and results in a relatively low tree’s depth by 64. For further comparison, Table 7 presents a comparison of several previous studies that investigated feature selection techniques.

Research	Method	Dataset Used	Training Accuracy
[28]	No Feature Selection + Random Forest	CIC IoT 2023	0.994
[29]	RFE-LR + Logistic Regression	CIC IoT 2023	0.978
[30]	No Feature Selection + LSTM	CIC IoT 2023	0.985
[31]	RFE-Decision Tree + Decision Tree	CIC IoT 2023	0.873
This Study	Gini Impurity Tree-based + Decision Tree	CIC IoT 2023	0.994

5. CONCLUSION

The primary objective of this study is to detect IoT attack using machine learning in order to achieve accurate and effective detection. The complexity of the dataset is addressed by using feature selection methods. Seven algorithms from three categories of feature selection were tested on the CIC IoT 2023 dataset. Each algorithm provided its own best subset of feature. Three top-performing algorithms were identified, representing the best from each category, with the Gini Impurity Tree-based algorithm emerging as the best feature selection algorithm. This algorithm produced a Decision Tree model with high accuracy, Macro-Precision, -Recall, and -F1-score. The complexity level of the Decision Tree model proved to be lower than None, with a reduced depth of 64, which result in faster prediction time and a more streamlined hierarchical structure.

CONFLICT OF INTEREST

The authors declare that they have no financial or personal relationships that could have appeared to influence the work reported in this paper. There are no conflicts of interest among the authors, nor with any organizations or entities related to the research object presented in this study.

ACKNOWLEDGEMENT

This study utilized the CIC IoT 2023 Dataset provided by the University of New Brunswick. The authors extend their deepest appreciation toward University of New Brunswick for valuable contribution to this study. This dataset played a crucial role in developing the machine learning-based intrusion detection system for detecting IoT attack.

REFERENCES

[1] A. K. M. Al-Qurabat, “A Lightweight Huffman-based Differential Encoding Lossless Compression Technique in IoT for Smart Agriculture,” *International Journal of Computing and Digital Systems*, vol. 11, no. 1, pp. 117–127, Jan. 2022, doi: 10.12785/ijcds/110109.
[2] M. Selvaraj and G. Uddin, “A Large-Scale Study of IoT Security Weaknesses and Vulnerabilities in the Wild,” Aug. 2023.
[3] S. Rode, R. Saraf, J. Veigas, N. Shetty, and S. Shardul, “Design and Fabrication of IoT based Agricultural Automation system,” in *2023 5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*, IEEE, Jan. 2023, pp. 1–6. doi: 10.1109/ICNTE56631.2023.10146721.
[4] M. Servetnyk and R. Servetnyk, “Emerging applications, technologies, and services in wireless communications: 5G to 6G evolution,” *Journal of Scientific Papers “Social development and Security,”* vol. 11, no. 2, Apr. 2021, doi: 10.33445/sds.2021.11.2.1.
[5] “Apa itu Sistem Deteksi Intrusi (IDS)? | IBM.” Accessed: Sep. 22, 2024. [Online]. Available: https://www.ibm.com/id-id/topics/intrusion-detection-system
[6] “Intrusion Detection Systems (IDS): Definition, Types, Purpose | Splunk.” Accessed: Sep. 24, 2024. [Online]. Available: https://www.splunk.com/en_us/blog/learn/ids-intrusion-detection-systems.html
[7] S. Sadhwani, B. Manibalan, R. Muthalagu, and P. Pawar, “A Lightweight Model for DDoS Attack Detection Using Machine Learning Techniques,” *Applied Sciences 2023, Vol. 13, Page 9937*, vol. 13, no. 17, p. 9937, Sep. 2023, doi: 10.3390/APP13179937.
[8] M. S. Mazhar *et al.*, “Forensic Analysis on Internet of Things (IoT) Device Using Machine-to-Machine (M2M) Framework,” *Electronics 2022, Vol. 11, Page 1126*, vol. 11, no. 7, p. 1126, Apr. 2022, doi: 10.3390/ELECTRONICS11071126.
[9] A. R. Gad, A. A. Nashat, and T. M. Barkat, “Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset,” *IEEE Access*, vol. 9, pp. 142206–142217, 2021, doi: 10.1109/ACCESS.2021.3120626.

[10] U. S. Musa, M. Chhabra, A. Ali, and M. Kaur, "Intrusion Detection System using Machine Learning Techniques: A Review," *Proceedings - International Conference on Smart Electronics and Communication, ICOSEC 2020*, pp. 149–155, Sep. 2020, doi: 10.1109/ICOSEC49089.2020.9215333.
[11] G. Rathod, V. Sabnis, and J. K. Jain, "Improving IoT Botnet Attack Detection using Machine Learning: Comparative Analysis of Feature Selection Methods and Classifiers in Intrusion Detection Systems," *2024 3rd International Conference for Innovation in Technology, INOCON 2024*, 2024, doi: 10.1109/INOCON60754.2024.10511883.
[12] F. De Keersmaeker, Y. Cao, G. K. Ndonda, and R. Sadre, "A Survey of Public IoT Datasets for Network Security Research," *IEEE Communications Surveys and Tutorials*, vol. 25, no. 3, pp. 1808–1840, 2023, doi: 10.1109/COMST.2023.3288942.
[13] A. Fatani, M. A. Elaziz, A. Dahou, M. A. A. Al-Qaness, and S. Lu, "IoT Intrusion Detection System Using Deep Learning and Enhanced Transient Search Optimization," *IEEE Access*, vol. 9, pp. 123448–123464, 2021, doi: 10.1109/ACCESS.2021.3109081.
[14] "IoT Dataset 2023 | Datasets | Research | Canadian Institute for Cybersecurity | UNB." Accessed: Sep. 22, 2024. [Online]. Available: https://www.unb.ca/cic/datasets/iotdataset-2023.html
[15] M. Naeem *et al.*, "Trends and Future Perspective Challenges in Big Data," *Smart Innovation, Systems and Technologies*, vol. 253, pp. 309–325, 2022, doi: 10.1007/978-981-16-5036-9_30.
[16] N. Sahllal and E. M. Souidi, "A Comparative Analysis of Sampling Techniques for Click-Through Rate Prediction in Native Advertising," *IEEE Access*, vol. 11, pp. 24511–24526, 2023, doi: 10.1109/ACCESS.2023.3255983.
[17] Z. Wang, H. Chen, S. Yang, X. Luo, D. Li, and J. Wang, "A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization," *PeerJ Comput Sci*, vol. 9, p. e1569, Sep. 2023, doi: 10.7717/PEERJ-CS.1569/SUPP-1.
[18] S. Gavel, A. S. Raghuvanshi, and S. Tiwari, "Distributed intrusion detection scheme using dual-axis dimensionality reduction for Internet of things (IoT)," *Journal of Supercomputing*, vol. 77, no. 9, pp. 10488–10511, Sep. 2021, doi: 10.1007/S11227-021-03697-5/TABLES/10.
[19] M. Tomaszewski and J. Osuchowski, "Effectiveness of Data Resampling in Mitigating Class Imbalance for Object Detection," 2020.
[20] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *J Big Data*, vol. 8, no. 1, p. 6, Dec. 2021, doi: 10.1186/s40537-020-00390-x.
[21] S. Jain, S. Bihani, S. Jaiswal, and A. Arora, "Impact of Feature Selection Algorithms on Network Intrusion Detection," *IEEE Symposium on Wireless Technology and Applications, ISWTA*, vol. 2023-August, pp. 66–71, 2023, doi: 10.1109/ISWTA58588.2023.10250134.
[22] J. Azimjonov and T. Kim, "Designing accurate lightweight intrusion detection systems for IoT networks using fine-tuned linear SVM and feature selectors," *Comput Secur*, vol. 137, p. 103598, Feb. 2024, doi: 10.1016/J.COSE.2023.103598.
[23] V. Senthilkumar and B. V. Kumar, "A Survey on Feature Selection Method for Product Review," *2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation, ICAECA 2021*, 2021, doi: 10.1109/ICAECA52838.2021.9675726.
[24] "1.1. Linear Models — scikit-learn 1.5.2 documentation." Accessed: Sep. 22, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/linear_model.html#lasso
[25] "6.3. Preprocessing data — scikit-learn 1.5.2 documentation." Accessed: Sep. 22, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-scaler
[26] "1.10. Decision Trees — scikit-learn 1.5.2 documentation." Accessed: Sep. 22, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html#decision-trees
[27] J. Yang and H. Wang, "Interpretability Analysis of Academic Achievement Prediction Based on Machine Learning," *Proceedings - 11th International Conference on Information Technology in Medicine and Education, ITME 2021*, pp. 475–479, 2021, doi: 10.1109/ITME53901.2021.00101.
[28] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment," *Sensors 2023, Vol. 23, Page 5941*, vol. 23, no. 13, p. 5941, Jun. 2023, doi: 10.3390/S23135941.

[29] N. D. Primadya, A. Nugraha, A. Luthfiarta, and S. Y. Fahrezi, “Optimasi Logistic Regression untuk Deteksi Serangan DoS pada Keamanan IoT,” *Jurnal Eksplora Informatika*, vol. 13, no. 2, pp. 245–252, Mar. 2024, doi: 10.30864/EKSPLORA.V13I2.1065.

[30] A. I. Jony and A. K. B. Arnob, “A long short-term memory based approach for detecting cyber attacks in IoT using CIC-IoT2023 dataset,” *Journal of Edge Computing*, vol. 3, no. 1, pp. 28–42, May 2024, doi: 10.55056/jec.648.

[31] D. Setiawan, A. Nugraha, and A. Luthfiarta, “Komparasi Teknik Feature Selection Dalam Klasifikasi Serangan IoT Menggunakan Algoritma Decision Tree,” *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 8, no. 1, pp. 83–93, Jan. 2024, doi: 10.30865/mib.v8i1.6987.