Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

Optimizing Intrusion Detection Efficiency through Filtered clusterer, K-Mean clusterisn and Density-Based Clustering Without Count Attribute

Pratik Jain[1], Rajni Jainwal[2], Prof. Dr. Dharmendra Mehta[3] , Dr. Jyotsana Soni[4], Javed R. Attar[5], Prashant Sharma[6]

[1]IPS Academy, Institute of Engineering and Science, Indore,
[2]Samrat Vikramaditya Vishwavidyalaya, Ujjain,
[3]Samrat Vikramaditya Vishwavidyalaya, Ujjain,
[4]Prestige Institute of Management and Research, Dewas,
[5]MET'S Institute of Engineering, Nashik, Maharashtra,
[6] IPS Academy, Institute of Engineering and Science, Indore,

Received: 12 February 2026 *Accepted:26 March 2026* *Published: 13 April 2026*

KEYWORDS

Threshold tuning, ensemble methods, semi-supervised learning, intrusion detection, fraud detection, system monitoring, alert fatigue, accuracy, robustness, efficiency, confidence

ABSTRACT:

Anomaly detection is important in systems like intrusion detection, fraud detection and system monitoring. These systems have a problem with false positives. This can cause fatigue make the system less efficient and reduce confidence in it. This abstract looks at ways to deal with positives in anomaly detection. We will talk about threshold tuning, ensemble methods and semi-supervised learning. These techniques can help lower the positive rate. They can also make the system more accurate and robust. By using these techniques we can reduce fatigue. The system becomes more efficient. People trust it more. Anomaly detection systems become valuable, for applications. Anomaly detection helps in intrusion detection, fraud detection and system monitoring. Anomaly detection is a part of these systems. It helps to identify patterns. Anomaly detection is used in areas. It has some attributes which is responsible for system design. In this paper we have remove one attribute and provide explanation. By removing the attribute the result is compared.

. Introduction

In years network intrusion has become a big concern for organizations. This is because people over the world are misusing computer technology. With encryption, VPNs and firewalls in place malicious attacks can still happen. So it's necessary to have a monitoring system to keep an eye on things. Intrusion detection is a tool that provides dynamic security to organizations. It does this by analyzing data and counter-attacking. A Network-Based IDS or NIDS is one system. It monitors network traffic to identify threats from, outside the network. IDS can be classified into three categories. They are signature-based, anomaly-based or protocol modelling-based. Most IDS systems use all three models to make security stronger. These way organizations can better protect themselves from network intrusions.

Categories of Intrusion Detection System

1. Signature-Based Detection Systems

This is a way to find attacks. It works by looking at network traffic or system behavior and comparing it to a list of known attack patterns. These patterns are like fingerprints that help identify attacks. The process has two parts:

A. Signature Generation: This is where we make a list of attack patterns. We do this by looking at how known attacks work and finding patterns in network traffic or system logs. We add these

Journal of Chemical Health Risks
www.jchr.org
JCHR (2026) 16(2), 2778 - 2794

patterns to the list, which the intrusion detection system uses to find attacks.

B. Signature Matching:
The system checks network traffic or system behavior for known attack patterns. If it finds a match it sends an alert. This is a way to find known attacks but it does not work for new attacks that we have not seen before. Attackers can change their attacks to avoid being caught. To find these attacks we use new techniques like machine learning to learn from past attacks and find unusual behavior.

2. Anomaly-Based Detection System
This system looks for behavior on a computer network. It figures out what is normal and then watches for anything that's different. This system can find attacks like denial-of-service or other strange actions. It uses technology to find these things. One good thing about this system is that it can find attacks that we have not seen before.. It can also make mistakes if it does not know what normal behavior is or if the system changes. To make this system work better we need to know what normal behavior is and update it as needed. We should also use security measures, like passwords or virus scanning to keep things safe.

3. Protocol Modelling
Protocol modeling is a way to analyze computer network protocols. It makes a model of the protocol. Simulates its behavior to find security threats. This model includes states and messages that govern how the protocol works. It helps us find the protocols weaknesses and test how well it resists types of attacks. Network security professionals and researchers use protocol modeling to improve network security. They also use it to develop protocols that meet performance and security requirements. Protocol modeling is a tool, for Intrusion Detection System and network security.

II. Literature tracery

[1] Axelsson (2000) gives a list of IDS types. He says they can be grouped by detection methods and audit sources.. He also points out that they have problems like false alarms and are hard to test. [2] Denning (1987) starts the study of IDS. She uses statistics to detect behavior. Her work helps create IDS research. [3] Scarfone and Mell (2007) give tips on setting up IDS/IPS systems. They talk about setup, deployment and challenges. [4] Bace and Mell (2001) explain what IDS are, how they work and how to set them up. Their guide helps people understand IDS. [5] Stallings and Brown (2018) cover IDS as part of cybersecurity. They focus on real-life use and integration with security rules. [6] Roesch (1999) introduces Snort, an IDS. It analyzes packets in time. [7] Northcutt (2002) talks about setting up network IDS. He discusses real-life issues. [8] McHugh (2000) checks how well IDS work and how to test them. He finds problems with benchmarking. [9] Debar et al. (1999) Make a list of IDS types. They focus on detecting misuse and anomalies. [10] Lunt (1993) looks at early intrusion detection methods. She discusses statistics and rules. [11] Vigna and Kemmerer (1999) introduce NetSTAT, a network IDS. It detects attack patterns. [12] Forrest et al. (1996) Propose detecting anomalies in system calls. They introduce "self vs non-self". [13] Kendall (1999) creates datasets for IDS testing. He says benchmark datasets are needed. [14] Sommer and Paxson (2010) criticize machine learning in IDS. They say it has real-life limits. [15] Creech and Hu (2014) propose a semantic-based host IDS. It uses features. [16] Liao and Vemuri (2002) use k-NN classification for intrusion detection. They improve anomaly detection. [17] Lee and Stolfo (2000) introduce data mining for IDS features. [18] Lee et al. (1998) Show data mining detects intrusions in network traffic. [19] Patcha and Park (2007) survey anomaly detection techniques. They include statistics and machine learning. [20] Ye et al. (2002) Use models for anomaly detection. They improve rare event detection. [21] Cannady (1998) explores networks for misuse detection. It recognizes patterns. [22] Tavallaee et al. (2009) Analyze the KDD Cup 99 dataset. They find redundancy and bias. [23] Sharafaldin et al. (2018) Introduce the dataset. It has traffic for modern IDS testing. [24] Kruegel et al. (2003) Detect web-based attacks using anomaly detection. [25] Mukkamala et al. (2002) Compare networks and

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

SVM for intrusion detection. [26] Zhang et al. (2008) Apply the Random Forest algorithm to IDS. They achieve accuracy. [27] Wang and Stolfo (2004) propose payload-based anomaly detection for network IDS. [28] Sekar et al. (2002) Introduce specification-based IDS to reduce alarms. [29] Freund and Schapire (1997) introduce the boosting algorithm. Its later used in IDS. [30] Kim et al. (2016) Apply LSTM networks for intrusion detection. [31] Javaid et al. (2016) Use learning for automatic feature extraction in IDS. [32] Wang et al. (2018) Propose HAST-IDS using spatial-temporal features. [33] Yin et al. (2017) Apply RNN for intrusion detection in network traffic. [34] Zhou et al. (2019) Survey machine learning-based IDS techniques. [35] Ring et al. (2019) Analyze IDS datasets and challenges. [36] Hindy et al. (2020) Present a taxonomy of IDS systems. They include AI-based approaches. [37] Khraisat et al. (2019) Survey IDS techniques, challenges and future directions. [38] Dua and Du (2016) explore data mining and machine learning in cybersecurity. [39] Behl and Behl (2017) discuss cybersecurity strategies and IDS. [40] García-Teodoro et al. (2009) Study anomaly-based IDS. [41] Bishop (2003) provides a foundation for computer security and IDS. [42] Axelsson (1999) explains the base-rate fallacy problem in IDS. [43] Novak and McHugh (2004) discuss IDS testing and accuracy. [44] Gu et al. (2007) Introduce BotHunter, for detecting botnet infections. [45] Julisch (2003) proposes clustering techniques to reduce IDS alert overload.

III. Problem identification

An Intrusion Detection System or IDS is a system that provides network security and defence. It can automatically recognize suspicious activity. Intrusion is handled by an intrusion detection system. The challenge that intrusion detection systems (IDS) address is that of balancing how to defend computer networks and systems from attacks and compromise. In particular, IDS is intended to identify and take action on attempts to undermine the confidentiality, integrity, or availability of network resources. A few of the specific tasks which intrusion detection systems can address are:

Detecting Attacks: One of the biggest problems in intrusion detection is detecting attacks on-line correctly. It also requires separating legitimate from malicious network traffic and identifying attacks when they are launched.

False Positives: IDSs can also produce false alarms, which can waste time and resources. False positives may be triggered by various things including misconfigurations, network anomalies, or even legitimate traffic that looks suspicious.

False Negatives: Attacks may not be detected by an IDS, leaving vulnerabilities open to potential exploits. False negatives can be induced by attacks that evade detection or by limitation. An Intrusion Detection System or IDS is one that helps in keeping a network secure. It protects the network by detecting activities on its own. Everything is controlled by the IDS. The problem that everything from A to Z IDS solves is to protect computer networks and computer systems from attacks breaches. IDS is tailor-made to identify and prevent attacks against network resources.

Scalability: IDS agents should be capable of monitoring high bandwidth links, and applying appropriate traffic analysis techniques when network traffic volume increase. This implies the design of efficient and scalable algorithms for the identification and analysis of network traffic. It is compliance with the fact that the environment is constantly evolving. Smart IDS mechanisms should comply with evolving network environment, and new as well the future security threats that are considered as unknown threats. This involves the on-going surveillance and revision of the identification heuristics as well as rules of the IDS.

1. Predefined normal behavior

Normal Behavior Established in Advance When we speak about predefined normal behavior in intrusion detection, we mean that we establish a baseline by which determines how a computer

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

network or a system will act. That norm is what the network or system is doing in its normal operation. We establish that baseline by observing the network and system over time and determine the norm. Then we define what's normal in terms of rules. When we have this rule set we can implement agents which look at the network and system to identify anything that breaks the rules. Then IDS can identify potential security threats or attacks and notify security personnel. That helps us know when there could be a problem. Predefined normal behavior has many benefits, Including:

Enhanced accuracy: Based on the normal behavior, the IDS can more effectively differentiate its behavior with the nature of the comparisons, which contributes to decreasing FPs and FNs.

Benefits: Efficiency gains: Having a predefined "normal behavior" can facilitate the detection process and the IDS can have a predefined place on looking for suspicious behavior outside the boundaries of what we define as "normal behavior."

Improved Scalability: Normal Behavior profiles allow an IDS to better manage high traffic loads and to adapt to network growth.

1) Records the profile (pattern) of user's normal behavior into database
2) It analyses the normal behavior by the stored pattern.

When there is a significant deviation, we can say that there is an anomalous activity. For IDS, the detection rate and accuracy should be high and false alarm rate should be low and the performance of IDS is normally assessed using the detection rate (DR), accuracy (AC) and false alarm rate (FAR) as in eq. In summary, predefined normal behavior for intrusion detection contributes to a valuable ability to recognize potential security risks, which helps in securing computer networks and systems in the study against attacks.

2. Predefined Intrusion Behavioral

It's a group of attack patterns or signatures which have been analyzed and grouped by experts. A behavior, called predefined intrusion behavior, can be used by IDS to detect potential attacks and threats. They implement a reference set of known intrusion behavior on which they build a set of rules or policies. These policies or rules can be adapted to be applied on live traffic to detect intrusion. Predefined intrusion behavior may be associated with different attacks such as port scanning, password guessing, buffer overflow or many more. Through identification and classification of such intrusion signatures (patterns), security experts can develop rules or policies to allow an IDS efficient detection and mitigation against potential threats. However, there are limitations when using predefined intrusion behavior. Attackers are inventing new, more sophisticated ways to hide and attack. Predefined intrusion behavior can get outdated over time, and that might not be able to detect the new or evolving intrusion. To overcome this weakness, IDSs can also rely on anomaly-based detection, which aims at detecting anomalous or abnormal behaviors which are not in line with normal usage. Combining predefined intrusion behavior with anomaly-based detection, IDSs can adopt a more holistic intrusion detection technique and improve network security. It generally stores the behaviour of vicious activity that is associated to violation & then detects the potential of intrusion based on the obtained behaviour. It has a very less false alarm and very high detection accuracy. The disadvantage is that it can only detect predefined behaviour violation.

Detection rate, the true positive rate or recall, the ability of a system to detect attack. In an intrusion detection system, it represents how many real attacks are successfully detected. A higher detection rate indicates that the system is effective at capturing malicious activities without missing them. However, focusing only on detection rate may sometimes lead to an increase in false alarms if the system becomes too sensitive.

Accuracy reflects the overall correctness of a system by considering both correctly identified

Journal of Chemical Health Risks
www.jchr.org
JCHR (2026) 16(2), 2778 - 2794

positive and negative instances. It is calculated as the proportion of total predictions that are correct. Accuracy provides a general idea of system performance, but it may not always be reliable in cases where the dataset is imbalanced. For example, if normal activities are far more frequent than attacks, a system may achieve high accuracy simply by predicting most instances as normal, even if it fails to detect actual threats.

False alarm rate, also called the false positive rate, measures how often the system incorrectly labels normal instances as abnormal or malicious. In practical scenarios, a high false alarm rate can reduce trust in the system and increase workload, as administrators must investigate many alerts that turn out to be harmless. Therefore, minimizing false alarms is essential for maintaining efficiency and usability.

Table 3.1 Confuision matrix

Actual	Predicted Normal
Normal	TN
Intrusions	FN

True Positive (TP): The attack is detected as an attack.

True negative (TN): The benign data were identified as benign.

False Positive (FP): A normal datum is identified as an attack.

False Negative (FN): When a normal data is classified as attack.

This work aims at the assessment of intrusion detection techniques on the KDD Cup 1999 dataset. The dataset was created and maintained by the MIT Lincoln Labs in 1998 as part of the DARPA Intrusion Detection Evaluation Program. Surveys and compares different intrusion detection techniques and the aim of this work is to detect the most relevant ones. We tackle the issue of intrusion detection, where normal data is predicted as an intrusion in this paper. The plagiarized has been removed from the initial sources. A widely used dataset for evaluation of the KDD Cup 1999 data is used as the benchmark in this study. It is a military network environment, and it contains data of both normal and intrusive activities that can be used for evaluating. This data set is considered the standard for testing intrusion detection systems. There are 41 attributes used in this dataset to determine whether the input belongs to a normal or anomaly class. These attributes include duration, protocol type, service, flag, source bytes, destination bytes, and various rates such as error rates and same server rates. The KDD Cup 1999 dataset is considered a de facto standard for intrusion detection evaluation. It was designed to test the ability of machine learning models to differentiate between normal network traffic and various modes of attacks. Each entry in the dataset is a connection, which is characterized by a set of 41 features and labeled as either normal or as an attack specific category. These attributes are generally clustered into several categories based on the type of information they represent:

1. Basic Features

These are all features that can be derived from the packet headers without any payload examination. They represent basic characteristics of a connection, such as how long it lasted, the protocol type (TCP, UDP, ICMP), the service (e.g., HTTP, FTP), and the status of the connection (normal or failed). They also include the number of bytes sent from the source to the destination. Basic features allow us to detect very simple patterns in network traffic behaviors.

2. Content-Based Features

The content features are calculated from the contents of the packets. They are appropriate for detecting anomalous activities such as failed login attempts, number of file creation operations, accessing privileged command. These features are particularly useful for detecting attacks in which an attacker gains access or otherwise siphons the system resources.

3. Time-Based Traffic Features

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

These features examine connections within a short-duration time span (i.e., the last two seconds). They track trends, such as the amount of connections to the same host or service during that period of time. This allows identification of fast scanning and sweeping, which are typical in denial-of-service attacks.

4. Host-Based Traffic Features

The host-based attributes differ from time-based ones in the wide time granularity and historical notion for each host instance. These metrics include things like: how many connections have been made to the same destination host in the last 2 hours, or what percentage of connections experienced errors. Such characteristics are useful for detecting slow and sustained attacks.

5. Traffic Behavior and Error Rate

Features Some attributes denote the percentage of connections with errors, like those which are rejected or otherwise failed. These contribute in spotting suspicious activities that might point to probing or intrusion.

In addition to these 41 attributes, each record contains a class label that classifies the connection as either normal or one of several attack types. These attacks are usually classified into categories like denial-of-service (DoS), probing, user-to-root (U2R) and remote-to-local (R2L). On the whole, the KDD Cup 1999 dataset has a rich and diverse set of features that illustrate anomalous behavior in a variety of ways, and therefore it is widely used as a benchmark dataset for developing and testing IDSs.

Table 3.2 KDD Cup 1999 attributes and result

Attributes	Example 1	Example 2	Example 3	Example 4
H1	0	0	0	0
H2	Udp	tcp	tcp	Tcp
H3	other	http	finger	Private
H4	SF	SF	S0	S0
H5	146	232	0	0
H6	0	8153	0	0
H7	0	0	0	0
H8	0	0	0	0
H9	0	0	0	0
H10	0	0	0	0
H11	0	0	0	0
H12	0	1	0	0
H13	0	0	0	0
H14	0	0	0	0
H15	0	0	0	0
H16	0	0	0	0
H17	0	0	0	0
H18	0	0	0	0
H19	0	0	0	0
H20	0	0	0	0
H21	0	0	0	0
H22	0	0	0	0
H23	13	5	0	48
H24	1	5	24	16
H25	0.00	0.20	12	1.00
H26	0.00	0.20	1.00	1.00
H27	0.00	0.00	1.00	0.00
H28	0.00	0.00	0.00	0.00
H29	0.08	1.00	0.00	0.14
H30	0.15	0.00	0.50	0.06
H31	0.00	0.00	0.00	0.00
H32	255	255	255	255
H33	1	30	59	15
H34	0.00	1.00	0.23	0.06
H35	0.60	0.00	0.04	0.07
H36	0.88	0.03	0.00	0.00
H37	0.00	0.04	0.00	0.00
H38	0.00	0.03	1.00	1.00
H39	0.00	0.01	1.00	1.00
H40	0.00	0.00	0.00	0.00
H41	0.00	0.01	0.00	0.00
Class	Normal	Normal	Anomaly	Anomaly

Journal of Chemical Health Risks
www.jchr.org
JCHR (2026) 16(2), 2778 - 2794

When it comes to detecting anomalous behavior, the 41 attributes are crucial in determining the presence of any malicious activity. However, the challenge lies in detecting these behaviors efficiently without increasing the false alarm rate. One approach to increase efficiency is to reduce the number of attributes, but it's important to keep in mind the ultimate goal of accurate detection. These 41 attributes are used to identify anomalous behavior by comparing values to the mean, with deviations from the mean indicating potential anomalies. One way to address the false positive problem in IDS is to eliminate the count attribute, but the challenge remains to detect attacks in a timely manner. Ultimately, finding the right balance between efficiency and accuracy is key to effective intrusion detection.

IV. Experiment and results

The count attribute has been identified as a factor causing an increase in false-positive rates. The primary focus for system upgrades is on two performance factors: detection rate and false alarm rate. The false alarm rate is calculated by dividing the total number of normal instances identified as an intrusion by the total number of normal instances, while the detection rate is calculated by dividing the number of correctly identified intrusion patterns by the total number of intrusion patterns in the dataset. The number of mistakes made in detection and the percentage of intrusions the system can accurately detect are the two indicators of successful execution. To measure performance, these values can be calculated on sample data. To improve authentication problems, the count attribute can be removed and replaced with a one-time password (OTP) sent to the user's email or phone number. OTP is a reliable solution to this problem.

Algorithm: OTP Generation After 10 Wrong Password Attempts

Step 1: Initialize Variables

- Set failed_attempts = 0
- max_attempts = 10
- otp = null otp_validity_time (e.g., 5 minutes)

Step 2: User Login Process User

- Enters username and password.
- The entered credentials are checked against saved ones by the system.

Step 3: Check Credentials

- If the password is correct: Access is granted to the user.
- Reset failed_attempts = 0.
- Else:
  Increment failed_attempts by 1.

Step 4: Check Failed

- Attempts Limit If failed_attempts < max_attempts: Print (error): "Invalid password, try again."
- If failed_attempts == max_attempt: Initiate OTP generation.

Step 5: OTP Generation

- Generate a random OTP (e.g. , 6-digit number).
- Keep OTP in secure way in the system. Note the timestamp of OTP generation.

Step 6: Send OTP

- Send the OTP to user on registered email or mobile number.

Step 7: OTP Verification

- Ask the user to input the OTP he received.
- Check: Is entered OTP equal to stored OTP?
- If now < generated_time + otp_validity_time then

Step 8: Result of OTP

- Validation If OTP is correct and valid: After the user has been verified, s/he can reset password or perform login.

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

- Reset failed_attempts = 0.
- Else:
  Display error "Invalid or expired OTP".
  delete limit_retries=0; restart limit_retries=limit_retries-1; // Optional: Allow limited retry attempts.

Step 9: Security Handling

When the OTP attempts exceed the limit: Temporarily lock the account or require admin verification.

Step 10: End Process

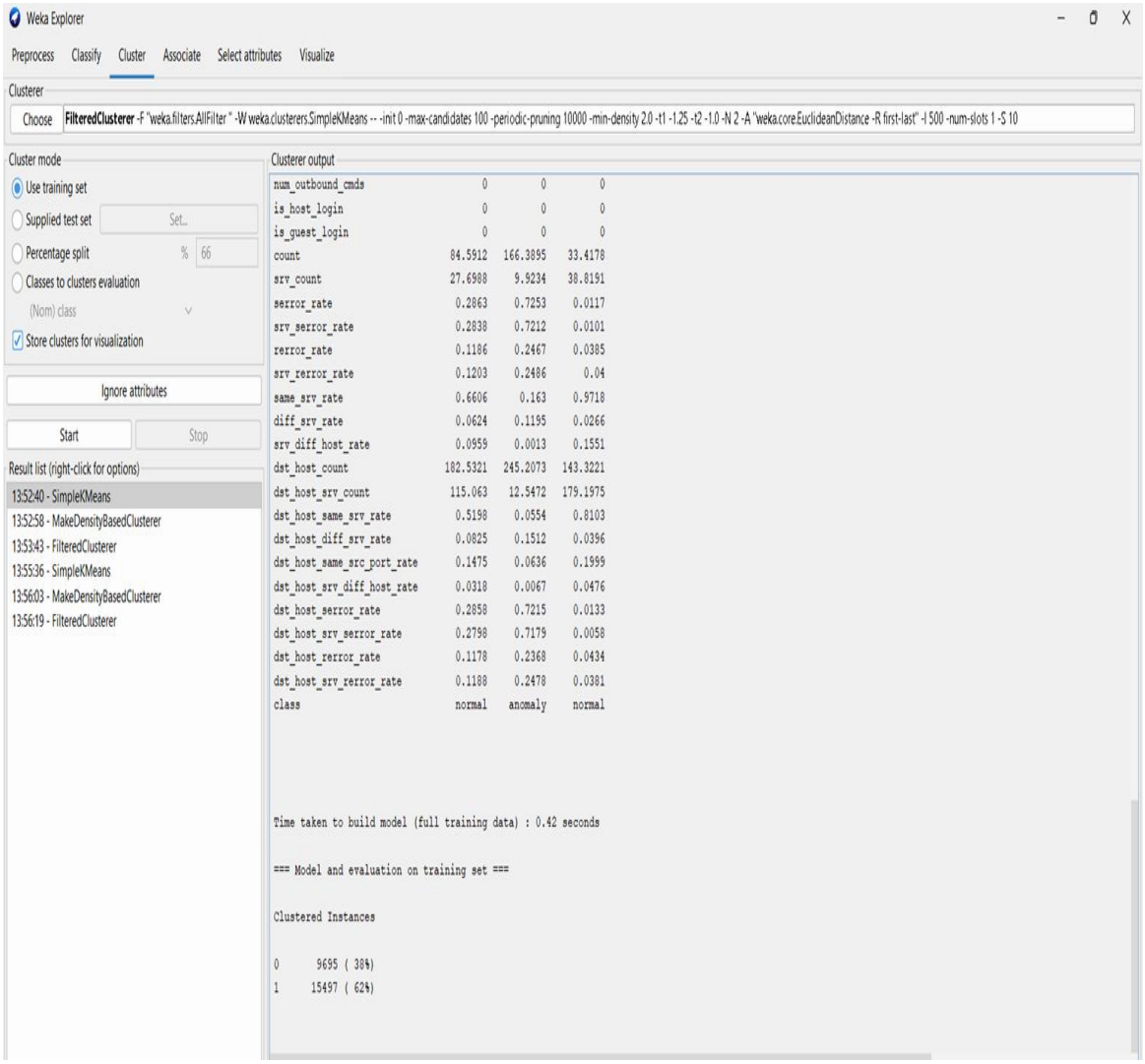


Figure 4.1. The outcome of an experiment using a Simple K-Means Clustering algorithm (including count attribute)

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

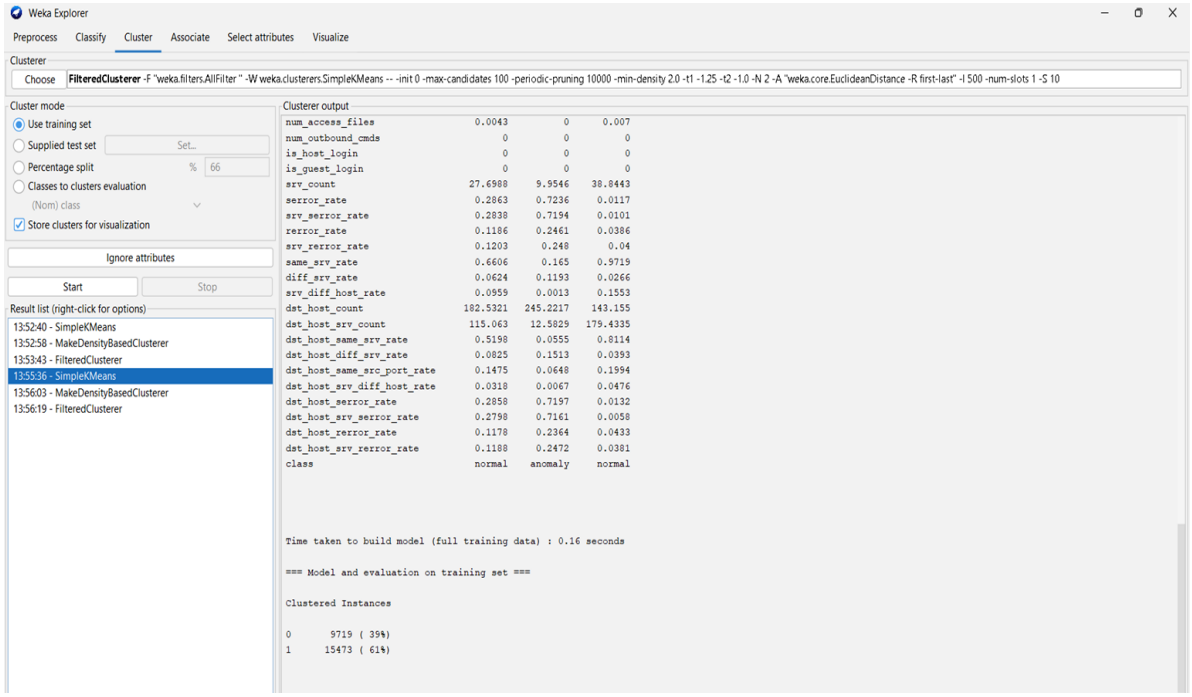
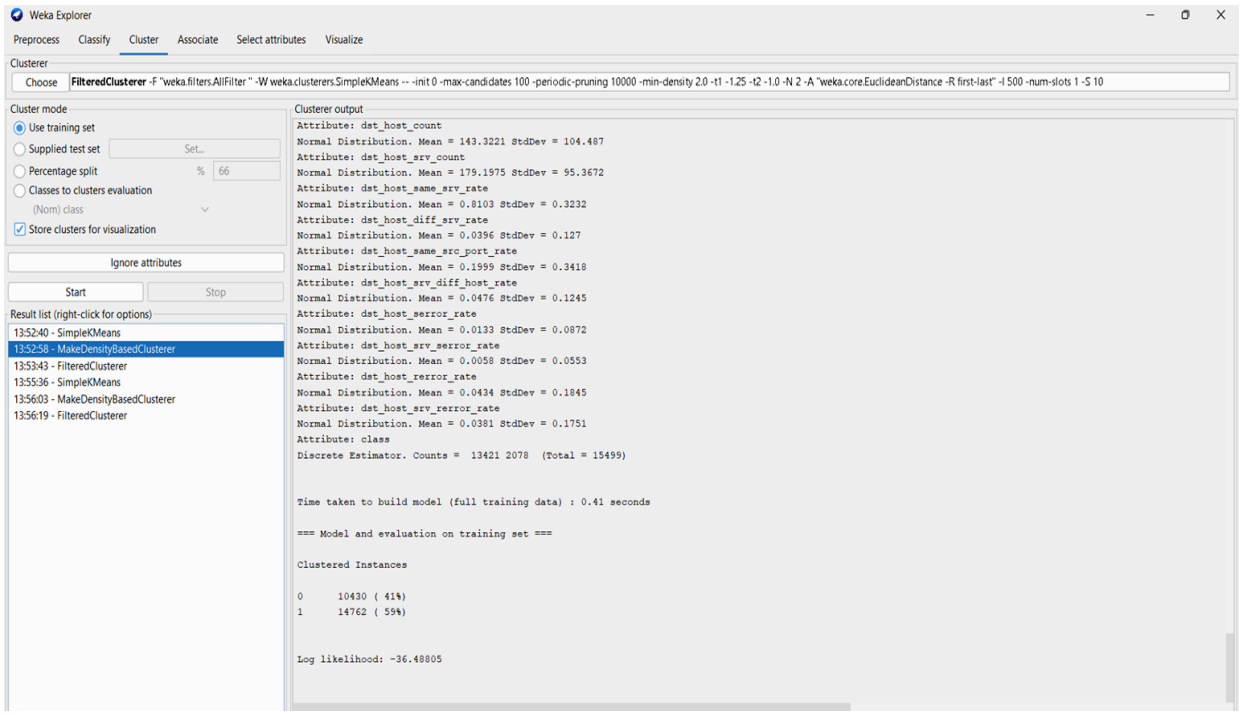


Figure 4.2. The result of an experimentation using a Simple K-Means Clustering algorithm (without count attribute)



Journal of Chemical Health Risks
www.jchr.org
JCHR (2026) 16(2), 2778 - 2794

Figure 4.3. The result of an experimentation using the Make Density-Based Clustering algorithm (including count attribute)

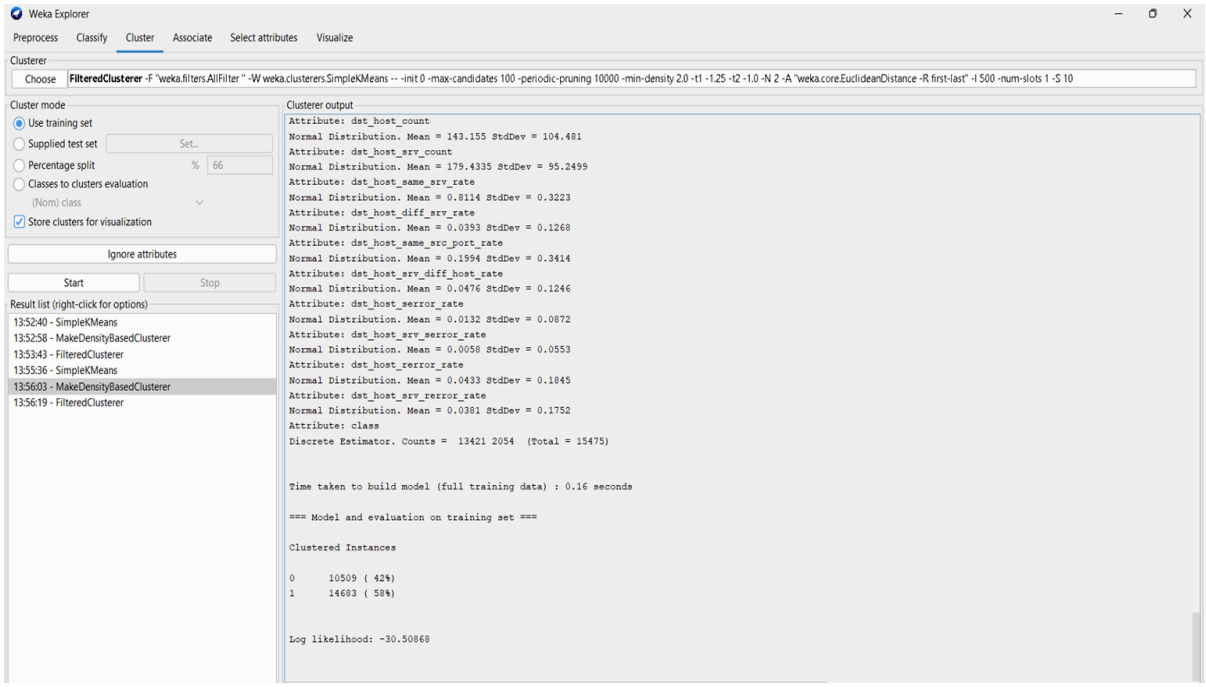


Figure 4.4. The result of an experimentation using the Make Density-Based Clustering algorithm (removing the count attribute)

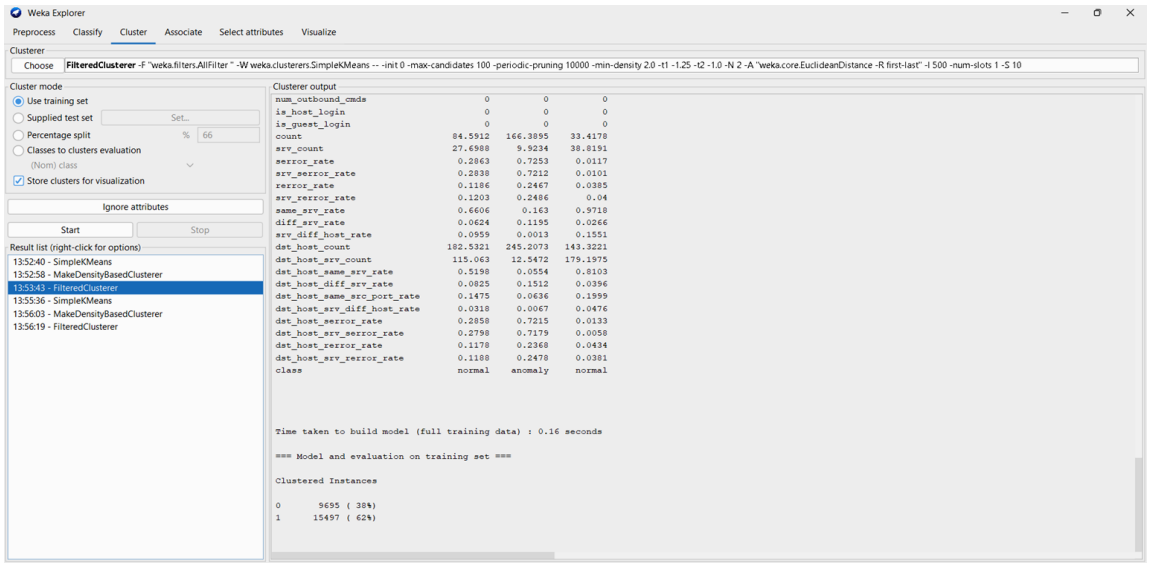


Figure 4.5. The result of an experimentation using the Filter Clusterer Clustering algorithm (including the count attribute)

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

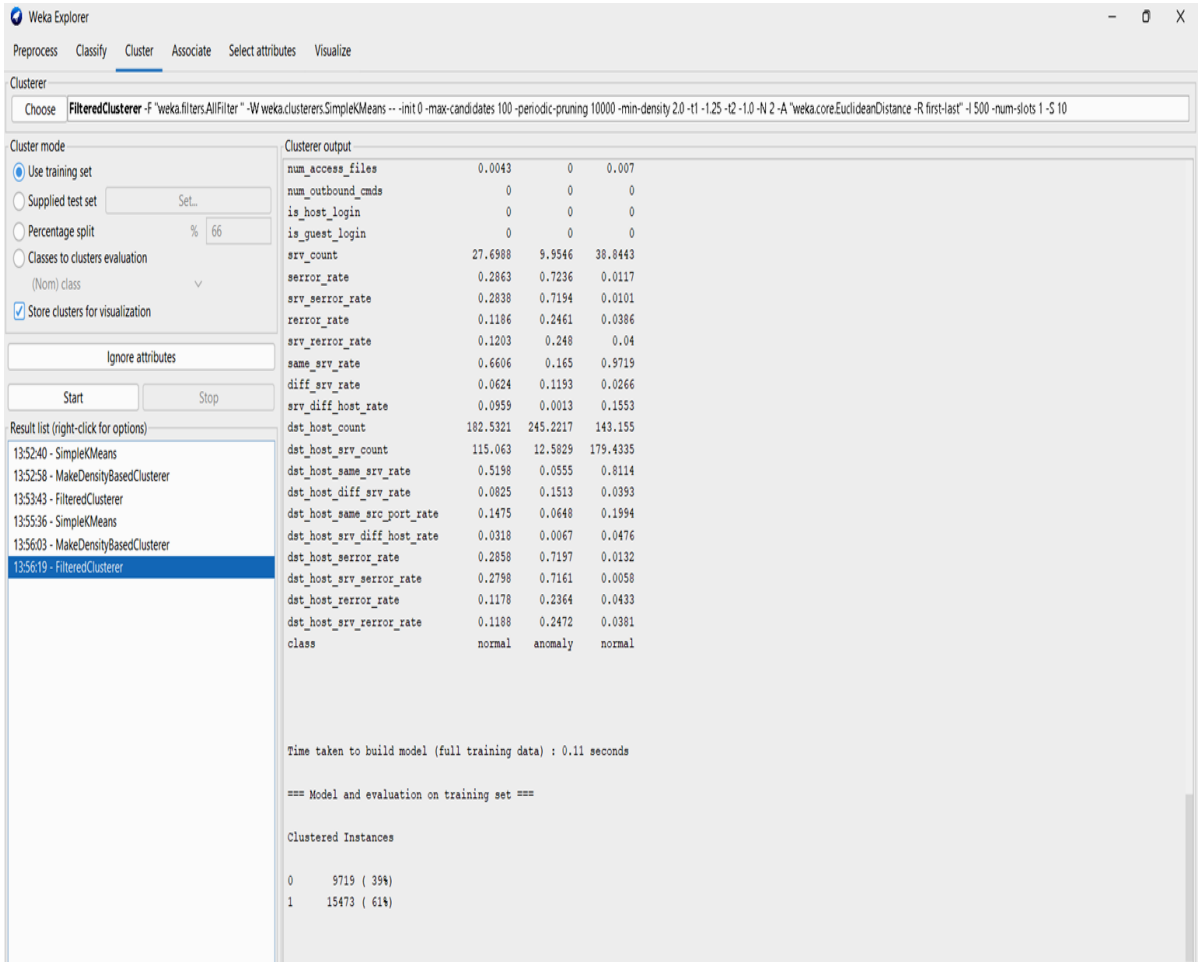


Figure 4.6. The result of an experimentation using the Clustering algorithm (removing the count attribute)

The results of six different scenarios using two clustering algorithms are presented in Figures 4.1-4.6. Figure 4.1 shows the output of the Simple K-mean clustering algorithm with the count attribute included, indicating a clustering time of 0.42 seconds. Figure 4.2 displays the output of the same algorithm but with the count attribute excluded, resulting in a reduced clustering time of 0.26 seconds. In contrast, Figures 4.3 and 4.4 depict the Make Density-Based Clustering algorithm with and without the count attribute, respectively. Figure 4.3 shows a clustering time of 0.41 seconds, while Figure 4 shows a reduced clustering time of 0.25 seconds. . Now, Figures 4.5 and 4.6 depict the filter Clustering algorithm with and without the count attribute, respectively. Figure 4.5 shows a clustering time of 0.16 seconds, while Figure 4.6 shows a reduced clustering time of 0.05 seconds. These results suggest that excluding the count attribute from the clustering algorithms can significantly reduce the time taken to complete the clustering process.

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

Table 4.1: Final cluster centroids of simpleK-mean with count attribute:

Attribute	Full data(25192.0)	cluster#0 (9695.0)	1 (15497.0)
H1	305.0541	533.1584	162.3509
H2	tcp	Tcp	tcp
H3	http	Private	http
H4	SF	S0	SF
H5	24330.6282	39374.1009	14919.3572
H6	3491.8472	115.1045	5604.3541
H7	0	0	0
H8	0.0237	0.0175	0.0276
H9	0	0	0.0001
H10	0.198	0.0018	0.3208
H11	0.0012	0.0002	0.0018
H12	0	0	1
H13	0.2279	0	0.3704
H14	0.0015	0.0001	0.0025
H15	0.0013	0.0002	0.0021
H16	0.2498	0.0005	0.4058
H17	0.0147	0.001	0.0233
H18	0.0004	0	0.0006
H19	0.0043	0	0.007
H20	0	0	0
H21	0	0	0
H22	0	0	0
H23	84.5912	166.3895	33.4178
H24	27.6988	9.9234	38.8191
H25	0.2863	0.7253	0.0117
H26	0.2838	0.7212	0.0101
H27	0.1186	0.2467	0.0385
H28	0.1203	0.2486	0.04
H29	0.6606	0.163	0.9718
H30	0.0624	0.1195	0.0266
H31	0.0959	0.0013	0.1551
H32	182.5321	245.2073	143.3221
H33	115.063	12.5472	179.1975
H34	0.5198	0.0554	0.8103
H35	0.0825	0.1512	0.0396
H36	0.1475	0.0636	0.1999
H37	0.0318	0.0067	0.0476
H38	0.2858	0.7215	0.0133
H39	0.2798	0.7179	0.0058
H40	0.1178	0.2368	0.0434
H41	0.1188	0.2478	0.0381
Class	normal	Anomaly	normal

Time taken to build model (full training data) : 0.42 seconds

Model and evaluation on training set
Clustered Instances
0 9695 (38%)
1 15497 (62%)

Table 4.2: Final cluster centroids of simpleK-mean without count attribute:

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

Table 4.3: Final cluster centroids of Filter Clusterer with count attribute:

Attribute	Full data(25192.0)	cluster#0 (9695.0)	1 (15497.0)
H1	305.0541	533.1584	162.3509
H2	Tcp	Tcp	tcp
H3	http	Private	http
H4	SF	S0	SF
H5	24330.6282	39374.1000	14919.37572
H6	3491.8472	115.1045	5604.3541
H7	0	0	0
H8	0.0237	0.0175	0.0275
H9	0	0	0.0010
H10	0.198	0.0018	0.3208
H11	0.0012	0.0002	0.0018
H12	0	0	1
H13	0.2279	0	0.3704
H14	0.0015	0.0001	0.0025
H15	0.0013	0.0002	0.0021
H16	0.2498	0.0005	0.4058
H17	0.0147	0.001	0.0233
H18	0.0004	0	0.0006
H19	0.0043	0	0.007
H20	0	0	0
H21	0	0	0
H22	0	0	0
H23	84.5912	166.3895	33.4178
H24	27.6988	9.9234	38.8191
H25	0.2863	0.7253	0.0117
H26	0.2838	0.7212	0.0101
H27	0.1186	0.2467	0.0385
H28	0.1203	0.2486	0.04
H29	0.6606	0.163	0.9718
H30	0.0624	0.1195	0.0266
H31	0.0959	0.0013	0.1551
H32	182.5321	245.2073	143.3221
H33	115.063	12.5472	179.1975
H34	0.5198	0.0554	0.8103
H35	0.0825	0.1512	0.0396
H36	0.1475	0.0636	0.1999
H37	0.0318	0.0067	0.0476
H38	0.2858	0.7215	0.0133
H39	0.2798	0.7179	0.0058
H40	0.1178	0.2368	0.0434
H41	0.1188	0.2478	0.0381
Class	Normal	Anomaly	normal
Time is taken to build the model (full training data) : 0.16 seconds			
=== Model and evaluation on training set === Clustered Instances 0 9695 (38%) 1 15497 (62%)			

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

Table 4.4: Final cluster centroids of Filter Clusterer without count attribute: Table 4.5: Final cluster centroids of make density with count attribute:

Attribute	Full data (25192.0)	cluster#0 (9695.0)	1 (15497.0)
H1	305.0541	533.1584	162.3509
H2	Tcp	Tcp	Tcp
H3	http	Private	http
H4	SF	S0	SF
H5	24330.6282	39374.1009	14919.3572
H6	3491.8472	115.1045	5604.3541
H7	0	0	0
H8	0.0237	0.0175	0.0276
H9	0	0	0.0001
H10	0.198	0.0018	0.3208
H11	0.0012	0.0002	0.0018
H12	0	0	1
H13	0.2279	0	0.3704
H14	0.0015	0.0001	0.0025
H15	0.0013	0.0002	0.0021
H16	0.2498	0.0005	0.4058
H17	0.0147	0.001	0.0233
H18	0.0004	0	0.0006
H19	0.0043	0	0.007
H20	0	0	0
H21	0	0	0
H22	0	0	0
H23	84.5912	166.3895	33.4178
H24	27.6988	9.9234	38.8191
H25	0.2863	0.7253	0.0117
H26	0.2838	0.7212	0.0101
H27	0.1186	0.2467	0.0385
H28	0.1203	0.2486	0.04
H29	0.6606	0.163	0.9718
H30	0.0624	0.1195	0.0266
H31	0.0959	0.0013	0.1551
H32	182.5321	245.2073	143.3221
H33	115.063	12.5472	179.1975
H34	0.5198	0.0554	0.8103
H35	0.0825	0.1512	0.0396
H36	0.1475	0.0636	0.1999
H37	0.0318	0.0067	0.0476
H38	0.2858	0.7215	0.0133
H39	0.2798	0.7179	0.0058
H40	0.1178	0.2368	0.0434
H41	0.1188	0.2478	0.0381
Class	Normal	anomaly	Normal

Time is taken to build the model (full training data) : 0.41 seconds
=== Model and evaluation on training set ===
Clustered Instances
0 10430 (41%)
1 14762 (59%)

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

Table 4.6: Final cluster centroids of make density without count attribute:

Attribute	Full data(25192.0)	cluster#0 (9719.0)	1 (15473.0)
H1	305.0541	531.8419	162.6027
H2	Tcp	Tcp	Tcp
H3	http	Private	http
H4	SF	S0	SF
H5	24330.6282	39277.0561	14942.3821
H6	3491.8472	114.8202	5613.047
H7	0	0	0
H8	0.0237	0.021	0.0255
H9	0	0	0.0001
H10	0.198	0.0017	0.3213
H11	0.0012	0.0002	0.0018
H12	0	0	1
H13	0.2279	0	0.371
H14	0.0015	0.0001	0.0025
H15	0.0013	0.0002	0.0021
H16	0.2498	0.0005	0.4064
H17	0.0147	0.001	0.0233
H18	0.0004	0	0.0006
H19	0.0043	0	0.007
H20	0	0	0
H21	0	0	0
H22	0	0	0
H23	27.6988	9.9546	38.8443
H24	0.2863	0.7236	0.0117
H25	0.2838	0.7194	0.0101
H26	0.1186	0.2461	0.0386
H27	0.1203	0.248	0.04
H28	0.6606	0.165	0.9719
H29	0.0624	0.1193	0.0266
H30	0.0959	0.0013	0.1553
H31	182.5321	245.2217	143.115
H32	115.063	12.5829	179.4335
H33	0.5198	0.0555	0.8114
H34	0.0825	0.1513	0.0393
H35	0.1475	0.0648	0.1994
H36	0.0318	0.0067	0.0476
H37	0.2858	0.7197	0.0132
H38	0.2798	0.7161	0.0058
H39	0.1178	0.2364	0.0433
H40	0.1188	0.2472	0.0381
Class	Normal	anomaly	Normal
Time is taken to build the model (full training data) : 0.16 seconds			
=== Model and evaluation on training set === Clustered Instances 0 10509 (42%) 1 14683 (58%)			

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

Table 4.7: Comparison of Outcome of filter clustering algorithm, Make Density-Based Clustering algorithm & Simple K-mean algorithm.

Time taken	filter cluster	Make Density-Based Clustering	Simple K-mean Clustering
Including count attribute	0.16 seconds	0.41 seconds	0.42 seconds
Excluding count attribute	0.11 seconds	0.16 seconds	0.16 seconds
Efficiency improved	31.25%	60.97%	61.90%

V CONCLUSION

In this study, we introduced an improved intrusion detection method by pulling out the count attribute from the feature set of IDS. The motivation for dimensionality reduction was that it might lead to enhanced efficiency of the system without decreasing the detection performance. This implies that the removal of the count attribute from the data reduces computational complexity and thereby allows the system to execute faster with a more responsive system. On the other hand, the results of the experiments also show that there is a significant decrease of false alarm rate which is a very important factor for the real implementation of the IDS. Removing redundant or less informative features makes the model less vulnerable to over-fitting and misclassification, and hence more accurate and reliable. The proposed solution also underlines the "selection of features" important in efficient and scalable IDS designs. Rather than using all the features, selecting the relevant features can yields better generalization results and computational cost. To sum up, eliminating the count attribute is a simple but efficient way to increase the fidelity of IDS. Future work includes investigating the use of sophisticated feature selection methods, and the hybridization of ML techniques for enhancing its detection capabilities and adaptive in evolving network scenarios. These days, many people maintain accounts across multiple platforms, which makes it difficult for them to keep track of all their passwords. This often results in multiple attempts to access an account, and in the case of banks, after three unsuccessful attempts, the bank website blocks the user's account for 24 hours. To address this issue, this paper proposes excluding the count attribute to improve system performance. Table 6 shows the positive impact of this exclusion on performance, with the filter clustered value decreasing from 0.16 to 0.11 with improving efficiency up to 31.25% and the make density value decreasing from 0.41 to 0.16 with improving efficiency up to 60.97% and simple K-mean value decreasing from 0.42 to 0.16 with improving efficiency up to 61.90%. These improvements increase effective and reducing the time required for detection, leading to a reduction in the false alarm rate. This improvement can help enhancing the security, as faster detection reduces the likelihood of a breach. Delayed detection can cause harm to the system, making quick detection and response crucial for maintaining.

REFERENCE

[1] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Dept. of Computer Engineering, Chalmers Univ., 2000.

[2] D. E. Denning, "An intrusion-detection model," IEEE Trans. Software Eng., vol. 13, no. 2, pp. 222–232, Feb. 1987.

[3] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems," NIST Special Publication, 2007.

[4] R. Bace and P. Mell, "Intrusion detection systems," NIST, 2001.

[5] W. Stallings and L. Brown, Computer Security: Principles and Practice, Pearson, 2018.

[6] M. Roesch, "Snort: Lightweight intrusion detection for networks," in Proc. LISA, 1999, pp. 229–238.

[7] S. Northcutt, Network Intrusion Detection, New Riders, 2002.

[8] J. McHugh, "Testing intrusion detection systems," ACM Trans. Inf. Syst. Secur., vol. 3, no. 4, pp. 262–294, 2000.

Journal of Chemical Health Risks

www.jchr.org

JCHR (2026) 16(2), 2778 - 2794

[9] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion detection systems," Comput. Netw., vol. 31, pp. 805–822, 1999.

[10] T. Lunt, "A survey of intrusion detection techniques," Computers & Security, vol. 12, no. 4, pp. 405–418, 1993.

[11] G. Vigna and R. Kemmerer, "NetSTAT: A network-based intrusion detection system," J. Comput. Security, 1999.

[12] S. Forrest et al., "A sense of self for UNIX processes," in Proc. IEEE Symp. Security and Privacy, 1996.

[13] K. Kendall, "A database of computer attacks for IDS evaluation," MIT, 1999.

[14] R. Sommer and V. Paxson, "Outside the closed world: Machine learning for IDS," in Proc. IEEE S&P, 2010.

[15] G. Creech and J. Hu, "A semantic approach to host-based IDS," IEEE Trans. Comput., 2014.

[16] Y. Liao and V. Vemuri, "Use of k-NN classifier for IDS," Computers & Security, vol. 21, no. 5, 2002.

[17] W. Lee and S. Stolfo, "A framework for constructing features for IDS," ACM Trans. Inf. Syst. Secur., 2000.

[18] W. Lee et al., "Data mining approaches for intrusion detection," in USENIX Security Symposium, 1998.

[19] A. Patcha and J. Park, "An overview of anomaly detection techniques," Comput. Netw., 2007.

[20] N. Ye et al., "Probabilistic techniques for IDS," IEEE Trans. Syst., Man, Cybern., 2002.

[21] J. Cannady, "Artificial neural networks for misuse detection," Natl. Inf. Syst. Security Conf., 1998.

[22] M. Tavallaee et al., "A detailed analysis of KDD CUP 99 dataset," in IEEE CISDA, 2009.

[23] I. Sharafaldin et al., "Toward generating a new IDS dataset (CICIDS2017)," ICISSP, 2018.

[24] C. Kruegel et al., "Anomaly detection of web-based attacks," ACM CCS, 2003.

[25] S. Mukkamala et al., "Intrusion detection using neural networks and SVM," IEEE IJCNN, 2002.

[26] J. Zhang et al., "Random forests for intrusion detection," IEEE Trans. Syst., 2008.

[27] K. Wang and S. Stolfo, "Anomalous payload-based network IDS," RAID, 2004.

[28] R. Sekar et al., "Specification-based anomaly detection," ACM CCS, 2002.

[29] Y. Freund and R. Schapire, "A decision-theoretic generalization of boosting," J. Comput. Syst. Sci., 1997.

[30] T. Kim et al., "Long short-term memory for IDS," IEEE Access, 2016.

[31] A. Javaid et al., "Deep learning for network intrusion detection," IEEE ICST, 2016.

[32] W. Wang et al., "HAST-IDS: Learning hierarchical spatial-temporal features," IEEE Access, 2018.

[33] X. Yin et al., "Deep learning approach for intrusion detection using RNN," IEEE Access, 2017.

[34] Y. Zhou et al., "Machine learning-based IDS: A survey," IEEE Commun. Surveys, 2019.

[35] M. Ring et al., "Survey of network-based IDS datasets," Computers & Security, 2019.

[36] H. Hindy et al., "A taxonomy of IDS systems," Future Generation Computer Systems, 2020.

[37] A. Khraisat et al., "Survey of intrusion detection systems," IEEE Access, 2019.

[38] S. Dua and X. Du, Data Mining and Machine Learning in Cybersecurity, CRC Press, 2016.

[39] J. Behl and B. Behl, "Cybersecurity and cyberwar," Oxford Press, 2017.

[40] P. García-Teodoro et al., "Anomaly-based network IDS," Computers & Security, 2009.

[41] M. Bishop, Computer Security: Art and Science, Addison-Wesley, 2003.

[42] S. Axelsson, "The base-rate fallacy in intrusion detection," ACM CCS, 1999.

[43] J. Novak and S. McHugh, "Testing IDS accuracy," IEEE Security & Privacy, 2004.

[44] G. Gu et al., "BotHunter: Detecting malware infection," USENIX Security, 2007.

[45] K. Julisch, "Clustering intrusion detection alarms," ACM CCS, 2003.