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

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

Enhancing Malware Detection in IoT Networks using Ensemble Learning on IoT-23 Dataset

Kurnia Anggriani*[1], Syakira Az Zahra[2], Agus Susanto[3]

[1,2,3]Informatics, Faculty of Engineering University of Bengkulu, Indonesia

Email: [1]kurnia.anggriani@unib.ac.id

Received : May 27, 2025; Revised : Jul 7, 2025; Accepted : Jul 7, 2025; Published : Aug 18, 2025

Abstract

The Internet of Things (IoT) has become a technological innovation that brings many benefits in various sectors, but also presents challenges, especially in terms of cybersecurity. One of the main threats is malware, which can damage devices, steal data, and disrupt system performance. With the increasing use of IoT, malware attacks on IoT devices are a serious concern. Previous research shows that malware detection models in IoT devices still have shortcomings, especially in terms of accuracy. One of the algorithms used in malware detection, Naïve Bayes, has been shown to provide low accuracy results. This study aims to improve the accuracy of malware detection on IoT networks by applying Ensemble learning techniques using traffic data from the IoT-23 dataset. The methodology used refers to the CRISP-DM (Cross Industry Standard Process for Data Mining) framework, which includes the stages of domain understanding, data understanding, data preparation, modelling, evaluation, and deployment. The results show that Ensemble learning improved the performance of individual models. Naïve Bayes as a single model produces an accuracy of 0.24, increasing to 0.35 when combined with AdaBoost, and 0.99 when combined with XGBoost. The combination of the three models also produced an accuracy of 0.99. These results demonstrate the effectiveness of ensemble learning in improving malware detection accuracy in IoT environments.

Keywords : *Ensemble Learning, IoT-23, Malware.*

1. INTRODUCTION

The Internet of Things (IoT) has become one of the most influential technological innovations in the last few decades. With the ability to connect various physical devices to the internet network, IoT offers many benefits in various sectors, from the manufacturing industry to households. However, this innovation also comes with significant challenges, one of which is cybersecurity threats, especially in the form of malware. Malware is a record or code that is passed over a network to infect, explore, steal, or play any action the attacker likes [1]. During the first half of 2022, Kaspersky, a leading antivirus company, detected and stopped 79,442 malware attacks targeting mobile devices in Indonesia. In addition, during the first six months of 2022, Indonesia ranked fourth in the world in terms of threats to mobile devices [2]. Malware threats are not only limited to mobile devices, but also include critical infrastructure that is the backbone of a country's operations [3]. Malware detection can be done through static and dynamic analysis, which complement each other. Static analysis examines files without executing code, using tools such as disassemblers and decompiles to quickly identify internal structures, but is weak against obfuscation techniques such as code obfuscation [3] .

Previous research that detected malware on IoT networks was conducted by Stoian (2020) who detected malware on IoT-23 networks using the Random Forest, Naïve Bayes, Multi Layers Perceptron, Support Vector Machine, and AdaBoost algorithms. The final result of this study is that the Random Forest algorithm has the highest accuracy rate of 99.5%. The Random Forest algorithm has good performance on large amounts of data but has the disadvantage of being a model, namely it is less

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

efficient for very rare data (many zero features). This study also found that the accuracy of Naïve Bayes has low performance in the accuracy of the F-1 score matrix, which is 0.23 out of 1.00 [4]. In a study by Pristyanto (2019) who conducted Ensemble learning between Naïve Bayes and AdaBoost to classify students' knowledge status on the DC Electrical Machines course and produced an accuracy of 91.98% [5]. Therefore, this study aims to see whether the accuracy of low-performance algorithms such as Naïve Bayes can be improved by combining it with other algorithms in detecting malware on IoT networks.

A. Malware

Malware is a malicious software designed to destroy computer systems and programs. Malware has many forms such as viruses, worms, Trojans, and spyware. Every year, many computer systems around the world are damaged by malware. [6]. Cybercriminals develop malware to steal data, bypass access controls, and gain access to personal computers or damage the target computer, its data, or applications. Today the malware industry has become very profitable, attracting more efforts from cybercriminals and causing exponential growth in the number, type, and complexity of malware created. In addition, generic anti-virus software alone cannot detect malware mutations and variants that make users and systems vulnerable at any given time[7]. Malware is created with the specific purpose of carrying out malicious activities that can cause great losses to its victims, such as eavesdropping, theft of personal information, to system destruction carried out by intruders on victim devices with various motives[8].

B. IoT-23

The IoT-23 dataset, a valuable resource for this study, offers a collection of network traffic captures from IoT devices. Developed by Sebastian Garcia, Agustin Parmisano, and Maria Jose Erquiaga in collaboration with Avast. Software-wise, this dataset contains examples of both good and malicious IoT network traffic in the real world [9]. The dataset consists of 20 malware samples and 3 benign samples. The three benign samples are from real IoT devices, namely Somfy door lock, Philips Hue, and Amazon Echo. Meanwhile, 20 malware samples were collected using Raspberry Pi. After the .pcap files were generated in the IoT-23 dataset, they were processed through Zeek Network Analyzer to generate log files. Zeek has the ability to generate connection log files that show the characteristics of the data flow between two entities. The .pcap files were manually analyzed to identify various property labels. Next, a Python script was run on the log files to add labels based on the analysis results. The size of the malware files varies from a few kilobytes to around 10 gigabytes, so the unit of analysis used was the flow [10].

C. Ensemble Learning

Ensemble learning is a machine learning paradigm where multiple models (often called "weak learners") are trained to solve the same problem and combined to get better results. The main hypothesis is that when weak models are combined, we can get a more accurate model[11]. The Ensemble method is a combination of several single classifiers to form a new classifier so that more accurate results are obtained. The Ensemble method combines several single classifiers with the aim of combining the advantages of each classifier so that the performance obtained in solving problems becomes better [5].

D. Naïve Bayes

The Naïve Bayes Classifier algorithm is an algorithm that is often used to classify data using calculations on probability [12]. Naïve Bayes is a simple probabilistic classification method, which calculates probability based on the frequency and combination of values from the available dataset. This algorithm uses Bayes' theorem and assumes that each attribute is independent or does not depend on the

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

class variable [13]. There are three types of Naïve Bayes Classifier that can be applied in machine learning training, namely Mutinomial Naïve Bayes, Bernoulli Naïve Bayes, and Gaussian Naïve Bayes. In several studies on malware detection using Naïve Bayes, the resulting accuracy tends to be smaller than other accuracies.

E. AdaBoost

AdaBoost is a classification method that works by combining several weak classifiers (which perform slightly better than guessing) on a dataset, then adding additional classifiers by giving more weight to cases that were misclassified by the previous classifiers. Therefore, this algorithm is more effective for datasets that are difficult to classify. The efficiency of this algorithm lies in its ability to only iterate on cases that have not been successfully classified correctly. [4]. The advantage of the AdaBoost algorithm is that it optimizes the Naïve Bayes algorithm as an estimator or weak learner algorithm so that it produces better accuracy [14].

F. XGBoost

XGBoost is an algorithm that has recently become a hot topic in applied machine learning and Kaggle competitions for both structured and unstructured data. XGBoost is an implementation of decision trees enhanced with gradient boosting techniques, designed for high speed and performance. XGBoost also has built-in capabilities to handle Missing Values. [14].

G. Related Research

Research conducted by Azmee et al., (2020) namely Analysis of Machine Learning Performance in detecting malware on PE. Researchers used Logistic Regression, K-Nearest Neighbor (k-NN), Random Forest, AdaBoost, Support Vector Machines (SVM), Decision Trees, XGBoost, Artificial Neural Network (ANN) and Extra Tree Classifier. The best training results were assessed on XGBoost at 98.6% [15].

Furthermore, research from Chiwariro & Pullagura (2023), this study aims to develop three machine learning models to detect and classify malicious software and compare the efficiency of the three models. Dataset A dataset consisting of data with 83 columns or features about malware and its types was collected from Kaggle, the results showed that the LightGBM algorithm reached 93.3%, followed by the XGBoost algorithm at 89.3% and finally, Logistic Regression at 84% [1].

This research by Tjahjadi & Santoso (2023) aims to detect malware using machine learning techniques. The dataset taken in this study was obtained from the internet, namely on the Website https://www.kaggle.com/datasets/amauricio/pe-files-malwares. Consisting of 19611 rows and 79 columns. The algorithm used in this study is Random Forest. The Random Forest model shows very good performance without requiring preprocessing on the data. Even though the data is unbalanced, the results are still optimal, and no techniques are needed to balance the data. Scaling is also not necessary, because Random Forest works by dividing data based on feature values, not performing direct calculations on its values. The results of the study revealed that this model achieved a precision level of 0.99 for the "Not Malware" and "Malware" classes, with a recall of 0.96 for "Not Malware" and 1.00 for "Malware". In addition, the f1-score obtained was 0.98 for "Not Malware" and 0.99 for "Malware" [16].

Further research conducted by Ari Sandriana et al., (2022) aims to classify malware attacks on IoT network traffic using the K-Nearest Neighbor (KNN) algorithm. The model was trained using 20 selected traffic anomaly datasets, the data was divided into 60% for training data and 40%. The results of the Classification or model training were carried out using the K-Nearest Neighbor (K-NN) algorithm based on predetermined training data and testing data, resulting in an accuracy value of 94% [17]. Unlike

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

previous works, this study explores the impact of ensembling weak classifiers with stronger learners to boost detection performance on the IoT-23 dataset."

2. METHOD

This research uses CRISP-DM, This study is a study that aims to show the performance of Ensemble learning techniques on the Naïve Bayes, XGBoost, and AdaBoost Algorithm models in detecting malware on IoT networks in the IoT-23 dataset. This study was chosen because it allows the author to conduct controlled testing of certain variables and analyze their influence on the results. The data in this study are secondary data, namely data collected by others. The data set consisting of data with 21 columns or features about malware and its types was collected from Kaggle. Kaggle is a data science and artificial intelligence platform where cash prize contests are published by large companies and organizations.

This research was designed using the CRISP-DM (Cross Industry Standard Process for Data Mining) method, a method for processing data mining that has been developed where the existing data will go through each structured and defined phase clearly and efficiently[18]. CRISP-DM offers a structured and comprehensive approach, making it suitable for use in various industrial sectors. This approach is also in managing and analyzing data, so that it can produce useful insights to support appropriate and strategic decision making. [19]. The following is the process in this research.:

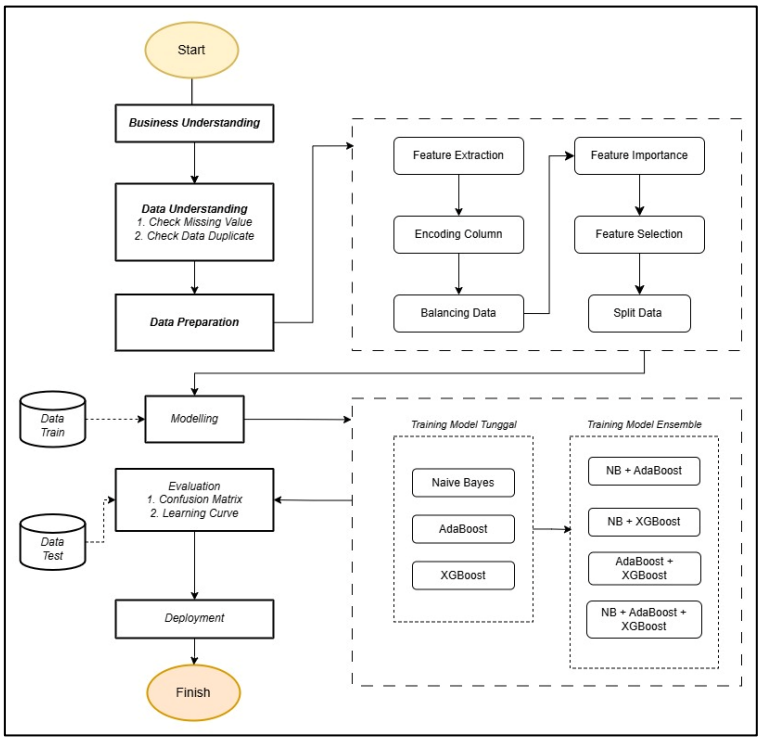

Figure 1. CRISP-DM Process in research

The first stage in the CRISP-DM method used in this study is Business Understanding. This stage aims to understand the background, needs, and objectives of the research in the context of business or real-world problems to be solved. The next step is understanding the data to be processed. At this stage, the IoT-23 dataset preparation process is carried out. The dataset is downloaded in .csv format, the dataset only contains one table with 21 columns totaling 1,048,576 rows. There are 8 label classes consisting of 7 malware classes and 1 benign class.

- Check Missing Value
- Check Duplicate Data

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

After that, data preparation. At this stage, it is like preparing and measuring the ingredients that will be cooked directly. In other words, these ingredients are datasets. The IoT-23 dataset will be explored, cleaned, and prepared as a new dataset that will be used in the Modeling process

- Feature Extraction
- Encoding Column
- Balancing Data
- Feature Importance
- Feature Selection
- Split Data

After the dataset has been divided into training data and test data. next is Modeling At this stage, the models used are Naïve Bayes, AdaBoost, XGBoost, and Ensemble of the three models. There are seven stages of modeling, namely: (1) Modeling with Naïve Bayes (2) Modeling with AdaBoost (3) Modeling with XGBoost (4) Modeling with Naïve Bayes+AdaBoost+XGBoost (5) Modeling with Naïve Bayes+AdaBoost (6) Modeling with Naïve Bayes+XGBoost (7) Modeling with AdaBoost and XGBoost. These models will be trained using training data.

Next is the evaluation stage. The trained model is tested using test data from a separate dataset to measure how well the model performs. This involves evaluating metrics such as accuracy, precision, recall, and F1 score. Implementation

After the model is evaluated, the best performing model is saved for use in production. This can be done by saving the model to a specific file format (e.g., .pkl for Python) that is integrated into the Malware Detection Website. This website is designed to help users analyze IoT networks suspected of containing malware. The system will use machine learning techniques to detect potential threats.

3. RESULT

3.1. Business Understanding

The main problem is the increasing number of malware attacks on the Internet of Things (IoT) network, which causes losses such as data leakage and potential damage to the devices used. With the development of IoT devices, security threats to the network have also increased significantly. Many IoT devices are not equipped with adequate security systems, making them vulnerable to malware attacks that can lead to data theft, device damage, or unauthorized access to the network.

3.2. Data Understanding

This study utilizes the IoT-23 dataset as the primary data source. The dataset used is a preprocessed version available open source through the Kaggle platform (kaggle.com/datasets/engraqeel/iot23preprocesseddata), developed by Aqeel Ahmed. This dataset is a combination of IoT-23 developed by Sebastian Garcia, Agustin Parmisano, and Maria Jose Erquiaga, which contains both malicious and non-malicious IoT network traffic data. At the Data Understanding stage, the aim is to understand the patterns and structures of the data [20], the first step taken is to see the overview of the IoT-23 dataset. This dataset consists of 21 columns and 1,048,575 rows. However, the IoT-23 dataset is not yet fully ready to be used to train the model, there is still data that cannot be accepted by the model, such as non-numeric data. Furthermore, an analysis of each column is also carried out to understand its meaning. Data analysis is important in order to determine which columns are relevant to use in the modeling process. The following is a table containing the columns in the IoT-23 Dataset Table:

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

Table 1. Iot-23 Dataset Column Explanation

No	Column	Description
1	ts	First package time.
2	uid	Unique identifier for each connection in the dataset.
3	id.orig_h	The source IP address (the host that initiated the connection).
4	id_orig_p	The source port number of the device initiating the connection.
5	id_resp_h	Destination IP address (host accepting the connection).
6	id_resp_p	The destination port number of the device receiving the connection.
7	proto	The network protocol used in the connection.
8	service	Network services or applications used
9	duration	How long the connection takes
10	orig_bytes	The number of bytes of payload sent by the originator.
11	resp-bytes	The number of bytes of payload sent by the respondent.
12	conn_state	Network connection status, showing how connections are opened and closed.
13	local_orig	If the connection originates locally, this value will be T.
14	local_resp	If the connection is responded to locally, this value will be T.
15	missed_bytes	The number of bytes missed in a message
16	history	Records connection status history as a string of letters.
17	orig_pkts	Number of packets sent to the device
18	orig_ip_bytes	The number of IP level bytes sent by the device
19	resp_pkts	Number of packages sent by respondents.
20	resp_ip_bytes	The number of IP level bytes sent by the responder.
21	label	Types of malware

Then data exploration is carried out by looking at Missing Values and duplicate data in the IoT-23 dataset. If there are any, they must be cleaned at the data preparation stage.

3.2.1. Check Missing Value

Missing Value is one of the important things in the data preparation stage. Missing Value/data is not only incomplete data, but can also appear in the form of outliers or values that do not match the previous value, or unreasonable filling [21] . Missing values contained in the data can damage the results of the modeling process. Data that receives missing value handling has better results than incomplete data [22].There is no Missing Value in all features and targets in the IoT-23 Dataset, which means that every data in the feature and target columns is filled in completely. This facilitates the pre-processing process because no imputation or data deletion techniques are required.

3.2.2. Check Duplicate Data

This process identifies and handles data that appears more than once in a dataset. Data duplication can lead to operational inefficiencies, waste of resources, errors in data analysis, and inaccurate decision making. In addition, duplication can increase storage costs and slow down the company's work system [23]. This process is important in data management, especially to ensure the accuracy and integrity of data in the model. In the IoT-23 dataset, no more than one identical data or duplicate data was found.

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

3.3. Data Preparation

3.3.1. Feature Extraction

Feature Extraction serves to simplify data without losing important information, so that machine learning algorithms can work more effectively. In the IoT-23 dataset, there are several types of similar labels, namely C&C-HeartBeat, C&C-Torii, and C&C-FileDownload. These three types of malware are part of the C&C malware. So by utilizing the extraction feature, these three types of malware will be equated into C&C files.

3.3.2. Encoding Column

Label Encoding is a method that converts categorical data into a numeric format [8]. In the IoT-23 dataset, there are 14 out of 21 columns that are non-numeric/object data types, namely ts, uid, id.orig_h, id.resp_h, service, proto, duration. The encoding performed on the IoT-23 dataset is to initialize non-numeric data into numeric.

3.3.3. Balancing Data

In the IoT-23 dataset, there is unbalanced data in each class. Data with unbalanced classes can cause classification performance to be less than optimal, because the model tends to focus more on the majority class in its classification process[24]. This imbalance is overcome by using the under sampling technique, which is to reduce some of the data in the majority class from the data. In this study, the author balanced the data by making all classes the same number with the smallest total class, namely 6883.

3.3.4. Fetuare Importance

Feature Importance is used to see the level of importance of a feature to the target/label results. This feature can also be used as a consideration to determine what features will be used in the Modeling process later. In its use, it also utilizes the Library owned by the Random Forest algorithm.

3.3.5. Feature Selection

This stage is used to select which features will be used in the Modeling process. Feature Selection is used in various applications to filter irrelevant or redundant features in the model used so that the model used can be optimal [25]. The Feature Selection method has a significant influence in the case of keyword extraction [26]. The features contained in the dataset totaling 21 features are ts, uid, id.orig_h, id_orig_p, id_resp_h, id_resp_p, proto, service, duration, orig_bytes, resp-bytes, conn_state, local_orig, local_resp, missed_bytes, history, orig_pkts, orig_ip_bytes, resp_pkts, and resp_ip_bytes. Feature selection by SelectFromModel with RandomForestClassifier is based on the importance score given by the model to each feature. The selected features are ['ts', 'id.orig_h', 'id.orig_p', 'id.resp_h', 'id.resp_p', 'duration, 'history', 'orig_pkts', 'orig_ip_bytes'] or 9 out of 21 features.

3.3.6. Split Data

At this stage is the final stage of dataset processing before finally entering the Modeling process. The dataset that previously amounted to 34,415 will be divided into 60% or 20,649 for training data and 40% or 13,766 for test data.

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

3.4. Modelling

3.4.1. Naïve Bayes

The function created to train a Naïve Bayes-based classification model using the Gaussian Naïve Bayes algorithm. The parameters used in this modeling process are as follows:

- priors: None
- var_smoothing: 1e-09

The parameters that are part of Gaussian Naïve Bayes have the following explanations, priors: None means the model does not use the specified prior probabilities, so the model will automatically calculate the initial probability (prior) of each class based on the distribution of training data. While var_smoothing: 1e-09 to stabilize the probability calculation means preventing numerical errors by adding small values to the variance

3.4.2. AdaBoost

The parameters used in this modeling process are as follows:

- algorithm: SAMME.R
- base_estimator: deprecated
- estimator: None
- learning_rate: 1.0
- n_estimators: 100
- random_state: 42
- algorithm: SAMME.R

The algorithm parameter: SAMME.R is used to specify the boosting variant used, where this method uses class probabilities in weighting the new model, rather than just discrete predictions like SAMME. This method is generally faster and more accurate, especially for data with more than two classes. The base_estimator parameter has been deprecated in recent versions of scikit-learn and has been replaced by estimator, which defaults to None. Next, learning_rate: 1.0 controls how much each weak learner contributes to the final model. Smaller values can help avoid overfitting, while larger values make the model more aggressive in learning. The n_estimators: 100 parameter specifies the number of weak learners used in boosting, where more estimators can increase the complexity of the model but also risk causing overfitting. Finally, random_state: 42 ensures that the model training results are reproducible with consistent results every time the model is run.

3.4.3. XGBoost

In the training process with XGBoost, the author uses the Random Search technique which is used for hyperparameter optimization. This is done to improve the model's performance and avoid overfitting or underfitting.

- objective: multi:Softmax
- eval_metric: mlogloss
- learning_rate: None
- n_estimators: None
- max_depth: None
- min_child_weight: None
- gamma: None
- subsample: None
- colsample_bytree: None

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

- reg_alpha: None, reg_lambda: None
- tree_method: None
- nthread: -1
- seed: 42

The objective: multi:Softmax parameter is used to perform multiclass classification with the Softmax activation function, which generates probabilities for each class and selects the class with the highest probability as the final prediction. Model evaluation is performed using eval_metric: mlogloss, which measures the multiclass log loss, a metric often used to assess the performance of multiclass classification models. Several other parameters such as learning_rate, n_estimators, max_depth, min_child_weight, gamma, subsample, colsample_bytree, reg_alpha, reg_lambda, and tree_method are currently set to None, meaning the model will use the default values if not set.

3.4.4. Ensemble

At this stage the author ensembles Naïve Bayes and AdaBoost, Naïve Bayes and XGBoost, AdaBoost and XGBoost, and then all three. By using Soft Voting Classifier. Soft Voting is a method in which the prediction results of several different models are calculated based on the average or probability of prediction, by giving certain weights to each model.

3.5. Evaluation

3.5.1. Naïve bayes

The result of the training experiment using Naïve Bayes was 0.24, which is a low result for a machine learning model. Here is a picture of the Naïve Bayes learning curve:

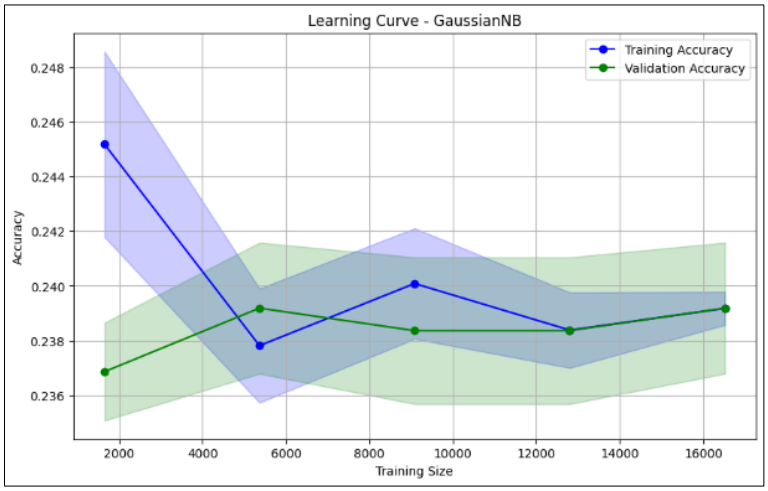

Figure 2. Learning Curve Naïve bayes

This Learning curve graph - GaussianNB shows the performance of the Gaussian Naïve Bayes model against the size of the training data. At the beginning the training accuracy is high, but as the training data increases, the accuracy decreases, indicating that the model can initially memorize small training data, but faces more challenges with larger data. This model tends to underfit because the training and validation accuracy are both low.

3.5.2. AdaBoost

The result of the training experiment using AdaBoost was 0.62. This result is neither low nor high for a machine learning model.

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

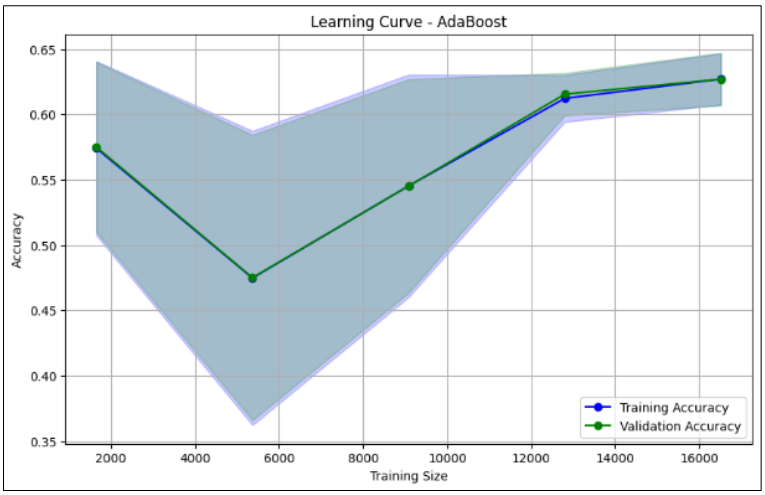

Figure 3. Learning curve AdaBoost

The green line shows lower validation accuracy at the beginning, validation accuracy starts at around 0.55, dropping to 0.45 when the data reaches 6000. The large variation at the beginning due to small data, narrows towards the end, indicating a more consistent evaluation.

3.5.3. XGBoost

The results of the training experiment using XGBoost showed a result of 1.00 where for a machine learning model, this number is a perfect result. The following is a graph of the learning curve during the modeling process:

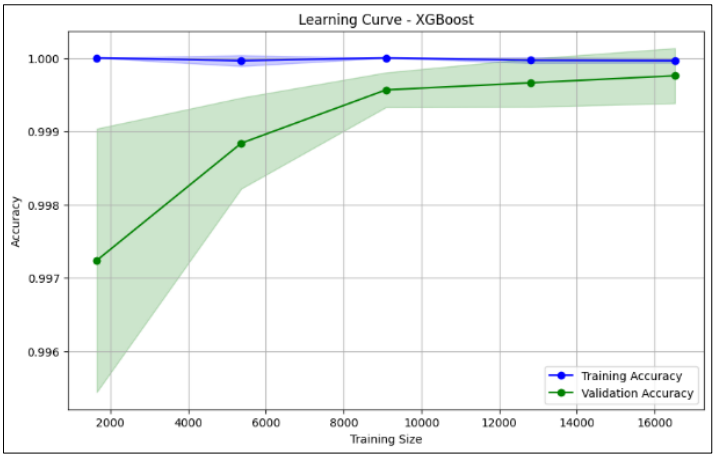

Figure 4. Learning curve XGBoost

The Learning curve graph in the figure shows the performance of the XGBoost (Extreme Gradient Boosting) model on the training and validation data as the amount of training data increases. The XGBoost model performs very well on this dataset. There is no indication of underfitting and no indication of overfitting.

3.5.4. Ensemble Naïve Bayes + AdaBoost

The results of the training experiment using Ensemble between Naive Bayes and AdaBoost with the Soft Voting technique showed a result of 0.35. The following is a graph of the learning curve during the modeling process:

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

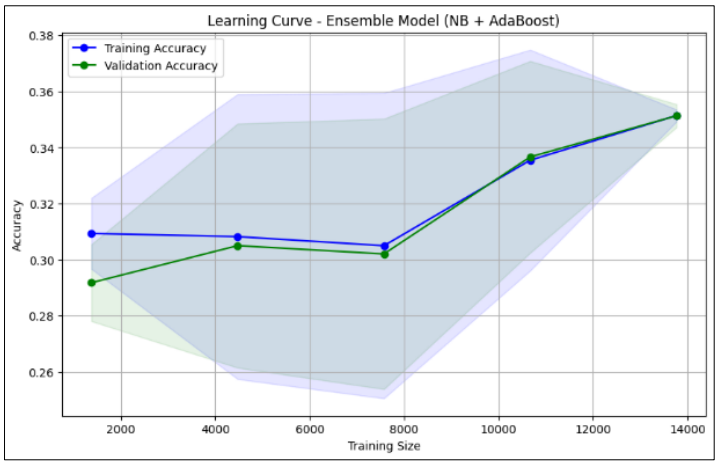

Figure 5. Learning curve ensemble Naïve bayes + AdaBoost

The learning curve graph in the figure shows the performance of the Ensemble Naïve Bayes and AdaBoost models, the blue line shows the performance of the model on the training data. This line shows that the training accuracy is stable and increases with the increase in the size of the training data.

3.5.5. Ensemble Naïve Bayes + XGBoost

The results of the training experiment using Ensemble between Naive Bayes and XGBoost with the Soft Voting technique showed a result of 0.99. The following is a graph of the learning curve during the modeling process:

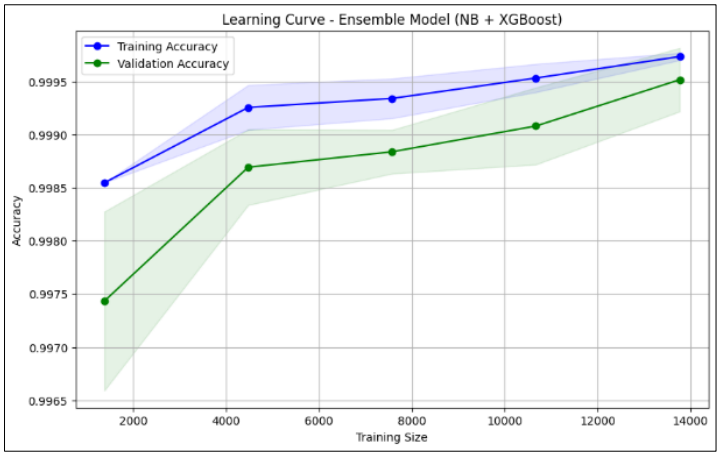

Figure 6. Learning curve ensemble Naïve bayes + XGBoost

The accuracy in this graph is 0.9993. The training curve illustrates the performance of the model on the training data. The accuracy of the model on the training data is very high from the beginning (>0.9990) and remains stable until the end. The combination of Naïve Bayes and XGBoost produces a model with very good performance, with almost perfect accuracy.

3.5.6. Ensemble AdaBoost + XGBoost

The results of the training experiment using Ensemble between AdaBoost and XGBoost is 0.99. The following is a picture of the Ensemble learning curve between AdaBoost and XGBoost:

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

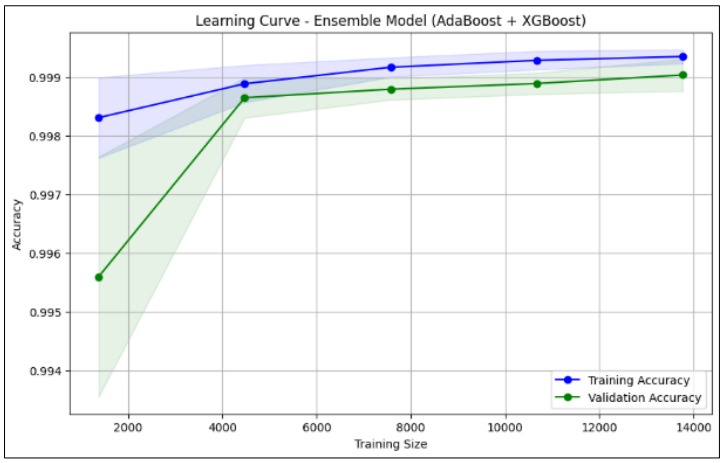

Figure 7. Learning curve ensemble AdaBoost + XGBoost

This figure shows the learning curve of the Ensemble model combining AdaBoost and XGBoost, with an accuracy of 0.9988. The training curve is stable from the beginning, indicating that the model is able to recognize patterns in the training data very well. The validation curve increases as the data increases and approaches the training accuracy, indicating good generalization ability.

3.5.7. Ensemble Naïve Bayes + AdaBoost +XGBoost

The results of the training experiment using Ensemble between Naïve Bayes and AdaBoost with the Soft Voting technique showed a result of 0.99. The following is a graph of the learning curve during the modeling process:

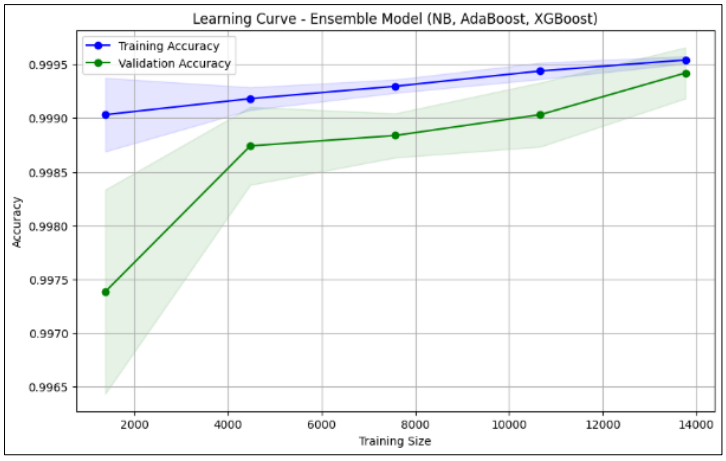

Figure 8. Learning curve ensemble Naïve bayes + AdaBoost + XGBoost

This figure shows the learning curve for an Ensemble model combining Naïve Bayes, AdaBoost, and XGBoost. The accuracy in this graph is around 0.9991 for the training data and increases to near that value on the validation data. The combination of Naïve Bayes, AdaBoost, and XGBoost produces a model with very good performance.

3.6. Deployment

At this stage, the model from the training results will be saved in .pkl format, then the model will be integrated into the malware detection website. The following is a display of the malware detection website interface that has been deployed:

Jurnal Teknik Informatika (JUTIF)
-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

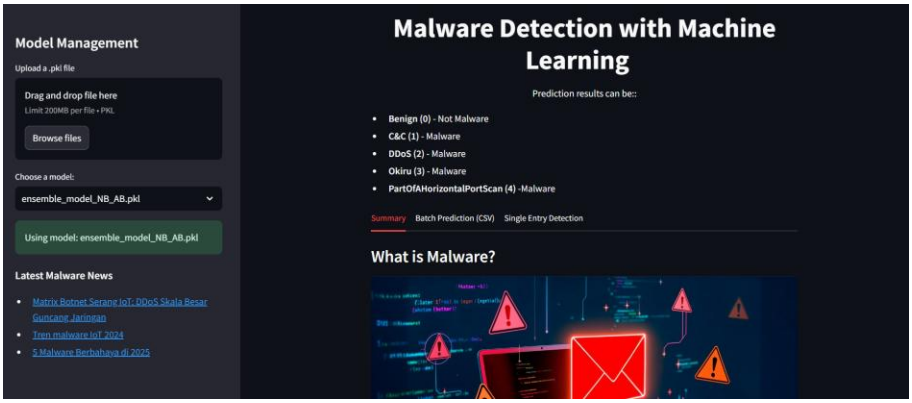

Figure 9. Malware detection website interface

Below is an example that will be detected whether it is malware or not.

ts	id.orig_h	id.orig_p	id.resp_h	id.resp_p	duration	history	orig_pkts	orig_ip_bytes
1545337568064110.0	100.0	43746.0	326094.0	8080.0	0.0	14.0	1.0	40.0
1545413030765110.0	98.0	51432.0	337298.0	6667.0	4442.0	14.0	3.0	180.0
1536227356883340.0	104.0	18344.0	371638.0	52869.0	2943.0	14.0	2.0	80.0

Figure 10. Data before malware detection

Then this is the test data file that has gone through the detection process by the system, the data will add one column named "Prediction" where this column will determine whether the network is Benign (0), C&C (1), DdoS (2), Okiru (3), and PartOfAHorizontalPortScan (4).

ts	id.orig_h	id.orig_p	id.resp_h	id.resp_p	duration	history	orig_pkts	orig_ip_bytes	Prediction
1545337568064110.0	100.0	43746.0	326094.0	8080.0	0.0	14.0	1.0	40.0	0
1545413030765110.0	98.0	51432.0	337298.0	6667.0	4442.0	14.0	3.0	180.0	1
1536227356883340.0	104.0	18344.0	371638.0	52869.0	2943.0	14.0	2.0	80.0	2

Figure 11. Data after malware detection

This study shows that Ensemble learning can improve the performance of low individual models. When compared to previous studies, the study can provide updates in malware detection using machine learning. The table below is a summary of the accuracy of each model trained with the IoT-23 dataset.

Table 2. Accuracy of Individual and Ensemble Models on IoT-23 Dataset

Model	Accuracy	Difference
Naïve Bayes	0.24	Fast with a time of 0.0289 seconds, but low accuracy
AdaBoost	0.62	Improves the accuracy of Naive Bayes, but still less than optimal
XGBoost	0.99	Very accurate, but computationally longer 431.1249 seconds
Naïve Bayes + AdaBoost	0.35	Poor performance because Naive Bayes still dominates
Naïve Bayes + XGBoost	0.99	Performance remains high thanks to XGBoost
AdaBoost + XGBoost	0.99	Combining boosting power, stable results
Naïve Bayes + AdaBoost + XGBoost	0.99	Optimal combination: leveraging the speed of Naïve Bayes, the boosting power of AdaBoost, and the high accuracy of XGBoost

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

4. DISCUSSIONS

This study shows that Ensemble learning can improve the performance of low individual models. When compared to previous studies, the study can provide updates in malware detection using machine learning.

Table 3. Comparison with previous research

Paper Title	Year	Dataset	Algorithm
Machine Learning for Anomaly Detection in IoT networks: Malware analysis on the IoT-23 Data set [4]	2020	IoT-23	*Random Forest* = 99.5%
Guardians of IoT : Malware Analysis of IoT Devices Using Machine Learning [9]	2024	IoT-23	*Random Forest* = 99%
An Efficient Android Malware Prediction Using Ensemble machine learning algorithms [27]	2021	Drebin Dataset	*LightGBM* = 99.5%
Performance analysis of machine learning classifiers for detecting PE Malware [15]	2020	PE	*XGBoost* = 98,6%
ENHANCING MALWARE DETECTION IN IoT NETWORKS USING ENSEMBLE LEARNING: A STUDY ON THE IoT-23 DATASET	2025	IoT-23	*Ensemble Naïve Bayes+ AdaBoost + XGBoost* = 99%

Overall, the Naïve Bayes model improved when ensembled with AdaBoost and XGBoost models. The AdaBoost model decreased when ensembled with Naïve Bayes, but increased when ensembled with XGBoost. This may be due to one of the disadvantages of AdaBoost being susceptible to overfitting, especially if the dataset contains a lot of noise. Furthermore, the results of this study indicate that the Ensemble learning method can improve the individual Naïve Bayes model in learning and is also able to reduce the performance of the AdaBoost model. The XGBoost model plays an important role because it can help individual models achieve very good accuracy. With the right combination of parameters, XGBoost is able to maximize the performance of individual models, making it a strong choice in various multi-class classification tasks. The findings contribute to the development of intelligent network monitoring systems for IoT security based on ensemble machine learning techniques

5. CONCLUSIONS

This study successfully answered the problem posed, namely how the application of ensemble learning can improve model performance in detecting malware on the IoT-23 network. The process begins with the data preprocessing stage which includes cleaning from missing values and duplicate data, as well as selecting relevant features. The cleaned data is then divided into training data and test data, where the training data is used for the model training process. The application of ensemble learning has been proven to be able to improve the performance of individual models that previously had low accuracy. For example, Naïve Bayes individually only produces an accuracy of 0.24, but increases to 0.35 when ensembled with AdaBoost and jumps to 0.99 when ensembled with XGBoost. The AdaBoost model itself obtains an accuracy of 0.62 individually, decreases when combined with Naïve Bayes to 0.35, but increases to 0.99 when combined with XGBoost. Meanwhile, XGBoost consistently shows high accuracy of 0.99, both individually and when combined with other models. The ensemble of the

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

three models gave a final accuracy of 0.99. In addition to the selection of the ensemble method, data quality also plays an important role in the success of the model. Further research could integrate additional datasets, perform real-time detection experiments, or apply feature reduction techniques to improve efficiency."

REFERENCES

[1] R. Chiwariro and L. Pullagura, "Malware Detection and Classification Using Machine Learning Algorithms," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 11, no. 8, pp. 1727–1738, 2023, doi: 10.22214/ijraset.2023.55255.

[2] Sharipuddin, R. S. Putra, M. F. Aulia, S. A. Maulana, and P. A. Jusia, "Android Security: Malware Detection with Convolutional Neural Network and Feature Analysis," *Media J. Gen. Comput. Sci.*, vol. 1, no. 1, pp. 7–13, 2023, doi: 10.62205/mjgcs.v1i1.7.

[3] D. Arianyah and I. V. Paputungan, "Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi," *J. Sains, Nalar, dan Apl. Teknol. Inf.*, vol. 3, no. 2, pp. 50–57, 2024, doi: 10.20885/snati.v4.i1.1.

[4] N. A. Stoian, "Machine Learning for Anomaly Detection in IoT networks: Malware analysis on the IoT-23 Data set," *Univ. Twente*, 2020.

[5] Y. Pristyanto, "Penerapan Metode Ensemble Untuk Meningkatkan Kinerja Algoritme Klasifikasi Pada Imbalanced Dataset," *J. Teknoinfo*, vol. 13, no. 1, p. 11, 2019, doi: 10.33365/jti.v13i1.184.

[6] M. A. Hama Saeed, "Malware in Computer Systems: Problems and Solutions," *IJID (International J. Informatics Dev.*, vol. 9, no. 1, p. 1, 2020, doi: 10.14421/ijid.2020.09101.

[7] M. N. Alenezi, H. Alabdulrazzaq, A. A. Alshaher, and M. M. Alkharang, "Evolution of Malware Threats and Techniques: A Review," *Int. J. Commun. Networks Inf. Secur.*, vol. 12, no. 3, pp. 326–337, 2020, doi: 10.17762/ijcnis.v12i3.4723.

[8] T. A. Cahyanto, V. Wahanggara, and D. Ramadana, "Analisis dan Deteksi Malware Menggunakan Metode Malware Analisis Dinamis dan Malware Analisis Statis," *J. Sist. Teknol. Inf. Indones.*, vol. 2, no. 1, pp. 19–30, 2017.

[9] D. Vijayanand and R. K. Singh, "Guardians of IoT : Malware Analysis of IoT Devices Using Machine Learning," vol. 45, no. 1, pp. 911–924, 2024.

[10] N. Abdalgawad, A. Sajun, Y. Kaddoura, I. A. Zualkernan, and F. Aloul, "Generative Deep Learning to Detect Cyberattacks for the IoT-23 Dataset," *IEEE Access*, vol. 10, pp. 6430–6441, 2022, doi: 10.1109/ACCESS.2021.3140015.

[11] L. M. Cendani and A. Wibowo, "Perbandingan Metode Ensemble Learning pada Klasifikasi Penyakit Diabetes," *J. Masy. Inform.*, vol. 13, no. 1, pp. 33–44, 2022, doi: 10.14710/jmasif.13.1.42912.

[12] N. Chatrina Siregar, R. Ruli, A. Siregar, ; M Yoga, and D. Sudirman, "Implementasi Metode Naive Bayes Classifier (NBC) Pada Komentar Warga Sekolah Mengenai Pelaksanaan Pembelajaran Jarak Jauh (PJJ)," *J. Teknol. Aliansi Perguru. Tinggi BUMN*, vol. 3, no. 1, pp. 102–110, 2020.

[13] E. Martantoh and N. Yanih, "Implementasi Metode Naïve Bayes Untuk Klasifikasi Karakteristik Kepribadiaan Siswa Di Sekolah MTS Darussa'adah Menggunakan Php Mysql," *J. Teknol. Sist. Inf.*, vol. 3, no. 2, pp. 166–175, 2022, doi: 10.35957/jtsi.v3i2.2896.

[14] L. Pebrianti, F. Aulia, H. Nisa, and K. Saputra S, "Implementation of the Adaboost Method to Optimize the Classification of Diabetes Diseases with the Naïve Bayes Algorithm," *J. Sist. dan Teknol. Inf.*, vol. 7, no. 2, pp. 122–127, 2022, [Online]. Available: http://jurnal.unmuhjember.ac.id/index.php/JUSTINDO

[15] A. A. Azmee, P. P. Choudhury, A. Md. Alam, O. Dutta, and M. I. Hossai, "Performance analysis of machine learning classifiers for detecting PE Malware," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 1, pp. 510–517, 2020, doi: 10.14569/ijacsa.2020.0110163.

[16] E. V. Tjahjadi and B. Santoso, "Klasifikasi Malware Menggunakan Teknik Machine Learning," *J. Ilm. Ilmu Komput.*, vol. 2, no. 1, pp. 60–70, 2023.

[17] Ari Sandriana, Rianto, and Firmansyah Maulana, "Klasifikasi serangan Malware terhadap Lalu Lintas Jaringan Internet of Things menggunakan Algoritma K-Nearest Neighbour (K-NN)," *E-JOINT (Electronica Electr. J. Innov. Technol.*, vol. 3, no. 1, pp. 12–22, 2022, doi: 10.35970/e-

Vol. 6, No. 4, Agustus 2025, Page. 1985-2000
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4782

joint.v3i1.1559.
[18] M. Ihsan, R. K. Niswatin, and D. Swanjaya, "Deteksi Ekspresi Wajah Menggunakan Tensorflow," *Joutica*, vol. 6, no. 1, p. 428, 2021, doi: 10.30736/jti.v6i1.554.
[19] H. Asyraf and M. E. Prasetya, "Implementasi Metode CRISP DM dan Algoritma Decision Tree Untuk Strategi Produksi Kerajinan Tangan pada UMKM A," *J. Media Inform. Budidarma*, vol. 8, no. 1, p. 94, 2024, doi: 10.30865/mib.v8i1.7050.
[20] F. Abdusyukur, "Penerapan Algoritma Support Vector Machine (Svm) Untuk Klasifikasi Pencemaran Nama Baik Di Media Sosial Twitter," *Komputa J. Ilm. Komput. dan Inform.*, vol. 12, no. 1, pp. 73–82, 2023, doi: 10.34010/komputa.v12i1.9418.
[21] I. J. Fadillah and C. D. Puspita, "Pemanfaatan Metode Weighted K-Nearest Neighbor Imputation (Weighted Knni) Untuk Mengatasi Missing Data," *Semin. Nas. Off. Stat.*, vol. 2020, no. 1, pp. 511–518, 2021, doi: 10.34123/semnasoffstat.v2020i1.409.
[22] A. S. Arifianto, K. D. Safitri, K. Agustianto, and I. G. Wiryawan, "PENGARUH PREDIKSI MISSING VALUE PADA THE EFFECT OF MISSING VALUE PREDICTION ON," vol. 9, no. 4, pp. 779–786, 2022, doi: 10.25126/jtiik.202294778.
[23] N. Hanifah, M. Irwan, and P. Nasution, "Manajemen Data Yang Efektif : Solusi Untuk Mencegah dan Mengatasi Duplikasi Data Dalam Perusahaan," vol. 3, no. 1, 2025.
[24] Y. E. Ardiningtyas, P. Heruningsih, and P. Rosa, "Analisis Balancing Data Untuk Meningkatkan Akurasi Dalam Klasifikasi," *Pros. Semin. Nas. Apl. Sains Teknol. 2021* , vol. 2021, no. Prosiding SNAST 2021, pp. A24–A28, 2021.
[25] O. Somantri, W. E. Nugroho, and A. R. Supriyono, "Penerapan Feature Selection Pada Algoritma Decision Tree Untuk Menentukan Pola Rekomendasi Dini Konseling Oman," *J. Sist. Komput. dan Inform.*, vol. 4, no. 2, pp. 272–279, 2022, doi: 10.30865/json.v4i2.5345.
[26] H. M. Lumbantobing, R. A. Marcellino, and I. C. Bu'ulolo, "Penerapan Metode Feature Selection pada Algoritma Naïve Bayes dalam Kasus Keyword Extraction," *Citee*, pp. 117–123, 2020.
[27] N. Al Sarah, F. Y. Rifat, M. S. Hossain, and H. S. Narman, "An Efficient Android Malware Prediction Using Ensemble machine learning algorithms," *Procedia Comput. Sci.*, vol. 191, no. 2019, pp. 184–191, 2021, doi: 10.1016/j.procs.2021.07.023.