

Machine Learning-Based Intrusion Detection System (IDS) for Classifying Types of Attacks on Computer Networks

Ryci Rahmatil Fiska^{1,a)}, Wahyat^{1,b)}, Dedi Hermawan¹⁾, Via Laurenz¹⁾, Izatul Fateha¹⁾

¹Politeknik Negeri Bengkalis, Bengkalis, Indonesia

^{a)}Corresponding author: rycirf@polbeng.ac.id

^{b)}wahyat@polbeng.ac.id

Abstract. Server security on a computer network is very important, maintaining the security of a computer network in order to maintain information, data and maintain infrastructure so that it can work and function properly and provide access rights to registered users, this research, aims to build an IDS (Intrusion Detection System) on the network and Server using Raspberry Pi with SNORT which is useful for monitoring Server activity when an attempted attack occurs. With the increasing complexity of network attacks carried out by attackers, intelligent and adaptive approaches are needed to detect and overcome these threats. Traditional methods such as rule-based or signatures are often not effective enough in the face of evolving attacks. The large amount of network traffic data makes it difficult to manually analyze and detect attacks. Naive Bayes has a very important role in the classification and detection of network attacks, both considered malicious and highly malicious, By using Naive Bayes, network security systems can become more proactive and adaptive to attacks. This technology not only helps in detecting familiar attacks but also enables identification and response to new or unknown attack techniques. Through proper classification, the system can provide better protection and reduce the impact of attacks.

Keywords: Naïve bayes, raspberry Pi, server, IDS, snort

INTRODUCTION

Computer network security has become one of the top priorities in the rapidly developing digital era. With the increasing cyber threats such as malware, denial of service (DoS), phishing, and zero-day attacks, the need for a system that is able to detect and respond to attacks quickly and accurately is very important. One of the main technologies used to address this problem is the Intrusion Detection System (IDS), which is designed to monitor and analyze network traffic to detect suspicious activity or attacks. Traditional IDS, which are generally signature-based or rule-based, are often not effective enough in dealing with new or previously unknown attacks. To overcome this deficiency, machine learning (ML)-based approaches have been introduced. One of the simplest yet effective ML algorithms in data classification is Naive Bayes.

Naive Bayes is a probabilistic machine learning algorithm based on Bayes' Theorem with the assumption of conditional independence between features. This algorithm has several characteristics that make it suitable for application in IDS, including:

Simplicity and Speed: Naive Bayes is a relatively simple and fast algorithm in performing computations. In a network environment that requires rapid detection of potential attacks, computational efficiency is very important.

Multiclass Data Processing Ability: In a computer network, traffic can involve various types of different attacks. Naive Bayes is very effective in handling multiclass classification tasks, which are useful in detecting various types of attacks at once.

Handling Large Datasets: Network attacks often occur in large volumes of data, and Naive Bayes is able to handle large datasets with high efficiency due to its simple probabilistic calculation scheme.

Ability in Anomaly Detection: IDS often has to detect rare attacks or new attacks that have not been previously known (zero-day attacks). The Naive Bayes algorithm, especially in anomaly detection mode, can help recognize unusual activity based on the probability distribution of the trained data.

METHODS

The research stage consists of several steps, starting with collecting the data needed to create a Machine Learning design. At this stage, identification of relevant data for research is carried out, as well as determining functional and non-functional needs, such as the objectives to be achieved by the system and the desired performance constraints. After the data is collected and the needs are determined, the next step is to create a good Machine Learning model design, including the selection of appropriate algorithms and methods. This design is then implemented, resulting in a Machine Learning model that is able to classify according to the research objectives using one of the Machine Learning models, namely Naïve Bayes. The final step is to create documentation and reports on the research results.

RESULTS AND DISCUSSION

The results of the research conducted in the form of an Intrusion Detection System (IDS) prototype at the Computer Network Laboratory of the Bengkalis State Polytechnic. There are 3 Intrusion Detection System (IDS) test scenarios, namely PING Attack, Port Scanning, and DOS/DDoS Attack using the Kali Linux operating system.

Test scenario 1 is ping Attack using Kali Linux with the command Address "ping 192.168.40.254", test scenario 2 is Port Scanning using Kali Linux with the command "nmap -sF 192.168.40.254", test scenario 3 is DOS / DDoS Attack using Kali Linux with the command "hping3 -S -p 80 192.168.40.254", the results of the test command scenario 1 ping attack can be shown in figure 5.2, the results of the test command scenario 2 port scanning can be shown in figure 5.3 and the results of the test command scenario 3 DOS / DDOS Attack in the image below:



```
kali@kali:~$ ping 192.168.40.254
PING 192.168.40.254 (192.168.40.254): 64 bytes of data:
64 bytes from 192.168.40.254: icmp_seq=1 ttl=64 time=0.118 ms
64 bytes from 192.168.40.254: icmp_seq=2 ttl=64 time=0.112 ms
64 bytes from 192.168.40.254: icmp_seq=3 ttl=64 time=0.143 ms
64 bytes from 192.168.40.254: icmp_seq=4 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=5 ttl=64 time=0.119 ms
64 bytes from 192.168.40.254: icmp_seq=6 ttl=64 time=0.121 ms
64 bytes from 192.168.40.254: icmp_seq=7 ttl=64 time=0.119 ms
64 bytes from 192.168.40.254: icmp_seq=8 ttl=64 time=0.118 ms
64 bytes from 192.168.40.254: icmp_seq=9 ttl=64 time=0.122 ms
64 bytes from 192.168.40.254: icmp_seq=10 ttl=64 time=0.126 ms
64 bytes from 192.168.40.254: icmp_seq=11 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=12 ttl=64 time=0.128 ms
64 bytes from 192.168.40.254: icmp_seq=13 ttl=64 time=0.144 ms
64 bytes from 192.168.40.254: icmp_seq=14 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=15 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=16 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=17 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=18 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=19 ttl=64 time=0.145 ms
64 bytes from 192.168.40.254: icmp_seq=20 ttl=64 time=0.145 ms
```



```
kali@kali:~$ nmap -sF 192.168.40.254
Starting Nmap 7.60 ( https://nmap.org ) at 2023-08-07 20:22 WIB
Nmap scan report for 192.168.40.254 (192.168.40.254)
Host is up (0.0011s latency).
Not shown: 998 closed tcp ports (reset)
port      state
|_
|_ http open|filtered
|_ http open|filtered
Nmap done: 1 IP address (1 host up) scanned in 1.45 seconds
kali@kali:~$
```

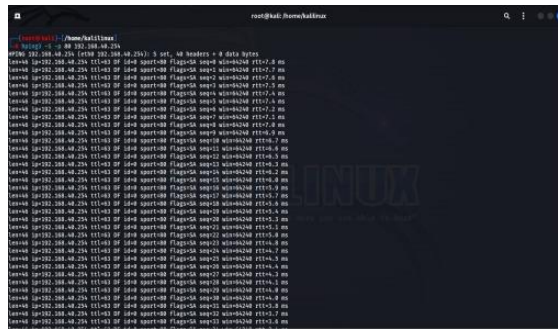


FIGURE 1. monitor types of attacks

Here are the results of 3 test scenarios, namely PING Attack, Port Scanning, and DOS/DDoS Attack, alerts or notifications of incoming attacks on Telegram BOT. The results of scenario 1 testing can be shown in figure 5.5, the results of scenario 2 testing can be shown in figure 5.6 and the results of scenario 3 testing can be shown in the figure below:



FIGURE 2. attack detection results

Types of Attacks in the Computer Network Lab

After detecting attacks in the computer network laboratory of the Bengkulu State Polytechnic, there are several types of attacks that often appear as in the table below:

TABLE 1. Labeling types of attacks

| Attack Type | Label | Amount |
|-----------------|----------------|--------|
| ICMP Ping | dangerous | 5 |
| ICMP Ping | very dangerous | 5 |
| SCAN FIN | dangerous | 6 |
| SCAN FIN | very dangerous | 6 |
| DDOS TCP Attack | dangerous | 5 |
| DDOS TCP Attack | very dangerous | 6 |
| Total | - | 33 |

Naïve Bayes classification

Calculating Dangerous and Very Dangerous probabilities of SCAN FIN attacks

Dangerous Probability

1. Prior probability:

- a. $P(\text{ICMP Ping}) = 10/33 = 0.30$
- b. $P(\text{SCAN FIN}) = 12/33 = 0.36$
- c. $P(\text{DDOS TCP Attack}) = 11/33 = 0.33$
- d. $P(\text{Dangerous}) = (\text{Number of Dangerous attacks}) / (\text{Total attacks}) = (5 + 6 + 5) / 33 = 0.48$
- e. $P(\text{Very Dangerous}) = (\text{Number of very Dangerous attacks}) / (\text{Total attacks}) = (5 + 6 + 6) / 33 = 0.5$

2. Conditional (Posterior) Probability:

- a. $P(\text{ICMP Ping} | \text{Dangerous}) = 5/16 = 0.31$
- b. $P(\text{ICMP Ping} | \text{Very Dangerous}) = 5/17 = 0.29$
- c. $P(\text{SCAN FIN} | \text{Dangerous}) = 6/16 = 0.38$
- d. $P(\text{SCAN FIN} | \text{Highly Dangerous}) = 6/17 = 0.35$
- e. $P(\text{DDOS TCP Attack} | \text{Dangerous}) = 5/16 = 0.31$
- f. $P(\text{DDOS TCP Attack} | \text{Highly Dangerous}) = 6/17 = 0.35$

3. Posterior Calculation (Using Bayes' Theorem):

Calculates the probability that an attack is Malicious or Very Dangerous based on the type of attack.

- a. Probability Dangerous | SCAN FIN

$$P(\text{Berbahaya} | \text{SCAN FIN}) = \frac{P(\text{SCAN FIN} | \text{Dangerous}) \times P(\text{Dangerous})}{P(\text{SCAN FIN})}$$

$$P(\text{Dangerous} | \text{SCAN FIN}) = \frac{0.38 \times 0.48}{0.36} = 0.51$$

- b. Probability Very Dangerous | SCAN FIN

$$P(\text{Very Dangerous} | \text{SCAN FIN}) = \frac{P(\text{SCAN FIN} | \text{Very Dangerous}) \times P(\text{Very Dangerous})}{P(\text{SCAN FIN})}$$

$$P(\text{Dangerous} | \text{SCAN FIN}) = \frac{0.38 \times 0.48}{0.36} = 0.51$$

From this calculation, the SCAN FIN attack has a slightly greater probability of being classified as Dangerous (0.51) than Very Dangerous (0.50).

ICMP Ping Calculation:

a. P(Dangerous |ICMP Ping):

$$P(\text{Dangerous} | \text{ICMP PING}) = \frac{P(\text{ICMPPING} | \text{Dangerous}) \times P(\text{Dangerous})}{P(\text{ICMP PING})}$$

$$P(\text{Dangerous} | \text{ICMP PING}) = \frac{0.31 \times 0.48}{0.30} = 0.50$$

b. P(Very Dangerous |ICMP PING):

$$P(\text{Very Dangerous} | \text{ICMP PING}) = \frac{P(\text{ICMPPING} | \text{Very Dangerous}) \times P(\text{Very Dangerous})}{P(\text{ICMP PING})}$$

$$P(\text{Very Dangerous} | \text{ICMP PING}) = \frac{0.29 \times 0.52}{0.30} = 0.50$$

The results are almost equal for ICMP Ping attacks as Malicious and Very Dangerous, both having a probability of 0.50.

DDOS TCP Attack Calculation:

a. P(Berbahaya|DDOSTCP Attack):

$$P(\text{Berbahaya} | \text{DDOSTCP Attack}) = \frac{P(\text{DDOSTCP Attack} | \text{Dangerous}) \times P(\text{Dangerous})}{P(\text{DDOSTCP Attack})}$$

$$P(\text{Dangerous} | \text{DDOSTCP Attack}) = \frac{0.31 \times 0.48}{0.33} = 0.45$$

For DDOS TCP Attack, there is a higher probability of being classified as Very Dangerous (0.55) than Dangerous (0.45).

1. Accuration

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Akurasi} = \frac{TP + TN}{TP + FN} = \frac{11 + 6}{11 + 6 + 0 + 5 + 11} = \frac{17}{33} \approx 0.5152(51.52)$$

2. Precision

$$Presisi = \frac{TP + TN}{TP + FP}$$

- Untuk Berbahaya:

$$Presisi (Dangerous) = \frac{11}{11 + 0} = 1(100\%)$$

- Untuk Sangat Berbahaya

$$Presisi (Very dangerous) = \frac{6}{6 + 0} = 1(100\%)$$

3. Recall

$$Recall = \frac{TP}{TP + FN}$$

- To be Dangerous

$$Recall(Dangerous) = \frac{11}{11 + 5} = \frac{11}{16} \approx 0.6875(68.75\%)$$

- To be very Dangerous

$$Recall(Very Dangerous) = \frac{6}{6 + 11} = \frac{6}{17} \approx 0.3529(35.29\%)$$

4. Calculating F1-Score

$$F1 - Score = 2 \times \frac{Presisi \times Recall}{Presisi + Recall}$$

- To be Dangerous

$$F1 - Score(Dangerous) = 2 \times \frac{1 \times 0.6875}{1 + 0.6875} = 2 \times \frac{0.6875}{1.6875} \approx 0.8148(81.48\%)$$

- To be Very Dangerous

$$F1 - Score(Very Dangerous) = 2 \times \frac{1 \times 0.3529}{1 + 0.3529} = 2 \times \frac{0.3529}{1.3529} \approx 0.5216(52.16\%)$$

Naïve Bayes Classification Using Google Colab

After the calculation is done manually, another calculation is done using Google Colab to make it easier to see the classification results based on the Naïve Bayes Model Evaluation graph.

```

# Definiskan nilai True Positives (TP), True Negatives (TN), False Positives (FP), dan False Negatives (FN)
TP_Berbahaya = 11 # True Positives untuk Berbahaya
TP_Sangat_Berbahaya = 6 # True Positives untuk Sangat Berbahaya
FN_Berbahaya = 5 # False Negatives untuk Berbahaya
FN_Sangat_Berbahaya = 11 # False Negatives untuk Sangat Berbahaya
TN = 0 # True Negatives (tidak ada dalam contoh ini)

# Hitung total untuk masing-masing label
TP_total = TP_Berbahaya + TP_Sangat_Berbahaya
FN_total = FN_Berbahaya + FN_Sangat_Berbahaya
FP_Berbahaya = 0 # False Positives untuk Berbahaya
FP_Sangat_Berbahaya = 0 # False Positives untuk Sangat Berbahaya

# Menghitung Akurasi
total_prediksi = TP_total + TN + FP_Berbahaya + FN_total
akurasi = (TP_total + TN) / total_prediksi if total_prediksi > 0 else 0


# Menghitung Presisi
presisi_Berbahaya = TP_Berbahaya / (TP_Berbahaya + FP_Berbahaya) if (TP_Berbahaya + FP_Berbahaya) > 0 else 0
presisi_Sangat_Berbahaya = TP_Sangat_Berbahaya / (TP_Sangat_Berbahaya + FP_Sangat_Berbahaya) if (TP_Sangat_Berbahaya + FP_Sangat_Berbahaya) > 0 else 0

# Menghitung Recall
recall_Berbahaya = TP_Berbahaya / (TP_Berbahaya + FN_Berbahaya) if (TP_Berbahaya + FN_Berbahaya) > 0 else 0
recall_Sangat_Berbahaya = TP_Sangat_Berbahaya / (TP_Sangat_Berbahaya + FN_Sangat_Berbahaya) if (TP_Sangat_Berbahaya + FN_Sangat_Berbahaya) > 0 else 0

# Menghitung F1-Score
f1_Berbahaya = (2 * presisi_Berbahaya * recall_Berbahaya) / (presisi_Berbahaya + recall_Berbahaya) if (presisi_Berbahaya + recall_Berbahaya) > 0 else 0
f1_Sangat_Berbahaya = (2 * presisi_Sangat_Berbahaya * recall_Sangat_Berbahaya) / (presisi_Sangat_Berbahaya + recall_Sangat_Berbahaya) if (presisi_Sangat_Berbahaya + recall_Sangat_Berbahaya) > 0 else 0

# Tampilkan hasil
print(f"Akurasi: {akurasi:.4f} ({akurasi * 100:.2f}%)")
print(f"Presisi Berbahaya: {presisi_Berbahaya:.4f} ({presisi_Berbahaya * 100:.2f}%)")
print(f"Presisi Sangat Berbahaya: {presisi_Sangat_Berbahaya:.4f} ({presisi_Sangat_Berbahaya * 100:.2f}%)")
print(f"Recall Berbahaya: {recall_Berbahaya:.4f} ({recall_Berbahaya * 100:.2f}%)")
print(f"Recall Sangat Berbahaya: {recall_Sangat_Berbahaya:.4f} ({recall_Sangat_Berbahaya * 100:.2f}%)")
print(f"F1-Score Berbahaya: {f1_Berbahaya:.4f}")
print(f"F1-Score Sangat Berbahaya: {f1_Sangat_Berbahaya:.4f}")

```


Akurasi: 0.5152 (51.52%)
Presisi Berbahaya: 1.0000 (100.00%)
Presisi Sangat Berbahaya: 1.0000 (100.00%)
Recall Berbahaya: 0.6875 (68.75%)
Recall Sangat Berbahaya: 0.3529 (35.29%)
F1-Score Berbahaya: 0.8148
F1-Score Sangat Berbahaya: 0.5217

```

import matplotlib.pyplot as plt

# Definiskan nilai True Positives (TP), True Negatives (TN), False Positives (FP), dan False Negatives (FN)
TP_Berbahaya = 11 # True Positives untuk Berbahaya
TP_Sangat_Berbahaya = 6 # True Positives untuk Sangat Berbahaya
FN_Berbahaya = 5 # False Negatives untuk Berbahaya
FN_Sangat_Berbahaya = 11 # False Negatives untuk Sangat Berbahaya
TN = 0 # True Negatives (tidak ada dalam contoh ini)

# Hitung total untuk masing-masing label
TP_total = TP_Berbahaya + TP_Sangat_Berbahaya
FN_total = FN_Berbahaya + FN_Sangat_Berbahaya
FP_Berbahaya = 0 # False Positives untuk Berbahaya
FP_Sangat_Berbahaya = 0 # False Positives untuk Sangat Berbahaya

# Menghitung Akurasi
total_prediksi = TP_total + TN + FP_Berbahaya + FN_total
akurasi = (TP_total + TN) / total_prediksi if total_prediksi > 0 else 0

# Menghitung Presisi
presisi_Berbahaya = TP_Berbahaya / (TP_Berbahaya + FP_Berbahaya) if (TP_Berbahaya + FP_Berbahaya) > 0 else 0
presisi_Sangat_Berbahaya = TP_Sangat_Berbahaya / (TP_Sangat_Berbahaya + FP_Sangat_Berbahaya) if (TP_Sangat_Berbahaya + FP_Sangat_Berbahaya) > 0 else 0

# Menghitung Recall
recall_Berbahaya = TP_Berbahaya / (TP_Berbahaya + FN_Berbahaya) if (TP_Berbahaya + FN_Berbahaya) > 0 else 0
recall_Sangat_Berbahaya = TP_Sangat_Berbahaya / (TP_Sangat_Berbahaya + FN_Sangat_Berbahaya) if (TP_Sangat_Berbahaya + FN_Sangat_Berbahaya) > 0 else 0

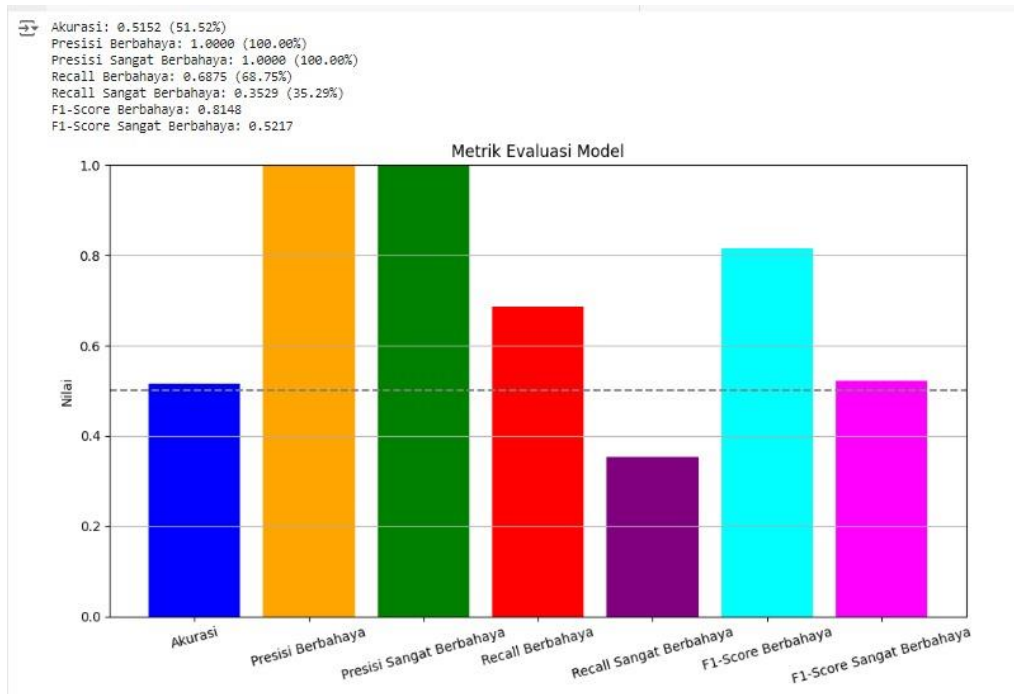
# Menghitung F1-Score
f1_Berbahaya = (2 * presisi_Berbahaya * recall_Berbahaya) / (presisi_Berbahaya + recall_Berbahaya) if (presisi_Berbahaya + recall_Berbahaya) > 0 else 0
f1_Sangat_Berbahaya = (2 * presisi_Sangat_Berbahaya * recall_Sangat_Berbahaya) / (presisi_Sangat_Berbahaya + recall_Sangat_Berbahaya) if (presisi_Sangat_Berbahaya + recall_Sangat_Berbahaya) > 0 else 0

# Tampilkan hasil
print(f"Akurasi: {akurasi:.4f} ({akurasi * 100:.2f}%)")
print(f"Presisi Berbahaya: {presisi_Berbahaya:.4f} ({presisi_Berbahaya * 100:.2f}%)")
print(f"Presisi Sangat Berbahaya: {presisi_Sangat_Berbahaya:.4f} ({presisi_Sangat_Berbahaya * 100:.2f}%)")
print(f"Recall Berbahaya: {recall_Berbahaya:.4f} ({recall_Berbahaya * 100:.2f}%)")
print(f"Recall Sangat Berbahaya: {recall_Sangat_Berbahaya:.4f} ({recall_Sangat_Berbahaya * 100:.2f}%)")
print(f"F1-Score Berbahaya: {f1_Berbahaya:.4f}")
print(f"F1-Score Sangat Berbahaya: {f1_Sangat_Berbahaya:.4f}")

# Visualisasi grafik
labels = ['Akurasi', 'Presisi Berbahaya', 'Presisi Sangat Berbahaya', 'Recall Berbahaya', 'Recall Sangat Berbahaya', 'F1-Score Berbahaya', 'F1-Score Sangat Berbahaya']
values = [akurasi, presisi_Berbahaya, presisi_Sangat_Berbahaya, recall_Berbahaya, recall_Sangat_Berbahaya, f1_Berbahaya, f1_Sangat_Berbahaya]

# Membuat grafik batang
plt.figure(figsize=(10, 6))
plt.bar(labels, values, color=['blue', 'orange', 'green', 'red', 'purple', 'cyan', 'magenta'])
plt.ylim(0, 1) # Set batas sumbu y antara 0 dan 1
plt.xlabel(yob.5, color='grey', linestyle='...') # Garis horizontal untuk referensi di 0.5
plt.title('Metrik Evaluasi Model')
plt.ylabel('Nilai')
plt.xticks(rotation=45)
plt.grid(axis='y')

```



CONCLUSIONS

From the results of the research conducted, it can be concluded that:

1. Testing is done with 3 scenarios, namely: PING Attack, Port Scanning, and DOS/DDoS Attack.
2. Naive Bayes often shows good performance in detecting common attacks, such as Denial of Service (DoS) or Probing attacks, which have clearer patterns in network traffic.
3. Overall the use of Naive Bayes for IDS has shown several advantages, such as decent accuracy, computational efficiency, and ease of implementation.

ACKNOWLEDGMENTS

Thank you for attention

REFERENCES

- [1] Anis, M., Hilmi, A., & Khujaemah, E. (2022). Network Security Monitoring with Intrusion Detection System. *Journal of Informatics Engineering (JUTIF)*, 3(2), 249-253. <https://doi.org/10.20884/1.jutif.2022.3.2.117>
- [2] A. T. Zy, A. T. Sasongko, and A. Z. Kamalia, "Application of Naïve Bayes Classifier, Support Vector Machine, and Decision Tree to Improve Network Security Threat Detection," *Online Media*, vol. 4, no. 1, pp. 610-617, 2023, doi: 10.30865/klik.v4i1.1134.
- [3] Ananda, D., & Suryono, R. R. (2024). Analysis of Public Sentiment towards Rohingya Refugees in Indonesia with Support Vector Machine and Naïve Bayes Methods. 8(April), 748-757. <https://doi.org/10.30865/mib.v8i2.7517>
- [4] Dwivedi, N., Katiyar, D., & Goel, G. (2022). A Comparative Study of Various Software Development Life Cycle (SDLC) Models. *International Journal of Research in Engineering, Science and Management*, 5(3), 141-144.

- [5] F. Veriarinal, “Classification of Computer Machine Malfunction Detection Systems Using the Naive Bayes Method,” *Angew. Chemie Int. Ed.* 6(11), 951-952., vol. 2, no. 10, pp. 5-24, 2024.
- [6] Ferdinand Louis, M. Ficky Duskarnaen, & Hamidillah Ajie. (2021). Speed Test of Raspberry Pi as Private Cloud Storage for Small Office Home Office: With a Case Study at Upt Tik. *PINTER: Journal of Informatics and Computer Engineering Education*, 5(2), 42-49. <https://doi.org/10.21009/pinter.5.2.7>
- [7] Mahesh, B. (2020). Machine Learning Algorithms - A Review. *International Journal of Science and Research*, 9(1), 381-386. <https://doi.org/10.21275/ART20203995>
- [8] Ridwan, R., Lubis, H., & Kustanto, P. (2020). Implementation of Neural Network Algorithm in Predicting Student Graduation Rate. *Journal of Budidarma Informatics Media*, 4(2), 286. <https://doi.org/10.30865/mib.v4i2.2035>
- [9] Satwika, I. K. S., Sudiarsa, I. W., & Swari, M. H. P. (2020). Intrusion Detection System (Ids) Using Raspberry Pi 3 Based on Snort Case Study: Stmik Stikom Indonesia. *SCAN - Journal of Information and Communication Technology*, 15(3), 2-7. <https://doi.org/10.33005/scan.v15i3.2279>
- [10] Suharyanto, C. E., & Maulana, A. (2020). Network Attached Storage (Nas) Design Using Raspberry Pi for Micro, Small and Medium Enterprises (Umkm). *JITK (Journal of Computer Science and Technology)*, 5(2),
- [11] S. Jaelani, “the Role of Information System Attack Classification in Strengthening National Security and Combating Cyberwarfare,” 2024.