KLIK: Kajian Ilmiah Informatika dan Komputer
ISSN 2723-3898 (Media Online)
Vol 4, No 1, Agustus 2023, Hal 610-617
DOI 10.30865/klik.v4i1.1134
https://djournals.com/klik

Penerapan Naïve Bayes Classifier, Support Vector Machine, dan Decision Tree untuk Meningkatkan Deteksi Ancaman Keamanan Jaringan

Ahmad Turmudi Zy, Ananto Tri Sasongko, Antika Zahrotul Kamalia[3]

Fakultas Teknik, Teknik Informatika, Universitas Pelita Bangsa, Bekasi, Indonesia
Email: [1,*]turmudi@pelitabangsa.ac.id, [2]ananto@pelitabangsa.ac.id, [3]antika@pelitabangsa.ac.id
Email Penulis Korespondensi: turmudi@pelitabangsa.ac.id,

Abstrak−Penelitian ini bertujuan untuk menerapkan tiga algoritma machine learning, yaitu Naïve Bayes Classifier, Support Vector Machine (SVM), dan Decision Tree dalam meningkatkan deteksi ancaman keamanan jaringan. Penelitian ini menggunakan data dari beberapa sumber untuk melatih model machine learning dan menguji performanya dalam mendeteksi ancaman keamanan jaringan seperti malware, ransomware dan spyware. Hasil penelitian menunjukkan bahwa ketiga algoritma machine learning dapat meningkatkan efektivitas deteksi ancaman keamanan jaringan dengan tingkat akurasi yang lebih tinggi dibandingkan dengan metode konvensional. Decision Tree memberikan hasil terbaik dengan presisi 0.99, diikuti oleh SVM dengan presisi 0.98, dan Naivebayes dengan Presisi 0.86.

Kata Kunci: Naïve Bayes Classifier; Support Vector Machine (SVM); Decision Tree; Machine Learning; Keamanan Jaringan

Abstract−This research aims to implement three machine learning algorithms, namely Naïve Bayes Classifier, Support Vector Machine (SVM), and Decision Tree, to enhance network security threat detection. The study utilizes data from multiple sources to train the machine learning models and evaluate their performance in detecting network security threats such as malware, ransomware, and spyware. The research results indicate that all three machine learning algorithms can improve the effectiveness of network security threat detection, surpassing conventional methods in terms of accuracy. Decision Tree yields the best results with a precision of 0.98, , followed by SVM with a precision of 90%, While Naïve Bayes Classifier a precision of 0.86.

Keywords: Naïve Bayes Classifier; Support Vector Machine (SVM); Decision Tree; Machine Learning; Network Security

1. PENDAHULUAN

Dalam era digital yang semakin maju ini, keamanan jaringan menjadi suatu aspek yang sangat penting bagi organisasi, perusahaan, dan pengguna internet pada umumnya. Ancaman terhadap keamanan jaringan dapat berasal dari berbagai sumber, seperti serangan malware, serangan DDoS, peretasan, dan pencurian data. Untuk melindungi jaringan mereka [1].

Dalam konteks ini, terdapat tiga algoritma machine learning yang secara luas digunakan untuk meningkatkan deteksi ancaman keamanan jaringan, yaitu Naïve Bayes Classifier, Support Vector Machine (SVM), dan Decision Tree. Ketiga algoritma ini memiliki pendekatan yang berbeda namun sama-sama efektif dalam mengidentifikasi pola dan mengklasifikasikan data untuk tujuan deteksi ancaman.

Pertama, Naïve Bayes Classifier adalah algoritma klasifikasi probabilistik yang didasarkan pada teorema Bayes [2]. Algoritma ini mengasumsikan bahwa setiap fitur yang ada dalam data independen satu sama lain. Naïve Bayes Classifier sangat populer dalam deteksi ancaman keamanan jaringan karena kinerja yang baik dan kecepatan komputasi yang tinggi. Dengan menggunakan metode ini, dapat dilakukan klasifikasi pada data yang masuk ke jaringan dan mengidentifikasi apakah data tersebut merupakan ancaman atau tidak.

Kedua, Support Vector Machine (SVM) adalah algoritma pembelajaran mesin yang digunakan untuk klasifikasi dan regresi. SVM bekerja dengan mencari hyperplane terbaik yang dapat memisahkan dua kelas data [3]. Dalam konteks deteksi ancaman keamanan jaringan, SVM digunakan untuk membangun model yang dapat memisahkan data normal dari data yang mencurigakan atau berpotensi berbahaya. Dengan demikian, SVM dapat membantu dalam mendeteksi serangan dan mengklasifikasikan data dengan akurasi tinggi [4].

Ketiga, Decision Tree adalah algoritma pembelajaran mesin yang menggunakan struktur pohon untuk mengambil keputusan berdasarkan serangkaian aturan. Algoritma ini membagi data menjadi kelompok-kelompok yang lebih kecil berdasarkan atribut yang relevan [5]. Dalam konteks deteksi ancaman keamanan jaringan, Decision Tree dapat digunakan untuk membangun model yang mempelajari pola dari data historis yang terkait dengan serangan keamanan [6]. Dengan memanfaatkan model ini, dapat dilakukan prediksi terhadap data baru dan mengidentifikasi apakah data tersebut mencurigakan atau tidak.

Melalui penerapan Naïve Bayes Classifier, Support Vector Machine, dan Decision Tree, deteksi ancaman keamanan jaringan dapat ditingkatkan secara signifikan. Ketiga algoritma ini memberikan pendekatan yang berbeda namun saling melengkapi dalam mengklasifikasikan dan mengidentifikasi ancaman keamanan. Dengan memanfaatkan kemampuan mesin untuk mempelajari pola dan menganalisis data secara cepat [7], organisasi dan pengguna internet dapat meningkatkan keamanan jaringan mereka dan mengambil langkah-langkah proaktif untuk melindungi data sensitif dan sistem mereka dari serangan yang berpotensi merusak.

Ada banyak sekali kejahatan di dunia maya dimana target mereka umumnya adalah mencuri informasi yang dapat dijual. Sehingga penting sekali disini unutk menciptakan system pendeteksi ganguan keamanan (IDS) Intrusion Detection System [8].

Beberapa sisitem pendeteksi gangguan keamanan (IDS) telah di coba dan sangat sedikit yang bisa dideteksi, namun dengan menggunakan Supervised Machine Learning IDS kami bisa mendeteksi hingga 99.97% gangguan kemanan [9].

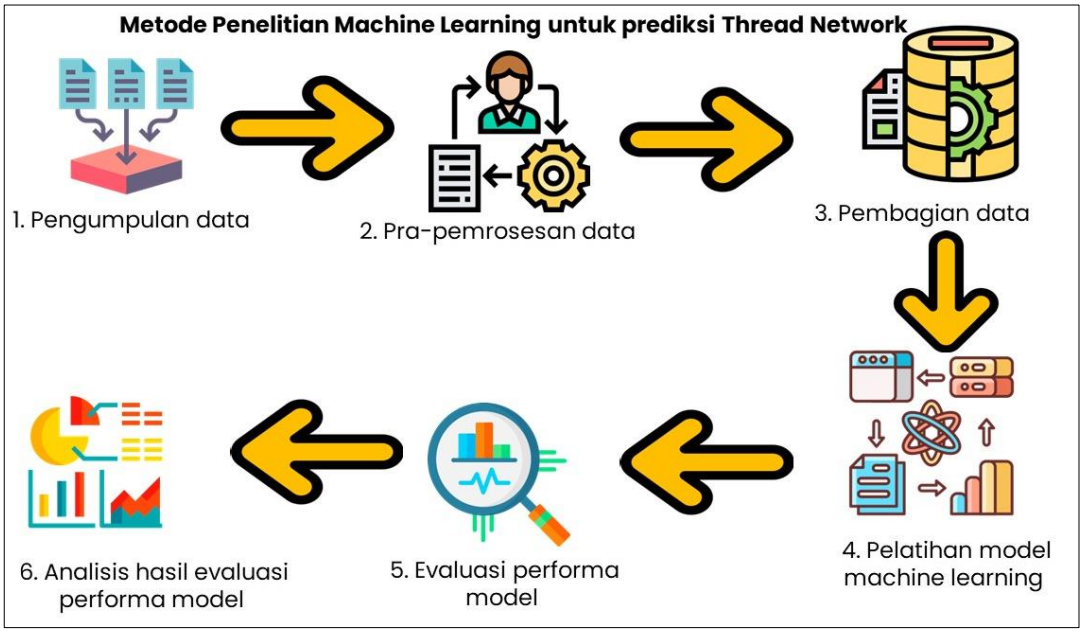
Di bidang keamanan informasi, deteksi serangan dan perlindungan informasi dari penyusup menjadi bidang penelitian massive saat ini [11]. Karena teknologi yang terus berubah dan metodologi modern, penyusup menggunakan berbagai mekanisme untuk menipu sasaran serangan., Untuk menjaga informasi tetap aman sangat penting berhati-hati dalam penyediaan keamanan jaringan dengan bantuan berbagai Teknik Intrusion Detection System (IDS) [12].

Karena banyaknya kerentanan yang ada dalam system jaringan sehingga banyak serangan cyber yang mengarah pada berbagai sistem jaringan [13]. Arah baru pada system pendeteksi ganguan keamanan (IDS). Adalah memanfaatkan model pembalajaran mesin untuk merancang sistem yang lebih kuat dengan tingkat deteksi yang lebih tinggi dan tingkat alarm palsu yang lebih rendah [14].

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Pada penelitian ini akan mengukur tingkat akurasi tiga model mesin *learning supervise learning*. Dengan model sebagia berikut:



Gambar 1. Tahapan pelaksanaan penlitian

a. Pengumpulan data dari beberapa sumber tentang ancaman keamanan jaringan seperti malware, ransomware, dan spyware.
b. Prapemrosesan data dengan membersihkan data yang tidak relevan dan melakukan normalisasi data. Pembagian data menjadi dua bagian yaitu data latih dan data uji.
c. Pelatihan model machine learning menggunakan tiga algoritma yaitu Naïve Bayes Classifier, Support Vector Machine (SVM), dan Decision Tree dengan menggunakan data latih.
d. Evaluasi performa model machine learning dengan menggunakan data uji untuk mengukur akurasi deteksi ancaman keamanan jaringan.
e. Analisis hasil evaluasi performa model machine learning untuk menentukan algoritma machine learning terbaik dalam meningkatkan deteksi ancaman keamanan jaringan.

2.2 Data Penelitian

Pada penelitian in kami menggunakan Dataset dari Kaggle [15], dengan kriteria data sebagai Berikut:

Dataset NSL-KDD (KDD Cup 1999 Data) adalah sebuah dataset yang digunakan untuk penelitian dalam bidang deteksi intrusi pada jaringan komputer. Dataset ini merupakan modifikasi dari dataset KDD Cup 1999, yang awalnya digunakan dalam kompetisi Knowledge Discovery and Data Mining (KDD) tahun 1999.

Berikut adalah beberapa informasi tentang kriteria data pada dataset NSL-KDD,

Jenis Data: Dataset NSL-KDD terdiri dari data yang dihasilkan dari simulasi serangan pada jaringan komputer. Data ini mencakup sejumlah atribut yang merepresentasikan koneksi jaringan.

Jumlah Atribut: Dataset NSL-KDD terdiri dari sejumlah atribut (features) yang merepresentasikan karakteristik dari setiap koneksi jaringan. Atribut ini meliputi informasi seperti protokol, jenis layanan, jumlah byte yang dikirim, dan lain-lain.

Klasifikasi Koneksi: Setiap koneksi dalam dataset NSL-KDD diklasifikasikan menjadi salah satu dari beberapa kategori, yaitu normal (normal connection) atau mencurigakan (suspicious connection). Kategori mencurigakan ini mencakup berbagai jenis serangan seperti Denial of Service (DoS), Probe, R2L (Remote-to-Local), dan U2R (User-to-Root).

Data Preprocessing: Dataset NSL-KDD telah melalui tahap preprocessing untuk mengatasi beberapa masalah pada dataset KDD Cup 1999. Preprocessing ini melibatkan eliminasi beberapa duplikasi, penanganan kelas minoritas, dan penggantian nilai-nilai yang hilang atau tidak valid.

Kriteria Inklusi: Dataset NSL-KDD mencakup berbagai jenis serangan yang mencoba untuk mensimulasikan skenario serangan pada jaringan komputer. Data yang direkam mencakup variasi serangan dan juga koneksi jaringan normal yang dapat digunakan sebagai acuan untuk deteksi intrusi.

Tujuan Penggunaan: Dataset NSL-KDD digunakan untuk penelitian dan pengembangan model deteksi intrusi berbasis data. Penelitian menggunakan dataset ini bertujuan untuk mengidentifikasi dan mengklasifikasikan serangan pada jaringan komputer serta meningkatkan keandalan sistem deteksi intrusi.

Penting untuk dicatat bahwa dataset NSL-KDD merupakan dataset yang umum digunakan dalam literatur penelitian terkait deteksi intrusi. Namun, karena kemajuan dalam teknologi dan keamanan jaringan, dataset ini mungkin sudah tidak lagi mencakup beberapa jenis serangan terbaru. Oleh karena itu, dalam penelitian deteksi intrusi saat ini, peneliti sering menggunakan dataset yang lebih mutakhir dan representatif dari serangan jaringan yang terbaru dan lebih kompleks.

Namun pada penelitian ini kami mencukupkan dengan penggunaan dataset ini, karena kami masih menganggap dataset ini relevan dengan dengan penelitian ini.

```
RangeIndex: 22543 entries, 0 to 22542
Data columns (total 43 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   duration                     22543 non-null  int64
 1   protocol_type                22543 non-null  object
 2   service                      22543 non-null  object
 3   flag                         22543 non-null  object
 4   src_bytes                    22543 non-null  int64
 5   dst_bytes                    22543 non-null  int64
 6   land                         22543 non-null  int64
 7   wrong_fragment               22543 non-null  int64
 8   urgent                       22543 non-null  int64
 9   hot                          22543 non-null  int64
 10  num_failed_logins            22543 non-null  int64
 11  logged_in                    22543 non-null  int64
 12  num_compromised              22543 non-null  int64
 13  root_shell                   22543 non-null  int64
 14  su_attempted                 22543 non-null  int64
 15  num_root                     22543 non-null  int64
 16  num_file_creations           22543 non-null  int64
 17  num_shells                   22543 non-null  int64
 18  num_access_files             22543 non-null  int64
 19  num_outbound_cmds            22543 non-null  int64
 20  is_host_login                22543 non-null  int64
 21  is_guest_login               22543 non-null  int64
 22  count                        22543 non-null  int64
 23  srv_count                    22543 non-null  int64
 24  serror_rate                  22543 non-null  float64
 25  srv_serror_rate              22543 non-null  float64
 26  rerror_rate                  22543 non-null  float64
 27  srv_rerror_rate              22543 non-null  float64
 28  same_srv_rate                22543 non-null  float64
 29  diff_srv_rate                22543 non-null  float64
 30  srv_diff_host_rate           22543 non-null  float64
 31  dst_host_count               22543 non-null  int64
 32  dst_host_srv_count           22543 non-null  int64
 33  dst_host_same_srv_rate       22543 non-null  float64
 34  dst_host_diff_srv_rate       22543 non-null  float64
 35  dst_host_same_src_port_rate  22543 non-null  float64
 36  dst_host_srv_diff_host_rate  22543 non-null  float64
 37  dst_host_serror_rate         22543 non-null  float64
 38  dst_host_srv_serror_rate     22543 non-null  float64
 39  dst_host_rerror_rate         22543 non-null  float64
 40  dst_host_srv_rerror_rate     22543 non-null  float64
 41  class                        22543 non-null  object
 42  level                        22543 non-null  int64
dtypes: float64(15), int64(24), object(4)
```

Gambar 2. Data Info model data

2.3 Google Colab

Penelitian ini menggunakan tools dari goole, yakni google Colab. Google Colab (Colaboratory) adalah sebuah layanan gratis dari Google yang memungkinkan pengguna untuk menulis dan mengeksekusi kode Python melalui browser web. Layanan ini berjalan di atas platform cloud milik Google, yang artinya tidak perlu menginstal atau mengkonfigurasi lingkungan Python di komputer lokal Anda. Google Colab sangat populer di kalangan peneliti, pengembang, dan pembelajar karena menyediakan banyak fitur berguna, Google Colab adalah alat yang sangat berguna untuk pemula maupun profesional dalam bidang data science, machine learning, dan pengolahan data secara umum. Dengan akses ke sumber daya komputasi tinggi, kemudahan berbagi, dan lingkungan yang interaktif, Google Colab telah menjadi pilihan populer untuk banyak proyek dan tugas komputasi berbasis Python.

3. HASIL DAN PEMBAHASAN

3.1 Pemrosesan Data

a. Memanggil data set dari Drive

```
+ Code  + Text

[1]  1 from google.colab import drive
     2 drive.mount('/content/drive')

     Mounted at /content/drive

     1 # library
     2 import io
     3 import pandas as pd
     4 import numpy as np
     5 import seaborn as sns                    #visualisation
     6 import matplotlib.pyplot as plt          #visualisation
     7 %matplotlib inline
     8 sns.set(color_codes=True)

[3]  1 # call data_train
     2 df_train = pd.read_csv('/content/drive/MyDrive/MyResearch/P
     3 df_train
```

Gambar 3. Proses pemanggilan data dari GDrive

Data set (data_train.txt) ini awalnya tidak ada header pada setiap kolomnya

	0	tcp	private	REJ	0.1	0.2	0.3	0.4	0.5	0.6	...	0.04.1	0.06.1	0.00.3	0.00.4	0.00.5	0.00.6	1.00.2	1.00.3	neptune	21
0	0	tcp	private	REJ	0	0	0	0	0	0	...	0.00	0.06	0.00	0.00	0.00	0.0	1.00	1.00	neptune	21
1	2	tcp	ftp_data	SF	12983	0	0	0	0	0	...	0.61	0.04	0.61	0.02	0.00	0.0	0.00	0.00	normal	21
2	0	icmp	eco_i	SF	20	0	0	0	0	0	...	1.00	0.00	1.00	0.28	0.00	0.0	0.00	0.00	saint	15
3	1	tcp	telnet	RSTO	0	15	0	0	0	0	...	0.31	0.17	0.03	0.02	0.00	0.0	0.83	0.71	mscan	11
4	0	tcp	http	SF	267	14515	0	0	0	0	...	1.00	0.00	0.01	0.03	0.01	0.0	0.00	0.00	normal	21
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
22538	0	tcp	smtp	SF	794	333	0	0	0	0	...	0.72	0.06	0.01	0.01	0.01	0.0	0.00	0.00	normal	21
22539	0	tcp	http	SF	317	938	0	0	0	0	...	1.00	0.00	0.01	0.01	0.01	0.0	0.00	0.00	normal	21
22540	0	tcp	http	SF	54540	8314	0	0	0	2	...	1.00	0.00	0.00	0.00	0.00	0.0	0.07	0.07	back	15
22541	0	udp	domain_u	SF	42	42	0	0	0	0	...	0.99	0.01	0.00	0.00	0.00	0.0	0.00	0.00	normal	21
22542	0	tcp	sunrpc	REJ	0	0	0	0	0	0	...	0.08	0.03	0.00	0.00	0.00	0.0	0.44	1.00	mscan	14

22543 rows × 43 columns

Gambar 4. Data awal dari data_train.txt tidak ada header pada setiap kolomnya

b. Pembuatan *header columns*

```
[5]  1 # make attribut for header
     2 columns = (['duration','protocol_type','service','flag','src_bytes','dst_bytes','land','wrong_fragment','urgent','hot'
     3 ,'num_failed_logins','logged_in','num_compromised','root_shell','su_attempted','num_root','num_file_creations'
     4 ,'num_shells','num_access_files','num_outbound_cmds','is_host_login','is_guest_login','count','srv_count','serror_rate'
     5 ,'srv_serror_rate','rerror_rate','srv_rerror_rate','same_srv_rate','diff_srv_rate','srv_diff_host_rate','dst_host_count','dst_host_srv_count'
     6 ,'dst_host_same_srv_rate','dst_host_diff_srv_rate','dst_host_same_src_port_rate','dst_host_srv_diff_host_rate','dst_host_serror_rate'
     7 ,'dst_host_srv_serror_rate','dst_host_rerror_rate','dst_host_srv_rerror_rate','class','level'])

[6]  1 # add header
     2 df_train.columns = columns
     3 df_test.columns = columns

     1 df_train
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_same_srv_rate	dst_host_diff_
0	0	tcp	private	REJ	0	0	0	0	0	0	...	0.00	
1	2	tcp	ftp_data	SF	12983	0	0	0	0	0	...	0.61	
2	0	icmp	eco_i	SF	20	0	0	0	0	0	...	1.00	
3	1	tcp	telnet	RSTO	0	15	0	0	0	0	...	0.31	
4	0	tcp	http	SF	267	14515	0	0	0	0	...	1.00	

Gambar 5. Pembuatan Data header untuk data_train.txt

c. *Dropping the duplicate rows*

```
[11]  1 df_train.shape

      (22543, 40)

[14]  1 duplicate_rows_df_train = df_train[df_train.duplicated()]
      2 print("number of duplicate rows: ", duplicate_rows_df_train.shape)

      number of duplicate rows:  (108, 40)

      1 df_train.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 22435 entries, 0 to 22542
Data columns (total 40 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   duration                     22435 non-null  int64
```

Gambar 6. Membuang Data *duplicate*

Membuang (*Drop*) Kolom, *protocol_type, service, flag*.

```
1 df_train.drop(columns=['protocol_type', 'service','flag'],inplace=True)
2 df_train. info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22543 entries, 0 to 22542
Data columns (total 40 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   duration                     22543 non-null  int64
 1   src_bytes                    22543 non-null  int64
 2   dst_bytes                    22543 non-null  int64
 3   land                         22543 non-null  int64
 4   wrong_fragment               22543 non-null  int64
 5   urgent                       22543 non-null  int64
 6   hot                          22543 non-null  int64
 7   num_failed_logins            22543 non-null  int64
 8   logged_in                    22543 non-null  int64
 9   num_compromised              22543 non-null  int64
 10  root_shell                   22543 non-null  int64
 11  su_attempted                 22543 non-null  int64
 12  num_root                     22543 non-null  int64
 13  num_file_creations           22543 non-null  int64
 14  num_shells                   22543 non-null  int64
 15  num_access_files             22543 non-null  int64
 16  num_outbound_cmds            22543 non-null  int64
 17  is_host_login                22543 non-null  int64
 18  is_guest_login               22543 non-null  int64
 19  count                        22543 non-null  int64
 20  srv_count                    22543 non-null  int64
 21  serror_rate                  22543 non-null  float64
 22  srv_serror_rate              22543 non-null  float64
 23  rerror_rate                  22543 non-null  float64
 24  srv_rerror_rate              22543 non-null  float64
 25  same_srv_rate                22543 non-null  float64
 26  diff_srv_rate                22543 non-null  float64
 27  srv_diff_host_rate           22543 non-null  float64
 28  dst_host_count               22543 non-null  int64
 29  dst_host_srv_count           22543 non-null  int64
 30  dst_host_same_srv_rate       22543 non-null  float64
 31  dst_host_diff_srv_rate       22543 non-null  float64
 32  dst_host_same_src_port_rate  22543 non-null  float64
 33  dst_host_srv_diff_host_rate  22543 non-null  float64
 34  dst_host_serror_rate         22543 non-null  float64
 35  dst_host_srv_serror_rate     22543 non-null  float64
 36  dst_host_rerror_rate         22543 non-null  float64
 37  dst_host_srv_rerror_rate     22543 non-null  float64
 38  class                        22543 non-null  object
 39  level                        22543 non-null  int64
dtypes: float64(15), int64(24), object(1)
memory usage: 6.9+ MB
```

Gambar 7. Menghapus kolom yang tidak memberi pengaruh pada hasil

d. Merubah Nilai *class*

```
[ ]  1 Results = set(df_train['class'].values)
     2 print(Results,end=" ")
```

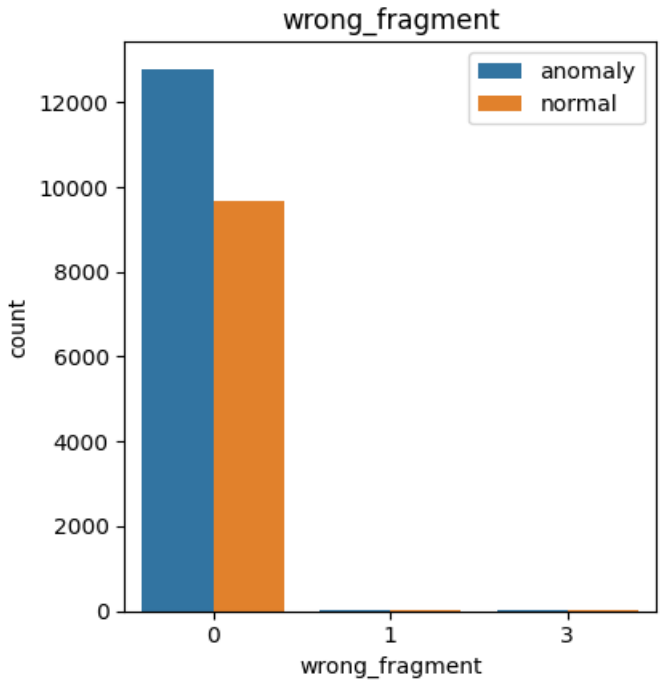
```
{'phf', 'named', 'normal', 'mscan', 'xterm', 'rootkit', 'pod', 'back', 'sendmail', 'teardrop', 'neptune', 'snmpgetattack', 'portsweep',
```

```
1 # replace target label
2 df_train.loc[df_train['class'] == "normal", "class"] = 'normal'
3 df_train.loc[df_train['class'] != 'normal', "class"] = 'anomaly'
```

```
[ ]  1 df_train.head(10)
```

Gambar 8. Merubah class menjadi *normal* dan *anomally*

Pada dataset NSL-KDD terdapat beberap imbalance data, dan berikut contoh data imbalance



Gambar 9. Grafik imbalance Data

3.2 Pengujian Data

Pengujian data menggunakan Google Colab, membandingkan 3 algorithma klasifikasi, Naïve Bayes, SVM, dan Decision Tree, dari data yang sudah di Kelola.

```
[ ]  1 from sklearn.metrics import classification_report
```

```
1 # Naïve Bayes
2 from sklearn.naive_bayes import GaussianNB
3 gaussian = GaussianNB()
4 gaussian.fit(X_train, Y_train)
5 Y_pred = gaussian.predict(X_test)
6 acc_gaussian = round(gaussian.score(X_train, Y_train) * 100, 2)
7 print('Gausian:'+str(acc_gaussian))
8 print(classification_report(Y_test, Y_pred))
9
10 # Support Vector Machine
11 from sklearn.svm import SVC
12 svc = SVC()
13 svc.fit(X_train, Y_train)
14 Y_pred = svc.predict(X_test)
15 acc_svc = round(svc.score(X_train, Y_train) * 100, 2)
16 print('SVC:'+str(acc_svc))
17 print(classification_report(Y_test, Y_pred))
18
19 # Decision Tree
20 from sklearn.tree import DecisionTreeClassifier
21 decision_tree = DecisionTreeClassifier()
22 decision_tree.fit(X_train, Y_train)
23 Y_pred = decision_tree.predict(X_test)
24 acc_decision_tree = round(decision_tree.score(X_train, Y_train) * 100, 2)
25 print('DT:'+str(acc_decision_tree))
26 print(classification_report(Y_test, Y_pred))
```

Gambar 10. Pengujian klasifikasi dengan 3 algoritma

3.3 Hasil Pengujian

Berikut adalah hasil dari pengujina yang dilakukan pada Goolge Colab.

Gausian:85.65	precision	recall	f1-score	support
anomaly	0.86	0.90	0.88	3868
normal	0.86	0.81	0.83	2895
accuracy			0.86	6763
macro avg	0.86	0.85	0.86	6763
weighted avg	0.86	0.86	0.86	6763
SVC:64.15	precision	recall	f1-score	support
anomaly	0.61	1.00	0.76	3868
normal	0.98	0.16	0.28	2895
accuracy			0.64	6763
macro avg	0.80	0.58	0.52	6763
weighted avg	0.77	0.64	0.56	6763
DT:100.0	precision	recall	f1-score	support
anomaly	1.00	1.00	1.00	3868
normal	0.99	0.99	0.99	2895
accuracy			0.99	6763
macro avg	0.99	0.99	0.99	6763
weighted avg	0.99	0.99	0.99	6763

Gambar 11. Hasil pengujian dengan 3 algoritma

4. KESIMPULAN

Hasil yang didapatkan pada penlitian kami adalah Decision Tree mendapatkan nilai terbaik dengan dengan hasil precision 0.99 pada posisi normal dan 1.00 pada anomaly, disusul oleh SVM dengan hasil precision 0.98 pada posisi normal dan 0.61 pada anomaly. Dan terahir adalah NaiveBayes dengan hasil precision 0.86 pada posisi normal dan 0.86 pada anomaly. Sehingga bis akita simpulkan bahwa untuk klasifikasi kami merekomendasikan unutk menggunakan Algorithma Decision tree sebagai pembuat keputusan untuk klasifikasi pada dataset NSLKDD.

REFERENCES

[1] P. Sharma, B. Dash, and M. F. Ansari, "Anti-Phishing Techniques – A Review of Cyber Defense Mechanisms," IJARCCE, vol. 11, no. 7, Jul. 2022, doi: 10.17148/ijarcce.2022.11728.
[2] A. Damuri, U. Riyanto, H. Rusdianto, and M. Aminudin, "Implementasi Data Mining dengan Algoritma Naïve Bayes Untuk Klasifikasi Kelayakan Penerima Bantuan Sembako," JURIKOM (Jurnal Riset Komputer), vol. 8, no. 6, p. 219, Dec. 2021, doi: 10.30865/jurikom.v8i6.3655.
[3] R. Sistem, P. N. Andono, and E. H. Rachmawanto, "JURNAL RESTI Evaluasi Ekstraksi Fitur GLCM dan LBP Menggunakan Multikernel SVM," vol. 1, no. 10, pp. 1–9, 2021.
[4] X. Xiong, S. Hu, D. Sun, S. Hao, H. Li, and G. Lin, "Detection of false data injection attack in power information physical system based on SVM–GAB algorithm," Energy Reports, vol. 8, pp. 1156–1164, Aug. 2022, doi: 10.1016/j.egyr.2022.02.290.
[5] Y. A. Wijaya, "Analisa Klasifikasi menggunakan Algoritma Decision Tree pada Data Log Firewall," JURSIMA (Jurnal Sistem Informasi dan …, 2021, [Online]. Available: https://ejournal.stmikgici.ac.id/index.php/jursima/article/view/303
[6] A. Yazdinejad, M. Kazemi, R. M. Parizi, A. Dehghantanha, and H. Karimipour, "An ensemble deep learning model for cyber threat hunting in industrial internet of things," Digital Communications and Networks, vol. 9, no. 1, pp. 101–110, Feb. 2023, doi: 10.1016/j.dcan.2022.09.008.
[7] B. Rathore, "Integration of Artificial Intelligence& It's Practices in Apparel Industry."
[8] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity, vol. 2, no. 1, Dec. 2019, doi: 10.1186/s42400-019-0038-7.
[9] M. Sharma, H. Elmiligi, and F. Gebali, "A Novel Intrusion Detection System for RPL-Based Cyber–Physical Systems," IEEE Canadian Journal of …, 2021, [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9429862/
[10] A. K. Dutta, R. Negi, and S. K. Shukla, "Robust multivariate anomaly-based intrusion detection system for cyber-physical systems," International Symposium on Cyber Security …, 2021, doi: 10.1007/978-3-030-78086-9_6.
[11] T. Lin, P. Wu, and F. Gao, "Information security of flowmeter communication network based on multi-sensor data fusion," Energy Reports, vol. 8, pp. 12643–12652, Nov. 2022, doi: 10.1016/j.egyr.2022.09.072.
[12] S. R. Khonde and U. Venugopal, "Hybrid architecture for distributed intrusion detection system," Ingenierie des Systemes d'Information, vol. 24, no. 1, pp. 19–28, 2019, doi: 10.18280/isi.240102.
[13] J. Ding, A. Qammar, Z. Zhang, A. Karim, and H. Ning, "Cyber Threats to Smart Grids: Review, Taxonomy, Potential Solutions, and Future Directions," Energies (Basel), vol. 15, no. 18, Sep. 2022, doi: 10.3390/en15186799.
[14] S. Moualla, K. Khorzom, and A. Jafar, "Improving the Performance of Machine Learning-Based Network Intrusion Detection Systems on the UNSW-NB15 Dataset," Comput Intell Neurosci, vol. 2021, pp. 1–13, Jun. 2021, doi: 10.1155/2021/5557577.
[15] "NSL-KDD | Kaggle." https://www.kaggle.com/datasets/hassan06/nslkdd (accessed Jul. 24, 2023).