

Spammer Detection On Computer Networks Using Gaussian Naïve Bayes Classifier And K-Medoids As Acquisition Training Data

OK Muhammad Majid Maulana^{✉1} Rizal Tjut Adek² Zara Yunizar³

¹Department of Informatic, Universitas Malikussaleh, Bukit Indah, Lhokseumawe, 24353, Indonesia, o.k.muhammad.200170055@mhs.unimal.ac.id

²Department of Informatic, Universitas Malikussaleh, Bukit Indah, Lhokseumawe, 24353, Indonesia, rizal@unimal.ac.id

³Department of Informatic, Universitas Malikussaleh, Bukit Indah, Lhokseumawe, 24353, Indonesia, zarayunizar@unimal.ac.id

[✉]Corresponding Author: o.k.muhammad.200170055@mhs.unimal.ac.id | Phone: +6281361471537

Abstract

This research focuses on the implementation of the Gaussian Naïve Bayes algorithm for spammer detection in computer networks, leveraging K-Medoids clustering for training data acquisition. The increasing number of internet users, combined with the challenges of detecting spam activity on a network, has made manual detection ineffective. This study addresses the need for automated spam detection using machine learning algorithms. The Gaussian Naïve Bayes algorithm was chosen for its simplicity and effectiveness in handling continuous data, making it suitable for classifying network traffic as either normal or spammer. To acquire labeled training data, K-Medoids clustering was employed, offering robustness against outliers, which traditional clustering algorithms like K-Means often struggle with. The research involved collecting traffic data from a Mikrotik Routerboard at various intervals, followed by data preprocessing to remove irrelevant or null features. After preprocessing, the data was clustered using K-Medoids into two groups: spammer and normal. The Gaussian Naïve Bayes classifier was then applied to the clustered data, producing a model with high accuracy, precision, recall, and F1-score. Specifically, the model achieved 99.71% accuracy, 100% precision, 99.71% recall, and a 99.85% F1-score, indicating a well-balanced performance in spam detection. The results demonstrate that the Gaussian Naïve Bayes algorithm, combined with K-Medoids clustering, is effective for detecting spammers in computer networks. Future research could explore higher-layer network traffic and broader datasets, utilizing different routers for a more comprehensive evaluation. This approach provides a reliable solution for network administrators seeking to improve network security by detecting and mitigating spam activity.

Keywords: Gaussian Naïve Bayes, Spammer Detection, K-Medoids Clustering, Network Traffic Classification

Introduction

The number of internet users in Indonesia reached 221,563,479 in 2024, out of a total population of 278,696,200 in 2023. [1]. This situation poses a challenge for network administrators because almost every computer, laptop, tablet, or mobile phone is connected to the internet through the same public IP address. A public IP address may get blacklisted due to malware on a user's device, which may unknowingly send spam. The malware can attach itself to computer files, such as executables or trojans, using social engineering techniques to disguise harmful files as safe ones. [2]. To reach their target, spammers continuously send data over the computer network. When the target is a user device located within the same network, the role of the device's defense system becomes crucial in preventing the attack. However, if the target is a server on the internet, the public server's firewall may automatically block the public Internet Protocol (IP) address. This means that all devices under the same public IP address will also be blocked and unable to access the server.

Detecting spam activity on a computer network cannot be done manually due to the large number of public IP users. Moreover, each user does not make just one connection to a single internet service, but many connections simultaneously. Additionally, TCP and UDP connections have a timeout period, where the connection is automatically terminated when the time expires. Therefore, spam detection must employ machine learning.

Machine learning (ML) is a field of study within artificial intelligence that focuses on the development and study of statistical algorithms capable of learning from data and generalizing to unseen data, enabling them to perform tasks without explicit instructions. In its application, classification algorithms are used by analyzing traffic activity data from the gateway router's firewall and identifying the type of device through an information gathering process on all devices in the local computer network. This supporting information helps to identify the spammer, allowing the system to provide detailed information to the Network Administrator for further action, such as blocking the spammer by MAC

address or other measures.

In the research titled Internet Network Classification in Malikussaleh University Using the Naïve Bayes Method, the classification of internet traffic using the Naïve Bayes algorithm was tested with 234 test data points. For the Faculty of Engineering, using 10,945 training data points, the results showed an accuracy of 95.73% and an error rate of 4.27%. For the Faculty of Economics, using 3,902 training data points, the accuracy was 48.72% with an error rate of 51.28%. Lastly, for the Faculty of Social and Political Sciences, using 10,711 training data points, the accuracy was 94.87% with an error rate of 5.18%[3]. The Gaussian Naïve Bayes algorithm has shown good results in many studies and is also easy to implement in spam detection systems, particularly when the available data is continuous, making it a suitable choice for this research.. The expected output is a Boolean statement (spammer or normal). However, like other classification algorithms, Gaussian Naïve Bayes requires training data that already contains labels. To acquire unlabeled data from network traffic, a clustering process is needed using a clustering algorithm. Therefore, the K-Medoids algorithm is used because it has the advantage of overcoming the weaknesses of K-Means, which is sensitive to outliers [4]. Subsequently, these two algorithms will be applied to the traffic data collected from the Mikrotik router.

Literature Review

1. Computer Networks

Computer network is a connection between two or more nodes with the primary goal of exchanging data. Computer networks communicate through communication media, allowing them to share data, information, programs, and hardware (such as printers, hard drives, and webcams). The benefits of computer networks include: Resource Sharing, which ensures that all programs, equipment, and especially data can be used by anyone connected to the network, regardless of the location of the resource or the user. A common example is shared printer usage; Communication Media, where computer networks enable distant users to communicate with each other; Data Integration, where data processing can be done across multiple computers, making it easier for users to access and process data. An example of this is a client-server database program; Entertainment, where network users can enjoy various entertainment services, such as Facebook, chat, and online games; Efficiency or time-saving, as users who share data remotely through a network that integrates data can save time and effort in searching for information. The data shared through the network is also constantly up-to-date [5].

2. Clustering

Using unsupervised learning analysis, clustering helps identify similar objects or individuals by utilizing statistical analysis of each object. The goal of clustering is to maximize the similarity between objects within a single cluster while minimizing the similarity between different clusters. Cluster analysis is known by many terms. To group objects, hierarchical clustering and partition-based clustering are the two main methods commonly used. Hierarchical clustering produces a hierarchy in the form of a dendrogram. This hierarchy will consist of similar individuals or objects, while separate hierarchies will consist of dissimilar objects. On the other hand, partition-based clustering is a technique used to classify a set of observations in a dataset into groups based on data similarities. One of the methods used in hierarchical clustering is single linkage, while a commonly used method in partition-based clustering is K-Medoids, which is an improvement over K-Means. [6].

3. Partitioning Around Medoids (K-Medoids)

Created by Leonard Kaufman and Peter J. Rousseeuw, the K-Medoid algorithm, also known as PAM (Partitioning Around Medoid), is a partitioning algorithm that divides a dataset into several groups, similar to K-Means. The difference between the two lies in the determination of the group centers. While the K-Means algorithm uses the mean value of each group as the group center, K-Medoids uses actual data points as representatives, known as medoids, to serve as the center of each group [7]. The K-Medoids algorithm handles outliers effectively; the result of randomly selected data is referred to as the medoid, or center data. According to the algorithm’s rules, any data point can serve as the medoid [8].

Table 1. Component of Partitioning Around Medoids

Component	Description
Number of Clusters (k)	The number of clusters to form. Determines how many medoids are selected.
Medoids	The central points of each cluster, chosen from actual data points. Minimize the effect of outliers compared to centroids used in K-Means.
Initial Medoid Selection	Selecting k initial medoids randomly from the dataset.
Cluster Assignment	Each data point is assigned to the nearest medoid based on the chosen distance metric.
Medoid Update	After assigning data points to clusters, the medoid is updated by selecting the point within the cluster that minimizes total dissimilarity.
Objective Function	The goal is to minimize the total dissimilarity between medoids and their assigned points, iterating until medoids no longer change (convergence).

Table 1 Describe every component include the number of clusters, medoids, initial medoid selection, cluster assignment, medoid update and objective function which are used to clustering the data.

4. Classification

Classification in machine learning refers to the process of determining which category or class an input data point

belongs to, based on its features. It is a type of supervised learning, where a model is trained on labeled data – data where the correct output (class label) is known – and then used to predict the class labels of new, unseen data. In classification, the model is trained on a dataset that includes both input features and their corresponding labels. The goal is to learn a mapping from the input features to the labels so that the model can generalize to new, unlabeled data. Thus, the primary objective of classification is to assign an object to a predefined class or category. The classification process can also be defined as the construction of a model that classifies an object based on its characteristics. Gaussian Naive Bayes (Gaussian NB) is one of the fundamental algorithms used to solve classification problems, particularly when the features are assumed to follow a normal (Gaussian) distribution [9].

5. Gaussian Naïve Bayes Classifier

Gaussian Naïve Bayes is one of the simplest classification algorithms, based on Bayes' theorem, and it uses data with target classes or labels that are categorical in nature. The Gaussian Naïve Bayes classification method is one of the most commonly used methods for calculating probabilities and statistics. It is widely used for solving binary and multiclass classification problems, where the goal is to predict the class label of new data based on its features. Gaussian Naïve Bayes classifies new data into one of these classes using the input features. The probability density function (PDF) of the Gaussian distribution is used to calculate the likelihood of a feature given the class. After computing the posterior probability for each class, Gaussian NB assigns the new data point to the class with the highest posterior probability. For example, if the algorithm predicts that a data point has a 70% chance of being classified as "spam" and a 30% chance of being classified as "not spam," it will classify the email as "spam." This model is based on the assumption that each feature (variable) in the dataset follows a normal (Gaussian) distribution and that all features are independent of one another (the naive assumption). The Gaussian distribution is represented as $P(X_i|y = c)$ which is the probability of finding the likelihood, where μ_c is the mean, and δ_c^2 is the standard deviation for class c [10].

Gaussian Naïve Bayes is easy to implement and computationally efficient, particularly when dealing with large datasets. This makes it ideal for real-time applications like network traffic monitoring, where quick decisions are crucial [11], [12]. The Gaussian variant of Naïve Bayes is designed for continuous data, which is typical in network traffic analysis. The algorithm models the probability distribution of each feature using a Gaussian distribution, making it well-suited for the continuous variables often found in network logs [12]. Although the independence assumption of Naïve Bayes is often violated in practice, the algorithm still performs remarkably well in many real-world scenarios, particularly in high-dimensional datasets like those used in spam detection. This makes it competitive with more complex models, especially when interpretability and speed are prioritized over marginal gains in accuracy [13]. Compared to more sophisticated machine learning models, Gaussian Naïve Bayes has lower memory and processing requirements, making it practical for environments with limited computational resources, such as real-time spam detection on network devices [12].

Table 2. Component of Gaussian Naive Bayes

Component	Description
Gaussian Distribution	Each feature X_i in Gaussian NB is assumed to follow a Gaussian (normal) distribution for a given class $y = c$.
Mean (μ_c)	The mean of each feature X_i for class $y = c$. It is used to calculate the Gaussian distribution.
Variance (δ_c^2)	The variance of each feature X_i for class $y = c$. It determines the spread of the data in the Gaussian distribution.
Prior Probability $P(y = c)$	The initial probability of each class $y = c$, calculated as the proportion of data in that class before observing the features.
Likelihood $P(X_i y = c)$	The probability of a feature X_i given a class $y = c$, calculated using the Gaussian distribution.
Posterior Probability $P(y = c X)$	The probability of class $y = c$ given the data X , calculated using Bayes' Theorem.
Classification Decision	The class $y = c$ with the highest posterior probability is chosen.

Table 2 Describe every component include the gaussian distribution, mean, variance, independence assumption, prior probability, likelihood, posterior probability, and classification decision which are used to calculate probabilities and predict the class for new data.

Materials & Methods

1. Data Collection

The network traffic data from e-UnimalNet Wi-Fi users in the Informatics Building is located on the Mikrotik Routerboard under the Firewall section and the Connection subsection. To meet the requirements for a training and testing data comparison, data was collected four times at different intervals (with a training-to-testing data ratio of 80:20) on the following dates: August 30, 2024 at 09:47, September 02, 2024 at 10:52 and 14:48, and September 03, 2024 at 10:22, for the clustering process. Below is the combined data from these four collection periods.

Table 3. Traffic Data for Clustering Process

id	protocol	Src-address	dst-address	...	orig-packets	orig-bytes	...	fasttrack	srnat	dstnat
----	----------	-------------	-------------	-----	--------------	------------	-----	-----------	-------	--------

1	tcp	10.33.37.25:33086	74.125.68.188:5228	...	8	1527	...	false	true	false
2	tcp	10.33.38.8:59437	20.198.119.143:443	...	69	6148	...	false	true	false
3	tcp	10.33.37.94:43248	142.250.115.188:443	...	13	1870	...	false	true	false
4	tcp	10.33.37.165:39456	74.125.68.188:5228	...	25	2622	...	false	true	false
5	tcp	10.33.37.46:43874	64.233.170.188:5228	...	16	2437	...	false	true	false
...
20376	udp	10.33.37.83:40379	10.33.37.1:53	...	1	69	...	false	false	false
20377	tcp	10.33.37.83:37854	142.251.175.95:443	...	13	1279	...	false	true	false
20378	tcp	10.33.37.251:51768	172.30.10.150:8080	...	6	1311	...	false	true	false
20379	tcp	10.33.37.83:38402	103.107.186.12:80	...	2	112	...	false	true	false
20380	tcp	10.33.37.215:60404	23.193.97.59:443	...	8	1033	...	false	true	false

Table 3 shows the ongoing traffic data during the four data collection periods, amounting to 20,380 rows of data. Several features, such as orig-packets and orig-bytes, are observed to have constantly changing values based on the number of packets sent and their size in bytes from each connection made by the network users (src-address) to the destination (dst-address), using the protocol applied in OSI Layer-4. Clearly, this data is still "dirty," so before the clustering process, data preprocessing is required.

2. Methods

The first step is the collection of traffic data from the Mikrotik Routerboard, located on the 2nd floor of the Informatics Department Building at the Faculty of Engineering, Malikussaleh University, on Jalan Batam, Bukit Indah Campus, Malikussaleh University. The data is collected using the RouterOS API protocol on port 8728. The process flow for spammer detection on computer networks using the Gaussian Naïve Bayes classifier and K-Medoids as acquisition training data is illustrated in Figure 1.

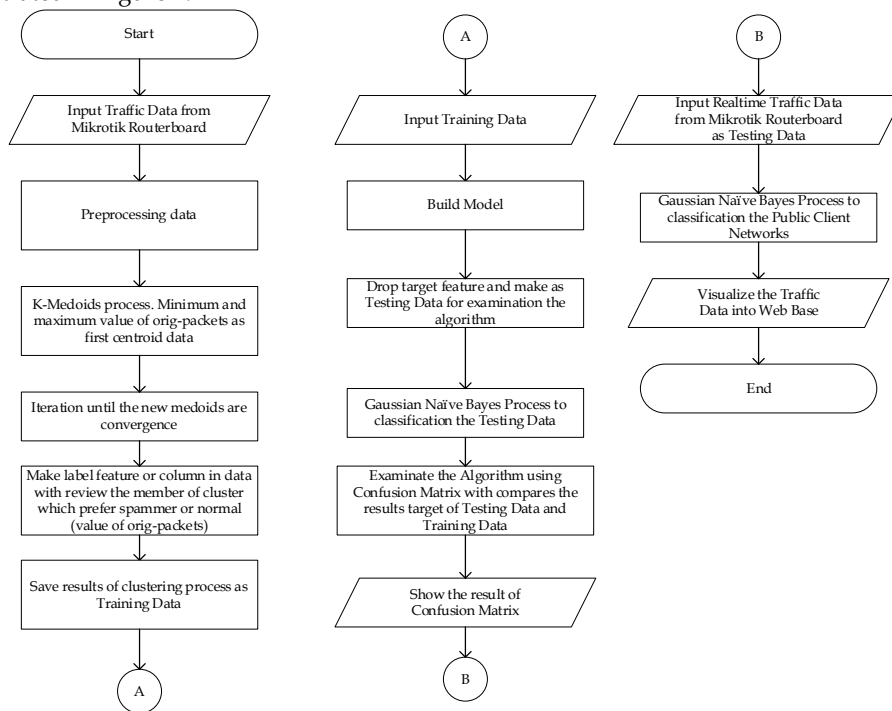


Figure 1. Algorithm System Design

The data then undergoes several preprocessing steps to make it relevant for the clustering and classification processes, such as removing features like timeout, tcp-state, icmp-type, icmp-code, icmp-id, reply-src-address, and reply-dst-address because their records are empty or null; removing rows with empty or null feature values; removing features such as orig-fasttrack-packets, orig-fasttrack-bytes, repl-fasttrack-packets, repl-fasttrack-bytes, expected, confirmed, dying, fasttrack, and dstnat because their records have the same values across all rows; splitting the string values in the src-address and dst-address columns, as the IP and port are combined, thus creating separate columns for IP and port; converting src-address-port and dst-address-port values that are none to 1; converting all string values true to 1 and false to 0; and converting the protocol column into numerical IDs according to the Layer-4 Transport Protocol standard on OSI.

After this, clustering is performed using the K-Medoids algorithm with $k = 2$, where the minimum and maximum values of the orig-packets feature serve as the two centroids. Equation (1) represents the distance matrix using Euclidean distance (d) with p_1 and p_2 being two data points and i adalah fitur, representing a feature, while Equation (2) represents the total distance between the new medoid and all points in the cluster.

$$d(p_1, p_2) = \sqrt{\sum_{i=1}^n (p_{1i} - p_{2i})^2} \quad (1)$$

$$\text{Total Dissimilarity} = \sum_{i=1}^n d(p_i, m) \quad (2)$$

The process then repeats by selecting a new medoid that has the lowest total dissimilarity value for each cluster, until both centroids converge (no longer change). A new feature called labels is created and filled with membership values for each row (1 or 0). The data is then reviewed to examine which memberships tend toward the highest values and which tend toward the lowest values for the orig-packets feature. High values indicate spammers, and low values indicate normal behavior.

After the clustering process, classification is performed using the Gaussian Naïve Bayes algorithm. The initial step involves inputting the training data acquired from the clustering process. A classification model is then created by calculating the mean and variance for each row of data for each class. Subsequently, using the previous training data, the labels feature is removed to test the classification algorithm. The classification process involves: Calculating the prior probabilities of spammer and normal based on the entire training data using Equation (3); Calculating the Gaussian probability distribution for both spammer and normal labels for each feature in each row using Equation (4); Calculating the independence assumption by multiplying the prior probability and the probability of each feature for both labels (spammer and normal) using Equation (5); Finally, calculating the posterior probability using Equation (6) for each label in each row.

$$P(y = c) = \frac{\text{Total Data in c class}}{\text{Total sum data}} \quad (3)$$

$$P(X_i | y = c) = \frac{1}{\delta_c \sqrt{2\pi}} e^{-\frac{(x_i - \mu_c)^2}{2\delta_c^2}} \quad (4)$$

$$P(X_1, X_2, \dots, X_n | y = c) = P(X_1 | y = c) \times P(X_2 | y = c) \times \dots \times P(X_n | y = c) \quad (5)$$

$$P(y = c | X) = \frac{P(X|y=c)P(y=c)}{P(X)} \quad (6)$$

Afterward, the class with the highest posterior probability is selected as the chosen class, resulting in the model's prediction for that data row. After obtaining the evaluation results of the classification algorithm, the next step is to implement the spammer detection on computer networks using Gaussian Naïve Bayes classifier and K-Medoids as acquisition training data into a fully operational system, as shown in Figure 2.

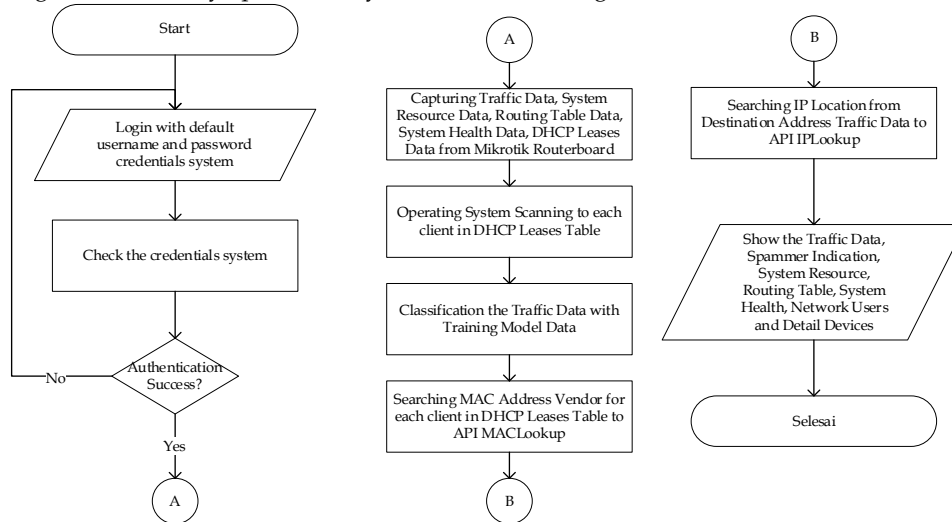


Figure 2. Web-Based System Design

In the running web-based system, the user, i.e., the network administrator, is required to log in using a username and password. If the input is correct, the system will retrieve real-time traffic data from the Mikrotik Routerboard, along with system resource data, routing table data, system health data, and DHCP Leases data. The system will also scan each client listed in the DHCP Lease table to identify the operating system of each device on the network. Then, the spammer detection process will be performed using the training data created in the previous steps, generating new target labels in the real-time traffic data. Additionally, the system will search for the Vendor of the MAC Address registered in the DHCP Leases to identify the vendor of the Network Interface Card (NIC) connected to the network as supporting data for device identification. The system will also locate the destination IP addresses in each row of the traffic data to determine the target and top access of the network. Afterward, the system will display the traffic data along with its spammer indications, Mikrotik system resources, routing table, system health, network users, and detailed information about their devices..

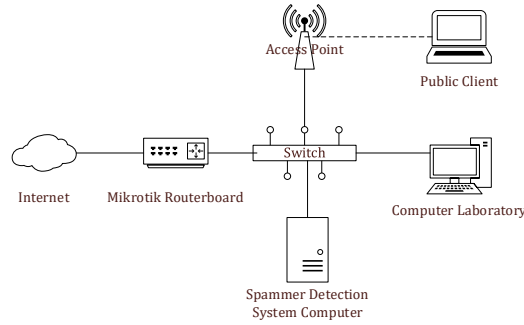


Figure 3. Spammer Detection Network Topology

Figure 3 illustrates the network topology of the system. The Mikrotik is connected to a Switch via the Ethernet port. Several devices are connected to the switch, including access points, computer labs, and general users, referred to as clients of the computer network. The Spammer Detection System will be connected to the switch, allowing it to interact with the Mikrotik to retrieve traffic data from users accessing the internet, as well as interact with the users to detect the type and identity of their devices.

This setup enables the spammer detection system to collect real-time data while also identifying the users on the network. Since the spam detection system is static and remains in place, an exception must be configured in the Mikrotik DHCP table to ensure that its IP address does not conflict with other devices.

3. Evaluation

By comparing the predicted target results with the training data, the algorithm is then evaluated using a Confusion Matrix, as shown in Table 4. A Confusion Matrix is a performance measurement tool in the form of a matrix used to calculate the classification accuracy for the classes used by the algorithm.

Table 4. Confusion Matrix

	Positive Prediction	Negative Prediction
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

The Confusion Matrix shows True Positive (TP) and True Negative (TN) values, which indicate the accuracy of the classification. Higher TP and TN values reflect greater accuracy, precision, and recall. Incorrectly predicted labels where the actual value is false are called False Positive (FP), while incorrect predictions where the actual value is true are referred to as False Negative (FN)[10].

From Table 4, the values for accuracy (Equation 7), precision (Equation 8), recall (Equation 9), and F1-score (Equation 10) can be calculated using the corresponding formulas.

$$accuracy (\%) = \frac{TP+TN}{TP+FN+TN+FP} \quad (7)$$

$$precision (\%) = \frac{TP}{TP+FP} \quad (8)$$

$$recall (\%) = \frac{TP}{TP+FN} \quad (9)$$

$$F1 \text{ score } (\%) = \frac{2 \times precision \times recall}{precision+recall} \quad (10)$$

The web-based system testing is conducted using Black Box Testing. Black Box Testing is a method used to identify errors in an application system, such as system function errors or missing application menus. It is a functional testing method for the application system. Black Box Testing focuses on testing the application based on its details, such as the user interface, the functions within the application, and the alignment of functional workflows with the system design intended by the developer. The type of Black Box Testing used is Functional Testing, which does not consider the programming language and is performed from the user's perspective to uncover inconsistencies and ambiguities in the data specifications. [14].

Results and Discussion

In the raw data collected from Mikrotik Routerboard traffic over four different time periods, there are 29 features with 20,380 rows. Upon investigation, it was found that there are 4 features with non-null values, namely tcp-state, icmp-type, icmp-code, and icmp-id, as shown in Figure 4.

#	Column	Non-Null	Count	Dtype
0	id	20380	non-null	object
1	protocol	20380	non-null	object
2	src-address	20380	non-null	object
3	dst-address	20380	non-null	object
4	reply-src-address	20380	non-null	object
5	reply-dst-address	20380	non-null	object
6	tcp-state	14498	non-null	object
7	timeout	20380	non-null	object
8	orig-packets	20380	non-null	int64
9	orig-bytes	20380	non-null	int64
10	orig-fasttrack-packets	20380	non-null	int64
11	orig-fasttrack-bytes	20380	non-null	int64
12	repl-packets	20380	non-null	int64
13	repl-bytes	20380	non-null	int64
14	repl-fasttrack-packets	20380	non-null	int64
15	repl-fasttrack-bytes	20380	non-null	int64
16	orig-rate	20380	non-null	int64
17	repl-rate	20380	non-null	int64
18	expected	20380	non-null	bool
19	seen-reply	20380	non-null	bool
20	assured	20380	non-null	bool
21	confirmed	20380	non-null	bool
22	dying	20380	non-null	bool
23	fasttrack	20380	non-null	bool
24	srcnat	20380	non-null	bool
25	dstnat	20380	non-null	bool
26	icmp-type	23	non-null	float64
27	icmp-code	23	non-null	float64
28	icmp-id	23	non-null	float64

Figure 4. Print Result of Describe Data frame Python

The next step is to perform data preprocessing to remove features with null values, as well as features that have the same value across all 20,380 rows. Clustering algorithm rely on complete data to compute results. Missing values can disrupt these calculations, making them inaccurate or impossible to compute [12]. Besides, Leaving null values untreated can lead to biased results or misinterpretation even distorting the outcome of the analysis [13].

Additionally, the string values of IP and port in the src-address, dst-address, reply-src-address, and reply-dst-address features are separated. The string values true and false are converted to Boolean, and the protocol is converted to numerical values. The purpose of separating the IP and port string values is to extract the port information while keeping the IP as an identifier for the connection row. The result is shown in Figure 5, where the number of features is reduced to 18.

#	Column	Non-Null	Count	Dtype
0	protocol	20380	non-null	int64
1	timeout	20380	non-null	object
2	orig-packets	20380	non-null	int64
3	orig-bytes	20380	non-null	int64
4	repl-packets	20380	non-null	int64
5	repl-bytes	20380	non-null	int64
6	orig-rate	20380	non-null	int64
7	repl-rate	20380	non-null	int64
8	seen-reply	20380	non-null	bool
9	assured	20380	non-null	bool
10	srcnat	20380	non-null	bool
11	src-address-ip	20380	non-null	object
12	src-address-port	20380	non-null	object
13	dst-address-ip	20380	non-null	object
14	dst-address-port	20380	non-null	object
15	reply-src-address-ip	20380	non-null	object
16	reply-src-address-port	20380	non-null	object
17	reply-dst-address-ip	20380	non-null	object
18	reply-dst-address-port	20380	non-null	object

Figure 5. Print Result of Describe Data frame Python After Preprocessing Data

In the clustering process using K-Medoids, not all features were included in the process, as some contain string values and serve as identifiers for the rows. Therefore, only the following 12 features were used: src-address-port, dst-address-port, protocol, orig-packets, orig-bytes, repl-packets, repl-bytes, orig-rate, repl-rate, seen-reply, assured, and srcnat.

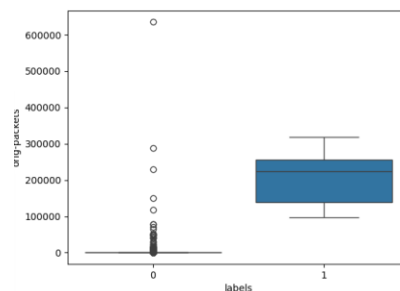


Figure 6. Boxplot of Result of Clustering Data with orig-packets feature

After the clustering process, the results showed 12 spammer data points and 20,368 normal data points, as illustrated in the boxplot in Figure 6. It can be observed that for label 0, the data distribution is uneven for orig-packets, with many outliers present. In contrast, for label 1, the distribution appears more balanced within the box, indicating a normal data

range with fewer outliers.

Orig-packets represents the number of packets sent by the network client, recorded on the Mikrotik Routerboard, and its values are irregular. Therefore, it can be concluded that in label 1, the values are more consistent, indicating that spammers are likely to belong to this label. As a result, the dataset consists of 12 spammer data points and 20,368 normal data points. The labels feature generated from the clustering process is then added to the main dataset after data preprocessing, making the data ready for model training.

In building the Gaussian Naïve Bayes classification model using the prepared training data, features such as timeout, src-address-ip, dst-address-ip, reply-src-address-ip, and reply-dst-address-ip must be separated because they contain string values. However, these features should not be removed as they serve an important role as identifiers. Next, the labels feature is separated to test the classification algorithm's accuracy by comparing the predicted labels with the testing labels. The training data is split randomly into training and testing data with an 80:20 ratio. The prediction results are evaluated using the Confusion Matrix, as shown in Table 5.

Table 5. Confusion Matrix of Network Traffic Computer

	Positive Prediction	Negative Prediction
Actual Positive	True Positive = 4063	False Negative = 12
Actual Negative	False Positive = 0	True Negative = 1

The results show that the True Positive (predicted positive with actual positive) is 4,063, the False Negative (predicted negative with actual positive) is 12, the False Positive (predicted positive with actual negative) is 0, and the True Negative (predicted negative with actual negative) is 1. When displayed in a bar chart, the evaluation results will be shown in Figure 7.

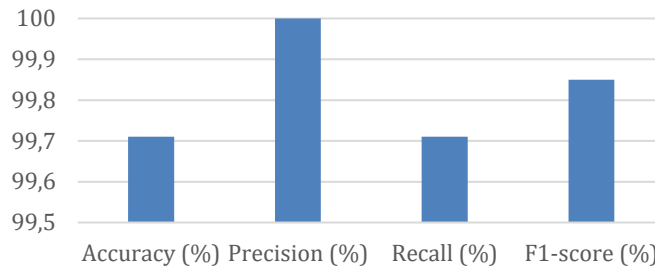


Figure 7. Evaluation Gaussian Naïve Bayes of Confusion Matrix

In this research using the Gaussian Naïve Bayes classification, the accuracy is 99.71%, meaning that 12 out of 4,076 data points were predicted incorrectly. The precision is 100%, indicating that none of the data points that were actually normal were predicted as spammers. The recall is 99.71%, meaning that 12 out of 4,076 data points that were actually spammers were predicted as normal. The F1-score is 99.85%, reflecting a well-balanced performance of the model in correctly detecting spammers.

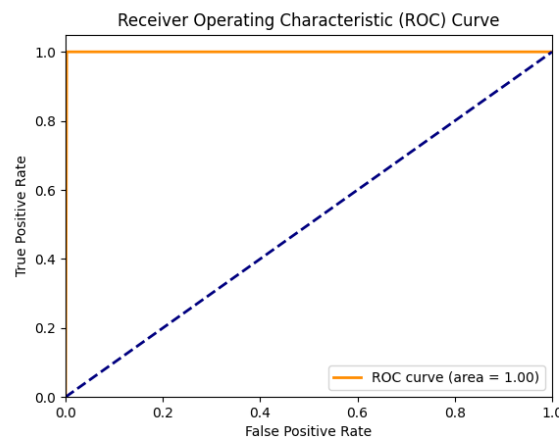


Figure 8. Receiver Operating Characteristic Curve of Evaluation Gaussian Naïve Bayes

As can be seen in Figure 8, True Positive Rate (TPR) describe the ROC curve hits the top-left corner, which means the classifier has a True Positive Rate (Sensitivity/Recall) of 1.0. This indicates that the model correctly classifies actual positives (spammers) with no mistakes. False Positive Rate (FPR) describe the curve remains flat along the top without any rise, indicating a False Positive Rate (FPR) of 0.0. This means that the model does not misclassify any negative instances (normal traffic) as positive (spammer). It handles all the negative cases correctly. An AUC of 1.0 suggests that the model's performance is perfect – it distinguishes between the two classes (spammer vs. normal) flawlessly. It means the classifier achieves 100% accuracy in predicting both classes across all decision thresholds.

Table 6. Accuracy, Precision, Recall, and F1-score Categories Percentage

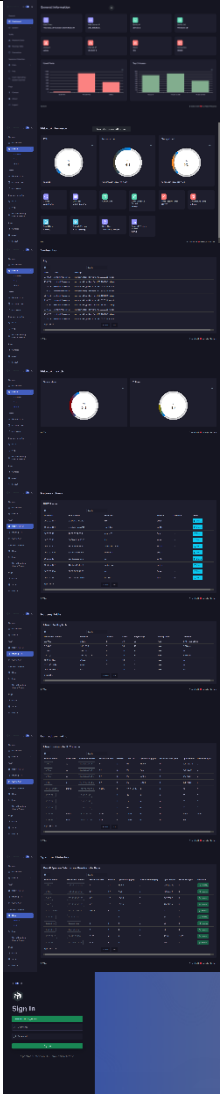
Percentage	Category for Accuracy, Precision, Recall, and F1-	Interpretation
------------	---	----------------

Score		
90% - 100%	Excellent	The model performs exceptionally well, making very few mistakes and having a great balance between precision and recall.
80% - 89%	Good	The model performs well but may have a few minor errors. Precision and recall are still fairly balanced and reliable.
70% - 79%	Fair	The model performs adequately with some acceptable errors. However, mistakes occur more often. It is acceptable but not ideal.
60% - 69%	Poor	The model has more errors, and precision or recall may be unbalanced. It can still be used, but significant improvement is needed.
50% - 59%	Very Poor	The model performs badly, with frequent errors such as both false positives and false negatives. Precision and recall might be severely unbalanced.
< 50%	Fail	The model is ineffective. Predictions could be nearly random, or worse than random guesses. It requires substantial improvement or restructuring.

Based on the category table in Table 6, the application of the Gaussian Naïve Bayes algorithm using 20,380 training data points for spammer detection on a computer network falls into the Excellent category. This means its effectiveness is well-aligned, as the model performs exceptionally well, making very few mistakes and maintaining a great balance between precision and recall.

For the testing of the web-based system implementation, black box testing was used to validate and assess the system's performance, as shown in Table 7.

Table 7. System Blackbox Testing

No	Feature	Test Case	Expected Result	Test Result	Outcome
1	Login Page	User inputs the correct username and password and clicks the sign-in button	Successfully logs into the system and displays the dashboard		Valid
2	System Resource Page	User clicks on the "System" menu and selects "Resource" from the sidebar	Displays Mikrotik resource information		Valid
3	System Log Page	User clicks on the "Log" menu from the sidebar	Displays the system log table		Valid
4	System Health Page	User clicks on the "Health" menu from the sidebar	Displays Mikrotik temperature and voltage information		Valid
5	Network Users Page	User clicks the "Network Users" button from the sidebar	Displays the DHCP Leases table		Valid
6	Routing Table Page	User clicks the "Routing Table" button from the sidebar	Displays the routing table		Valid
7	Connection Traffic Page	User clicks the "Connection" menu from the sidebar	Displays connection traffic data		Valid
8	Spammer Detection Page	User clicks the "Data" menu under the spammer detection sidebar	Displays connection traffic data with an additional indication column		Valid
9	Logout Page	User clicks the "Logout" button from the sidebar	Redirects to the login page with a success flash message		Valid

Conclusions

The results of this research demonstrate that the application of the Gaussian Naïve Bayes algorithm is well-suited for the classification model of spammers on computer networks, as it shows a high evaluation level and effectively classifies network traffic into normal and spammer categories. The evaluation results for the Gaussian Naïve Bayes algorithm show an accuracy of 99.71%, meaning that 12 out of 4,076 data points were predicted incorrectly. The precision is 100%, indicating that no data points that were actually normal were predicted as spammers. The recall is 99.71%, meaning that 12 out of 4,076 data points that were actually spammers were predicted as normal. The F1-score is 99.85%, indicating a well-balanced model performance in accurately detecting spammers. With all four parameters scoring above 70%, it is proven that the Gaussian Naïve Bayes algorithm is capable of delivering high-performance results.

The training data acquisition, based on the results of K-Medoids clustering of network traffic data into two clusters, identified 12 spammer data points and 20,368 normal data points out of a total of 20,380 data points. For label 0, the data distribution for orig-packets is uneven and contains many outliers. In contrast, label 1 is almost balanced within the box, indicating a normal data range with fewer outliers. The results demonstrate that dividing the network traffic data into two categories based on proximity to the orig-packets variable makes the distinction between spammer and normal data relevant, as label 1 values are more consistently distributed, indicating the location of spammers, with the assumption that the traffic data was collected during peak hours and over four different time periods.

In this research, the data collected was limited to Layer-4 OSI network traffic. Future researchers could use higher-layer traffic, closer to Layer-7 OSI, where blocking processes are frequently carried out by firewall software. Additionally, to keep bias low, future research could use training data that is labeled based on the decisions of a network administrator. This research also used data from a single Router Endpoint Gateway Mikrotik Routerboard. Future researchers are encouraged to collect data from other routers, allowing the model to be applied in various environments and broader settings. Black box testing of the system has been conducted, and all features have been validated as functioning correctly. For future development, the existing system can be further enhanced to accommodate larger network scales, such as MAN (Metropolitan Area Network) up to the Main Gateway Router, before directly interfacing with the public IP. This would provide a broader perspective on the system's scalability and efficiency.

Acknowledgments

The author would like to express gratitude to the Technical Implementation Unit of Central Computer Center, Malikussaleh University and the Informatics Engineering Program of Universitas Malikussaleh for their assistance, facilitation, and support throughout the research, from data collection and system development to the completion of this paper.

References

- [1] APJII, "Jumlah Pengguna Internet Indonesia Tembus 221 Juta Orang," Asosiasi Penyelenggara Jasa Internet Indonesia. Accessed: Jun. 06, 2024. [Online]. Available: <https://apjii.or.id/berita/d/apjii-jumlah-pengguna-internet-indonesia-tembus-221-juta-orang>
- [2] F. P. E. Putra, A. Zulfikri, M. A. Huda, Hasbullah, Mahendra, and M. Surur, "Analisis Keamanan Jaringan Dari Serangan Malware Menggunakan Firewall Filtering Dengan Port Blocking," *Digital Transformation Technology (Digitech)*, vol. 3, no. 2, pp. 857–863, Sep. 2023.
- [3] E. Darnila, Z. Yunizar, and D. Gibran Alinda, "INTERNET NETWORK CLASSIFICATION IN MALIKUSSALEH UNIVERSITY USING NAÏVE BAYES METHOD," *METHOMIKA: Jurnal Manajemen Informatika & Komputerisasi Akuntansi*, vol. 5, no. 1, pp. 48–53, Apr. 2021, doi: 10.46880/jmika.Vol5No1.pp48-53.
- [4] F. Zahra, A. Khalif, and B. N. Sari, "PENGELOMPOKAN TINGKAT KEMISKINAN DI SETIAP PROVINSI DI INDONESIA MENGGUNAKAN ALGORITMA K-MEDOIDS," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 2, pp. 1243–1249, 2024, doi: 10.23960/jitet.v12i2.4199.
- [5] A. V. Mananggal, Ifrina Mewengkang, and A. C. Djamen, "PERANCANGAN JARINGAN KOMPUTER DI SMK MENGGUNAKAN CISCO PACKET TRACER," *Jurnal Pendidikan Teknologi Informasi dan Komunikasi*, vol. 1, no. 2, pp. 119–131, Apr. 2021.
- [6] A. V. Septiani, R. A. Hasibuan, A. Fitrianto, Erfiani, and A. N. Pradana, "Penerapan Metode K-Medoids dalam Pengklasteran Kab/Kota di Provinsi Jawa Barat Berdasarkan Intensitas Bencana Alam di Jawa Barat pada Tahun 2020-2021," *Statistika*, vol. 23, no. 2, pp. 147–155, Nov. 2023, doi: 10.29313/statistika.v23i2.3057.
- [7] Rizal, H. A. K. Aidilof, Mukhlis, and K. Nur, "Penerapan Algoritma K-Medoid Dalam Perbandingan Daya Serap Akademik Siswa Sekolah Perkotaan dan Sekolah Pedesaan Selama Masa Pandemi," *TEKNO KOMPAK*, vol. 16, no. 2, pp. 85–97, 2022.
- [8] U. Linarti, A. Rahmawati, A. Hendri Soleliza Jones, and L. Zahrotun, "Penerapan Metode K-Medoids Guna Pengelompokan Data Usaha Mikro, Kecil dan Menengah (UMKM) Bidang Kuliner Di Kota Yogyakarta," *Jurnal Ilmu Komputer dan Sistem Informasi (JIKOMSI)*, vol. 7, no. 1, pp. 37–45, 2024.
- [9] M. Afriansyah, J. Saputra, V. Yoga Pudya Ardhana, Y. Sa, and U. Qamarul Huda Badaruddin, "ALGORITMA NAIVE BAYES YANG EFISIEN UNTUK KLASIFIKASI BUAH PISANG RAJA BERDASARKAN FITUR WARNA," *Journal of Information Systems Management and Digital Business (JISMDB)*, vol. 1, no. 2, pp. 236–248, Jan. 2024.

- [10] Y. Naufal, R. Putro, A. Afriansyah, and R. Bagaskara, “Penggunaan Algoritma Gaussian Naïve Bayes & Decision Tree Untuk Klasifikasi Tingkat Kemenangan Pada Game Mobile Legends,” *JUKI : Jurnal Komputer dan Informatika*, vol. 6, no. 1, pp. 10–26, May 2024.
- [11] Prashant, “Implementation of Gaussian Naive Bayes in Python Sklearn,” *Analytics Vidhya*. Accessed: Oct. 24, 2024. [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/11/implementation-of-gaussian-naive-bayes-in-python-sklearn/>
- [12] Rohan Vats, “Gaussian Naive Bayes: What You Need to Know?,” *upGrad*. Accessed: Oct. 24, 2024. [Online]. Available: <https://www.upgrad.com/blog/gaussian-naive-bayes/>
- [13] D. Sharma *et al.*, “Naive Bayes, Clearly Explained,” *Statquest!!!*. Accessed: Oct. 24, 2024. [Online]. Available: <https://statquest.org/naive-bayes-clearly-explained/>
- [14] N. Muhammad Arofiq, R. Ferdo Erlangga, A. Irawan, and A. Saifudin, “Pengujian Fungsional Aplikasi Inventory Barang Kedatangan Dengan Metode Black Box Testing Bagi Pemula,” *OKTAL : Jurnal Ilmu Komputer dan Science*, vol. 2, no. 5, pp. 1322–1330, May 2023, [Online]. Available: <https://journal.mediapublikasi.id/index.php/oktal>