Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Аналіз мережевого трафіку в режимі реального часу на основі ансамблевих методів машинного навчання

Виконав: студент VI курсу, групи СБм-62
спеціальності 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

(підпис) Юречко О.
(прізвище та ініціали)

Керівник (підпис) Баран І.О.
(прізвище та ініціали)

Нормоконтроль (підпис) Тимощук Д.І.
(прізвище та ініціали)

Завідувач кафедри (підпис) Загородна Н.В.
(прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль - 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

Студенту Юречку Олександру
(прізвище, ім'я, по батькові)

1. Тема роботи Аналіз мережевого трафіку в режимі реального часу на основі ансамблевих методів машинного навчання

Керівник роботи Баран Ігор Олегович, к.т.н., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «20» 11 2024 року № 4/7-1112

2. Термін подання студентом завершеної роботи 22.12.2024 р.

3. Вихідні дані до роботи наукові літературні джерела

4. Зміст роботи (перелік питань, які потрібно розробити)
1. Аналіз предметної області.
2. Теоретична основа та проектування застосунку.
3. Реалізація та тестування розробки.
4. Охорона праці та безпека в надзвичайних ситуаціях

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема роботи. 2. Актуальність. 3. Мета, задачі, об'єкт, предмет дослідження, наукова новизна. 4. Категорії ансамблевих методів машинного навчання. Набір даних. 5. Функціональні вимоги до проектованого застосунку. 6. Середовище виконання та програмні засоби розробки. 7. Діаграма варіантів використання. 8. Діаграма компонентів застосунку. 9. Передобробка набору даних. 10. Результати реалізації ансамблевих алгоритмів. 11. Результати A/B тестування. 12. Скріншоти роботи створеного застосунку. 14. Висновки. Основні результати дослідження.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., зав. каф. КС		
Безпека в надзвичайних ситуаціях	Клепчик В.М., проректор з АГРБ		

7. Дата видачі завдання 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	20.11 – 22.11	*Виконано*
2.	Підбір джерел аналіз мережевого трафіку в реальному		
	часі та ансамблеві методи машинного навчання	23.11 – 26.11	*Виконано*
3.	Опрацювання джерел про застосування ансамблеві		
	методи машинного навчання для аналізу трафіку в		
	реальному часі	27.11 – 30.11	*Виконано*
4.	Виконання дослідження щодо аналізу мережевого		
	трафіку засобами ансамблевих методів	01.12 – 06.12	
5	Розробка архітектури застосунку та написання коду	07.12 – 10.12	
6.	Оформлення розділу «Аналіз предметної області»	11.12 – 13.12	*Виконано*
7.	Оформлення розділу «Теоретична основа та проектування застосунку»	14.12 – 15.12	*Виконано*
8.	Оформлення розділу «Реалізація та тестування розробки»	16.12 – 18.12	*Виконано*
9.	Виконання завдання до підрозділу «Охорона праці та		
	безпека в надзвичайних ситуаціях»	06.12 – 16.12	*Виконано*
10.	Оформлення кваліфікаційної роботи	14.12 – 19.12	*Виконано*
11.	Нормоконтроль	18.12 – 20.12	*Виконано*
12.	Перевірка на плагіат	16.12 – 19.12	*Виконано*
13.	Попередній захист кваліфікаційної роботи	17.12 – 20.12	*Виконано*
14.	Захист кваліфікаційної роботи	23.12	

Студент _____ Юречко О.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Баран І.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Аналіз мережевого трафіку в режимі реального часу на основі ансамблевих методів машинного навчання // Юречко Олександр // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра кібербезпеки, група СБм-62 // Тернопіль, 2024 // с. – 70, рис. – 42, табл. – 5, слайд. – 13, бібліогр. –47.

Ключові слова: аналізатор пакетів, машинне навчання, мережевий трафік, метрики якості, AdaBoost, Random Forest

У першому розділі описується предметна область та проводиться аналіз наукової літератури.

Другий розділ присвячений розгляду алгоритмів Random Forest та AdaBoost, проведенню аналізу набору даних. Було розглянуто завдання класифікації трафіку, архітектуру мережі TCP/IP та аналізатори пакетів.

Також було визначено основні вимоги до застосунку та розроблено діаграми варіантів використання та компонентів.

Третій розділ присвячено реалізації ансамблевих моделей машинного навчання та застосунку. Також описується процес тестування та оцінки якості розробленого застосунку та навчених моделей.

Розроблений застосунок було протестовано для роботи в режимі реального часу (класифіковано звичайний трафік) і для роботи із локальними файлами (класифіковано шкідливий трафік).

ANNOTATION

Network traffic analysis in real time based on ensemble machine learning methods // Yurechko Oleksandr // Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cyber Security // Ternopil, 2024 // p. - 70, Fig. – 42, Table - 5, Slides - 13, References - 47.

Keywords: sniffer, machine learning, network traffic, quality metrics, AdaBoost, Random Forest

The first chapter describes the subject area and analyzes the scientific literature.

The second chapter is devoted to consideration of Random Forest and AdaBoost algorithms, analysis of the data set. The task of traffic classification, TCP/IP network architecture and packet analyzers were considered.

Basic application requirements were also identified and use case and component diagrams were developed.

The third section is devoted to the implementation of ensemble models of machine learning and application. The process of testing and evaluating the quality of the developed application and trained models is also described.

The developed application was tested for real-time operation (classified normal traffic) and for operation with local files (classified malicious traffic).

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ СКОРОЧЕНЬ І ТЕРМІНІВ

CICFlowMeter - інструмент, для аналізу та генерації потоку трафіку, що використовується для визначення параметрів трафіку для наборів даних канадського інституту кібербезпеки.

Chi Square - статистичний метод полягає у вирахуванні суми квадратів різниць отриманих значень та очікуваних значень, поділених на очікуване значення. Може використовуватися виділення атрибутів даних/.

DDoS атака (Distributed Denial of Service) - тип мережевої атаки, заснований на обмеженнях пропускної спроможності мережевих ресурсів з метою перевищити здатність сайту обробляти інформацію та викликати відмову в обслуговуванні.

Flood атаки - атаки, що полягають у навантаженні певних протоколів зв'язку (TCP, UDP, ICMP) з метою відключення сервера клієнта.

IDS (Intrusion Detection System) - система безпеки, що спостерігає за комп'ютерами та мережевим трафіком. При виявленні підозрілої активності система відправляє оповіщення про виявлення.

TCP – Transmission Control Protocol.

UDP – User Datagram Protocol.

A/B-тестування (A/B testing, Split testing) — метод маркетингового дослідження, суть якого полягає в тому, що контрольна група елементів порівнюється з набором тестових груп, в яких один або декілька показників були змінені, для того, щоб з'ясувати, які зі змін покращують цільовий показник.

Ансамблеві методи - це парадигма машинного навчання, де кілька моделей, часто званих «слабкими учнями», навчаються для вирішення однієї і тієї ж проблеми і об'єднуються для отримання кращих результатів.

НД – набір даних.

Машинне навчання – клас методів штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися.

ЗМІСТ

ВСТУП

Актуальність теми. В даний час продовжується зростання кількості пристроїв користувача, розумних пристроїв, що об'єднуються в системи розумних будинків, камер спостереження. Одночасно з цим збільшується обсяг трафіку, використовуваного людьми та системами. Будь-який з представлених вище пристроїв може бути схильний до зламів і використано при проведенні DDoS-атак зловмисниками [3].

Проведений аналіз джерел [8, 9] свідчить, що в Україні за даними Держспецзв’язку кількість кібератак у 2023 році порівняно з 2022 роком зросла, на 16 %., а всіх кіберінцидентів на 62,5 % Найбільше постраждали від зафіксованих атак сайти уряду та різних урядових організацій (майже 350 атак), місцеві органи влади (більше 270 атак), підприємства та організації у галузі кібербезпеки та оборони (175 атак).

У зв'язку з цим, розробка нових систем виявлення вторгнень останніми роками стає дедалі більш пріоритетним завданням багатьох компаній.

Мета дослідження: проведення ґрунтовного аналізу мережного трафіку в режимі реального часу на основі ансамблевих методів машинного навчання та створення програмного застосунку для цього.

В роботі поставлено та розв’язано наступні задачі:

– провести огляд наукової літератури;
– вибрати алгоритми класифікації;
– реалізувати вибрані алгоритми класифікації;
– здійснити оцінку якості використаних алгоритмів;
– реалізувати застосунок для аналізу трафіку в реальному часі;
– провести тестування розробленого застосунку.

Об’єкт дослідження: мережевий трафік.

Предмет дослідження: аналіз трафіку мережі в онлайні із використанням методів машинного навчання.

Методи дослідження: наукові публікації українських та зарубіжних учених за тематикою дослідження, фундаментальні положення кібербезпеки.

Наукова новизна отриманих результатів:

– отримав подальший розвиток онлайн механізм аналізу трафіку мережі із використанням ансамблевих методів машинного навчання.

Практичне значення одержаних результатів. Можуть бути застосовані для успішного класифікування як звичайного трафіку із сайтів у реальному часі, так і шкідливого трафіку під час зчитування інформації з файлу.

Апробація. Результати дослідження апробовано на XII науково-технічній конференції «Інформаційні моделі, системи та технології» у вигляді опублікованих тез.

Структура роботи. Робота складається з пояснювальної записки та графічної частини. Пояснювальна записка складається з вступу, 4 розділів, висновків, списку використаної літератури та застосунків. Обсяг роботи: пояснювальна записка – 70 арк. формату А4, графічна частина – 14 слайдів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

У цьому розділі будуть проаналізовані праці, в яких розглянуті питання дослідження виявлення вторгнень для мережевого трафіку із застосуванням методів машинного навчання [1], в т.ч і ансамблевих [2].

1.1 Публікація 1

У роботі [10] розглядається побудова системи виявлення DDoS атак [3] з урахуванням алгоритму Random Forest. НД було зібрано вручну, використовуючи такий інструмент створення DDoS -атак як TFN2K. У ході аналізу були виявлені відмінності та закономірності нормального трафіку та трафіку під час проведення деяких типів атак, таких як [5] TCP flood, UDP flood, ICMP flood, результати представлені на рис. 1.1 – 1.3.

Table 1. Normal TCP data is compared with TCP flood attack packets.

type of data	Normal TCP data	TCP flood attack data
Difference point 1	Packet sequence number is regular	Packet sequence number is random
Difference point 2	The source IP is regular, and the destination IP is more than one	Source IP is confusing, only one destination IP
Difference point 3	Packet identification bits are different	The packet identifier is the same as the packet sent by the TCP flood attack

Рисунок 1.1 – Порівняння звичайного TCP з TCP під час атаки

Table 2. Normal UDP data is compared with UDP flood attack packets.

type of data	Normal UDP data	UDP flood attack data
Difference point 1	Specific port number	Random port number
Difference point 2	Source IP is regular	Source IP confusion
Difference point 3	Irregular packet length	The packet length is the same

Рисунок 1.2 – Порівняння звичайного UDP з UDP під час атаки

Table 3. Normal ICMP data is compared with ICMP flood attack packets.

type of data	Normal ICMP data	ICMP flood attack data
Difference point 1	Packet sequence increment	Packet sequence confusion
Difference point 2	Invariant identifier	Random identifier
Difference point 3	One to one or one to many	IP addresses are rarely duplicated

Рисунок 1.3 – Порівняння звичайного ICMP та ICMP під час атаки

Для перевірки ефективності моделі частина НД, котра залишилася опісля навчання, була додана до звичайного трафіку. Результати навчання моделі представлені на рис. 1.4 – 1.6.

Table 4. TCP flood attack detection result.

Algorithm model	The sampling period (T)/s	2	4	6	8
Random forest	FR	0.14	0.15	0.15	0.16
	DR	99.15	98.69	98.50	98.10
	AR	99.93	99.67	99.57	99.49
SVM	FR	0.25	0.50	0.43	0.68
	DR	98.15	97.25	96.14	94.48
	AR	98.93	98.5	98.38	98.2

Рисунок 1.4 – Результати навчання моделі для TCP атак

Table 5. UDP flood attack detection result.

Algorithm model	The sampling period (T)/s	2	4	6	8
Random forest	FR	0.24	0.30	0.53	0.42
	DR	97.75	96.83	95.23	93.13
	AR	99.93	99.67	99.57	99.49
SVM	FR	0.25	0.48	0.49	0.51
	DR	99.16	98.95	98.43	98.05
	AR	98.93	98.5	98.38	98.2

Рисунок 1.5 – Результати навчання моделі для UDP атак

Table 6. ICMP flood attack detection result.

Algorithm model	The sampling period (T)/s	2	4	6	8
Random forest	FR	0.12	0.28	0.75	1.06
	DR	99.14	98.63	98.42	97.91
	AR	99.87	99.67	99.14	98.56
SVM	FR	0.91	1.12	2.75	3.33
	DR	98.22	97.22	96.34	94.41
	AR	98.87	97.79	96.90	95.49

Рисунок 1.6 – Результати навчання моделі для атак ICMP

Як метрики результатів використовувалися FR, DR, AR. FR (false positive rate) - ставлення FP (False Positive) до загальної кількості певних позитивних записів. DR (detection rate) – відношення TN (True Negative) записів, до загальної кількості негативно визначених записів. AR (total detection rate) – відношення правильно визначених записів до загальної кількості записів.

Як видно з наведених вище рисунків, модель, навчена за допомогою алгоритму Random Forest, має більш високі результати завдання виявлення вторгнень.

1.2 Публікація 2

У роботі [11] розглядаються способи поліпшення існуючих IDS [4] з допомогою CFS + Ensemble Classifiers. Як алгоритми навчання обрані AdaBoost, Random Forest, Reptree та J48. Моделі навчаються на НД KDD99 та NSL-KDD. НД були розділені на навчання 70% і тестів 30%, кожен відповідно.

У ході проведення дослідження було виділено 6 параметрів для бінарної класифікації на НД KDD99 та 11 параметрів для багатокласової класифікації з НД KDD99. Було виділено 13 параметрів для бінарної та багатокласової класифікації у НД NSL-KDD.

Як метрики обрані: FPR; Accuracy; TruePositive; Precision; Recall; F1.

Результати проведеного дослідження наведено на рис. 1.7. Вони свідчать, що моделі, навчені за допомогою алгоритму Random Forest, виявилися найефективнішими на НД KDD99 та NSL-KDD.

Method	Accuracy Detection Rate (%)	FR Rate (%)
DAR Ensemble	78.88	N/A
Naive Bayes-KNN-CF	82.00	05.43
Feature Selection + SVM	82.37	15.00
GAR Forest + Symmatrixal Uncertainity	85.00	12.20
Bagging j48	84.25	02.79
PCA+PSO	99.40	0.60
Propose Model Bagging Random Forest (KDD99 dataset)	**99.90**	**0.00**
Propose Model Bagging Random Forest (NSLKDD dataset)	**98.60**	**0.50**

Рисунок 1.7 – Порівняння оцінок якості класифікації навчених моделей

1.3 Публікація 3

У публікації [12] розглядаються та порівнюються алгоритми Real AdaBoost, Gentle AdaBoost та Modest AdaBoost. Вони були використані для навчання IDS, що виконує бінарну класифікацію. Gentle AdaBoost відрізняється більшою стабільністю в роботі, це дозволяє отримати точніші результати. Modest AdaBoost було створено для зменшення помилки узагальнення ціною більшої помилки навчання. Для створення слабких класифікаторів використовуються Classification and Regression Trees (CART) із різною глибиною.

Дослідження проводилося на кількох НД: KDD99; UNSW-NB15; NSL-KDD; TRAbID; CICIDS2017.

Для метрики якості навчання використовували ErrorRate. Щоб виявити залежність якості навчання від глибини дерев, були створені моделі з глибинами 2-5.

Алгоритми Real AdaBoost і Gentle AdaBoost набагато краще підходять для бінарної класифікації, так як вони мають у середньому на 70% краще частоту помилки в порівнянні з Modest AdaBoost, їх крива частоти помилки вирівнюється за менше ітерацій (рис. 1.8). Modest AdaBoost швидше навчається,

але для досягнення прийнятних результатів потрібна велика глибина дерев, що збільшує час навчання.

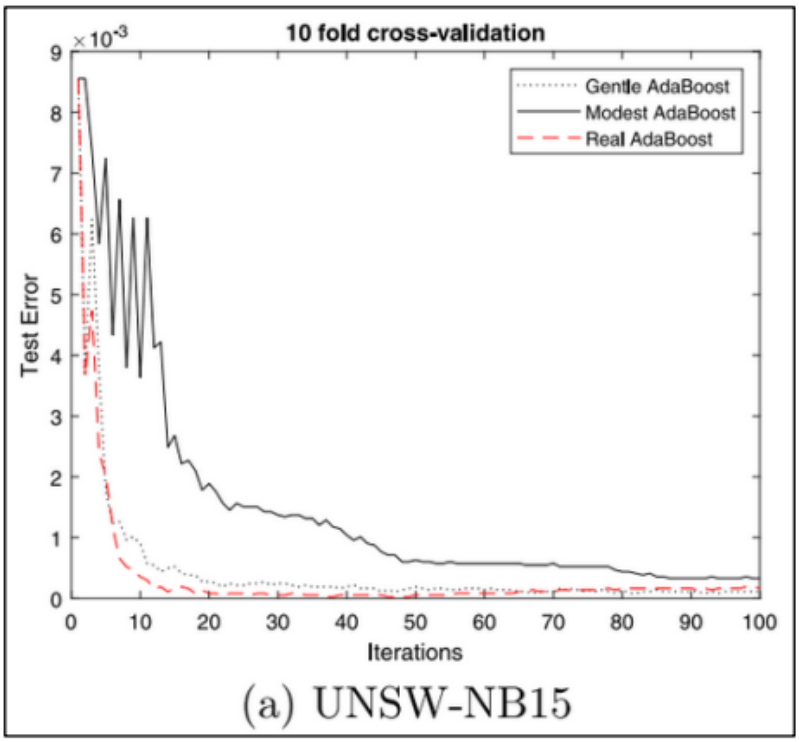


Рисунок 1.8 – Крива помилки алгоритмів

1.4 Публікація 4

У роботі досліджується ефективність tri-Adaboost алгоритму для побудови IDS. Сутність tri-Adaboost полягає у використанні слабких учнів, навчених трьома різними алгоритмами Adaboost – це discrete, real та gentle. Для виділення необхідних атрибутів використали метод chi-square [7]. Як НД для навчання моделі використовується KDD99.

Дані для навчання розбиваються на дві категорії: розмічені та нерозмічені. Перші використовують для навчання слабких учнів. Другі задіяні у процесі донавчання моделі з урахуванням результатів навчання слабких учнів.

Схема роботи алгоритму наведена на рис. 1.9.

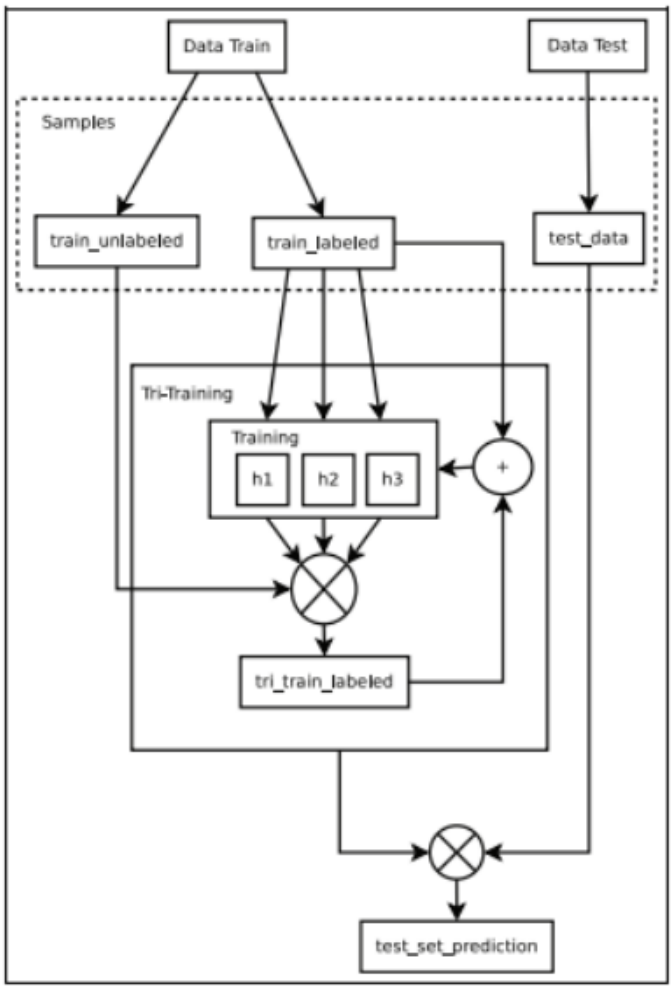


Рисунок 1.9 – Схема роботи алгоритму

Для отримання результатів алгоритму було проведено 30 симуляцій. Як метрики якості навчання були обрані Accuracy, Precision, Sensitivity, False Positive Rate та Detection rate. Результати навчання для моделі з усіма атрибутами та з половиною атрибутів показані на рис. 1.10.

Name	FFV	SFV
Testing size	12,000	12,000
Unlabeled size	32,000	32,000
Labeled size	6000	6000
Accuracy (%)	96.97	**96.99**
Precision (%)	98.57	**98.61**
Sensitivity (%)	97.81	97.80
Detection rate (%)	88.96	88.91
False positive rate (%)	7.42	**7.26**
Train-time (s)	85.13	**47.63**
Predict-time (s)	4.82	**2.66**

Рисунок 1.10 – Порівняння результатів моделей для різної кількості атрибутів

Також у ході дослідження було виявлено залежність якості навчання моделі від кількості нерозмічених даних у НД. Зі збільшенням нерозмічених даних збільшується ефективність. При цьому зберігається співвідношення нерозмічених до розмічених 10 до 1. Залежність представлена на рис. 1.11.

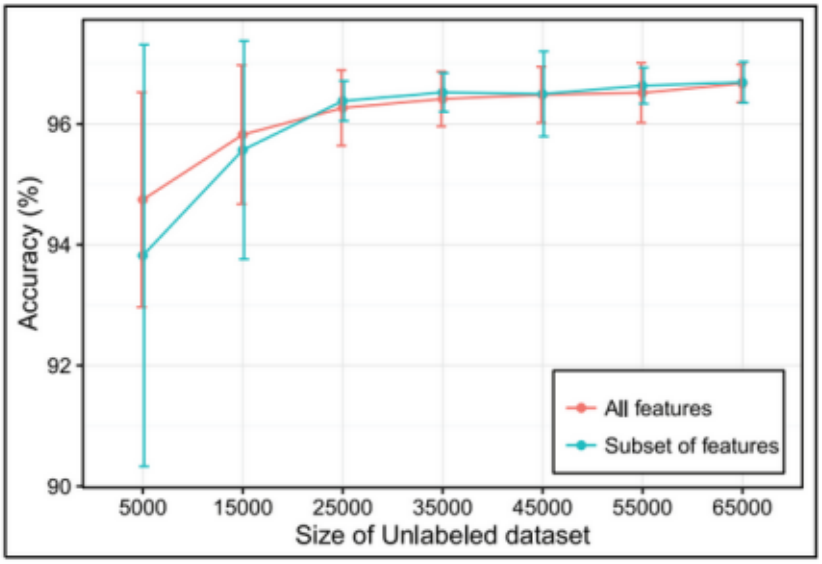


Рисунок 1.11 – Залежність точності від розміру нерозміченого НД

У ході роботи було виявлено підвищення ефективності та точності IDS, навченої за допомогою tri-Adaboost алгоритму, результати навчання наведено на рис. 1.12.

	STA	SSM
Detection rate (%)	89.05	97.34
False positive rate (%)	**7.54**	8.65
Precision rate (%)	**98.54**	92.35
Detection time (s)	**7.34**	98.14

Рисунок 1.12 – Результати навчання

Для тестового НД, що складається з 45 000 записів, точність алгоритму становила 98,08%. При цьому, для навчання моделі використовувалося 450 розмічених записів і 4500 нерозмічених.

1. Публікація 5

У статті [14] розглянуто метод побудови системи детектування мережевих атак із зстосуванням алгоритмів машинного навчання. Як алгоритми обрані K-Nearest Neighbours (KNN), Quadratic discriminant analysis (QDA), Iterative Dichotomiser 3 (ID3), Random Forest (RF), Adaptive Boosting (AdaBoost), Multilayer Perceptron (MLP) та Naive Bayes (NB).

Як НД для навчання та тестування було обрано Bot-IoT. Він був розділений на дві частини: 80% складає навчальна частина та 20% тестова. Для отримання атрибутів трафіку був використаний інструмент CICFlowMeter. За допомогою нього було виявлено 84 атрибути.

Для зменшення кількості атрибутів використали алгоритм Random Forest Regressor. Цей алгоритм зменшив кількість параметрів з 84 до 7.

Ваги атрибутів показані на рис. 1.13.

Для метрики ефективності моделі було обрано Precision, Recall, Accuracy та F-measure.

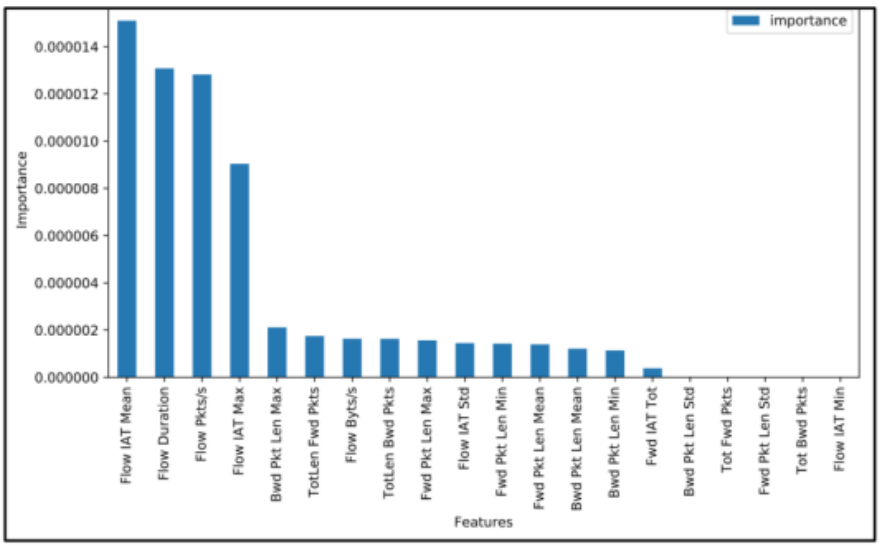


Рисунок 1.13 – Порівняльна діаграма важливості параметрів класифікації

Оцінку ефективності було проведено у кількох варіаціях: таблиця результатів з окремих типів атак, використовуючи F-Measures; метрики окремих алгоритмів для 13 атрибутів, для 7 атрибутів.

Результати представлені нижче на рис. 1.14. Як видно з отриманих даних, найкращим алгоритмом для вирішення задачі виявлення атак став KNN із результатом метрики F-Measures 0,99. Алгоритми AdaBoost та Random Forest теж показали високу ефективність, отримавши результати 0,97, проте виграли за часом роботи.

ML Algorithm	Accuracy	Precision	Recall	F-Measure	Time
NB	0.78	0.84	0.78	*0.75*	**5.056**
QDA	0.88	0.89	0.88	0.87	6.1964
RF	0.98	0.98	0.98	0.98	27.0328
ID3	0.99	0.99	0.99	0.99	19.3447
Adaboost	1.0	1.0	1.0	**1.0**	308.9403
MLP	0.84	0.88	0.84	0.83	1011.5001
KNN	0.99	0.99	0.99	0.99	*2052.1801*

Рисунок 1.14 – Порівняння метрик алгоритмів

1.6 Публікація 6

Робота [15] розглядає методи побудови NIDS (Network Intrusion Detection System) на основі методів машинного навчання.

Розглядаються алгоритми Decision Tree, Random Forest, Bagging, AdaBoost, K-nearest Neighbours, N-centroid, LinearSVC, RBFSVC та L-BFGS.

Дослідження проводили на НД CICIDS2017. Для отримання атрибутів трафіку використовувався CICFlowMeter, який отримав 84 атрибути.

Для метрики точності навченої системи було обрано Balanced Accuracy, Precision, Recall, F1-score та ROC-AUC.

У ході роботи було виявлено найкращі параметри алгоритмів для досягнення найкращих результатів навчання моделі, результати представлені на рис. 1.15.

Parameter tuning search results		
Algorithm	Parameters	Search results
dtree	max_features max_depth	no clear winners no clear winners
rforest	max_features max_depth	no clear winners no clear winners
bag	max_features max_samples	0.7 / 0.8 (18/21) 0.9 / 1.0 (19/21)
ada	n_estimators learning_rate	no clear winners 0.6 (7/21)
knn	n_neighbors distance_metric	1 (20/21) manhattan (17/21)
linsvc	max_iterations tolerance	1000 (16/21) $10e-5$ (21/21)
binlr	max_iterations tolerance	1000 (13/21) $10e-3$ (21/21)

Рисунок 1.15 – Найкращі параметри

Згідно з результатами дослідження, дерево-подібні класифікатори виявилися найточнішими на тестовому НД, із середнім результатом виявлення DoS/DDoS атак вище 99%. AdaBoost зміг отримати високий результат у виявленні Infiltration атак, показник Balanced Accuracy був у діапазоні 75-79,2%. Проте, спостерігався низький результат метрики Recall. Таблиця з результатами для Random Forrest представлена на рис. 1.16.

rforest	1: FTP / SSH bruteforce	MinMax	0.9919	0.9958	0.8855	0.7946	1.0000	0.9958
		No	1.0000	0.9997	0.9997	1.0000	0.9993	0.9997
		Z	0.9922	0.9960	0.8866	0.7963	1.0000	0.9960
	2: DoS / Heartbleed	MinMax	0.9999	0.9999	0.9999	0.9999	0.9998	0.9999
		No	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
		Z	0.9999	0.9999	0.9998	0.9999	0.9998	0.9999
	3: Web attacks	MinMax	0.9997	0.9898	0.9870	0.9945	0.9796	0.9898
		No	0.9997	0.9927	0.9901	0.9947	0.9855	0.9927
		Z	0.9998	0.9944	0.9923	0.9958	0.9889	0.9944
	4: Infiltration	MinMax	0.9999	0.8437	0.7586	0.8462	0.6875	0.8437
		No	0.9999	0.7000	0.5714	1.0000	0.4000	0.7000
		Z	0.9999	0.6875	0.4800	0.6667	0.3750	0.6875
	5: Botnet	MinMax	0.9961	0.8102	0.7657	1.0000	0.6204	0.8102
		No	1.0000	0.9992	0.9976	0.9968	0.9984	0.9992
		Z	0.9999	0.9976	0.9961	0.9969	0.9953	0.9976
	6: Portscan	MinMax	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
		No	0.9999	0.9999	0.9999	1.0000	0.9998	0.9999
		Z	0.9999	0.9999	0.9999	1.0000	0.9998	0.9999
	7: DDoS	MinMax	0.9999	0.9999	0.9999	1.0000	0.9998	0.9999
		No	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000
		Z	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
	8: Merged (all types)	MinMax	0.9999	0.9998	0.9998	0.9999	0.9997	0.9998
		No	0.9996	0.9992	0.9991	0.9997	0.9984	0.9992
		Z	0.9999	0.9998	0.9998	0.9999	0.9997	0.9998

Рисунок 1.16 – Результати для моделі Random Forest

1.7 Публікація 7

У статті [16] описано механізм додавання системи виявлення атак SDN. У таких мережах SDN рівень керування мережею є відокремленим від рівня передавання даних та втілюється програмно (рис. 1.17).

Для створення віртуального оточення, в рамках якого можна реалізувати SDN, використали Mininet. Як перемикач для системи був обраний OpenVSwitch. Для реалізації протоколу OpenFlow було використано sFlow-RT.

Модель виявлення атак навчена за допомогою алгоритму AdaBoost з однорівневими деревами вибору як слабкі учні. Модель була реалізована серед WEKA.

НД було зібрано за допомогою імітації атак, реалізованих за допомогою бібліотеки Scapy для Python. НД містить інформацію про DDoS -атаки для 6 протоколів: TCP; UDP; ICMP; ARP; IPv4; SSH.

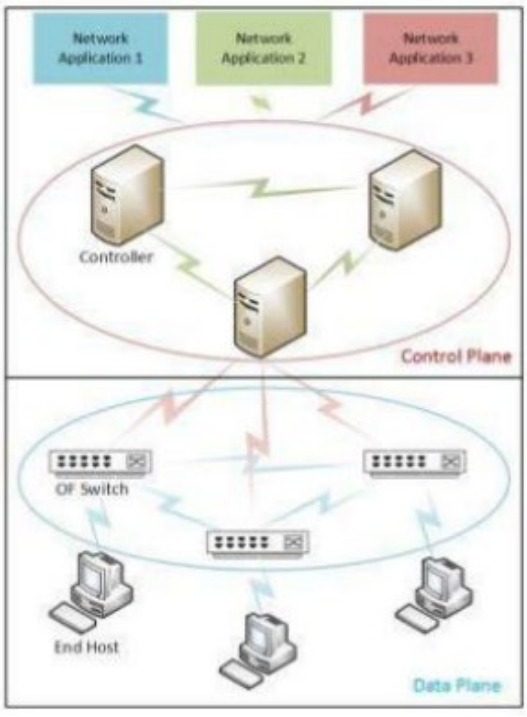


Рисунок 1.17 – Схема роботи SDN мереж

Як метрик якості навчання моделі були обрані: Accuracy, Precision, Recall, F-measures. Результати представлені на рис. 1.18. У роботі модель, навчена з

допомогою алгоритму AdaBoost, змогла отримати точність 93% в SDN системі.

No	Techniques	Precision	Recall	F-Measure	ROC Area
1	Bayes Net	0.889	0.885	0.885	0.863
2	Nave Bayes	0.731	0.705	0.693	0.707
3	Multilayer Perceptron	0.836	0.836	0.836	0.834
4	Support Vector Machine(Kernel=3)	0.853	0.852	0.852	0.853
5	AdaBoost(Decision Stump as weak classi er)	0.934	0.934	0.934	0.887
6	J48 decision tree	0.903	0.902	0.901	0.880
7	Random Forest	0.837	0.836	0.836	0.899

Рисунок 1.18 – Результати навчання моделей

1.8 Висновки до першого розділу

У ході аналізу предметної галузі було проведено огляд літератури.

Було виявлено, що дослідження у сфері застосування алгоритмів машинного навчання для побудови сучасних систем виявлення вторгнень дуже актуальні.

Також було виявлено високу ефективність обраних ансамблевих алгоритмів.

2 ТЕОРЕТИЧНА ОСНОВА ТА ПРОЕКТУВАННЯ ЗАСТОСУНКУ

2.1 Завдання класифікації трафіку

Завданням класифікації в машинному навчанні називається клас завдань, у яких наявна множина об'єктів з певними ознаками, котрі розділені на деяку множину класів, і потрібно на основі навчальної вибірки побудувати алгоритм, що здатний визначити клас для довільного об'єкта вихідного НД або нових даних [17].

Для мережного трафіку задача класифікації може бути сформульована так: визначення класу трафіку з урахуванням деяких вхідних характеристик мережного трафіку. Як вхідні характеристики можуть бути отримані дані пакетів, числові та частотні характеристики. Як вихідні характеристики може використовуватися ідентифікатор конкретного застосунку, або ідентифікатор класу трафіку [18] .

2.2 Ансамблеві методи машинного навчання

Ансамблеві методи є галуззю машинного навчання, котра на вирішення завдання навчає множину моделей про «слабких учнів» і поєднує їх задля отримання кращого результату [3].

Можна виділити три категорії ансамблевих алгоритмів [13].

1. Bagging або Bootstrap Aggregation. У цій категорії паралельно та незалежно навчаються кілька «слабких учнів», результат класифікації визначається голосуванням. Для організації майже повної незалежності дані для навчання слабких учнів вибираються з навчальної вибірки за допомогою статистичного методу bootstrap. Суть методу можна описати так, виробляється генерація вибірок розміру B з вихідного НД розміру N здійснюється шляхом випадкового вибору з повтореннями елементів у кожному з B разів (рис. 2.1) [2].

Рисунок 2.1 – Приклад роботи бутстреп

2. Boosting. У цій категорії алгоритмів група «слабких учнів» навчається послідовно, кожен новий «слабкий учень» навчається на даних, які були погано класифіковані попереднім учнем. Результат передбачення визначається кожним слабким учнем, ними накладається вага моделі, і, скомбінувавши результати передбачень, визначається результат. Даний вид навчання дозволяє отримувати точні передбачення на всіх типах даних [19] .

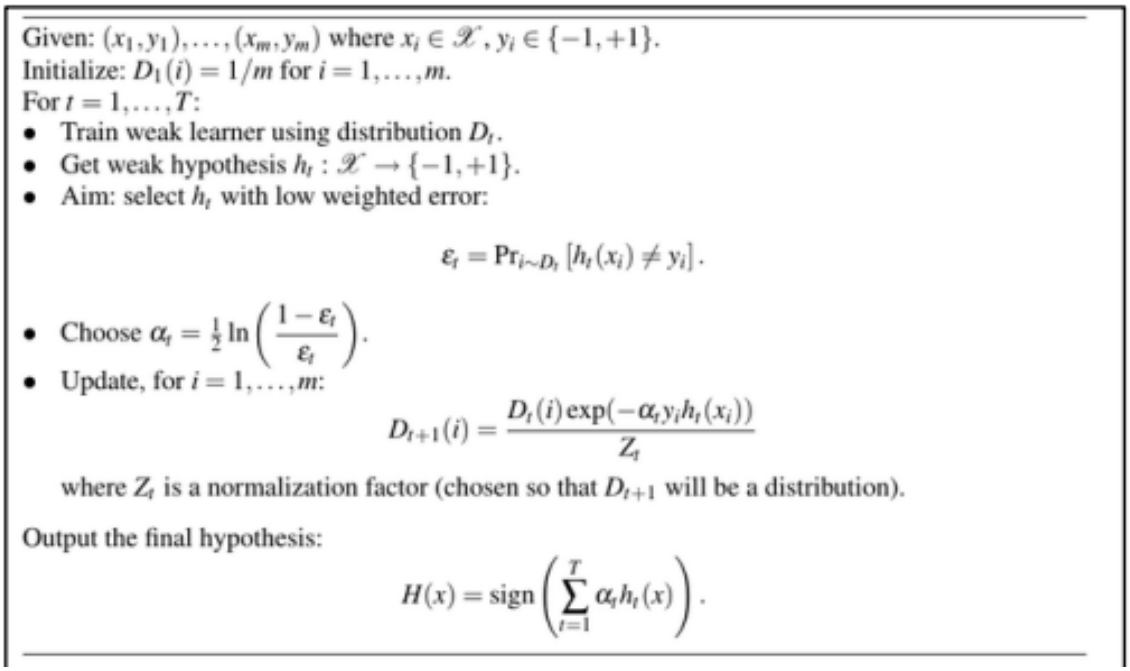
3. Stacking. У цій категорії алгоритмів «слабкі учні» навчені різними методами, а результати їх прогнозів навчають мета-модель, яка визначає кінцевий результат прогнозу. Властиво "слабким учнем" може бути і сама мета-модель [2].

З метою здобуття поставленої мети обрано два ансамблеві алгоритми машинного навчання: Random Forest; AdaBoost.

Random Forest – представник ансамблевих алгоритмів машинного навчання категорії bagging. Модель, навчена за допомогою алгоритму, є колекцією дерев вибору, кожне з яких навчається незалежно і паралельно, а результат виконання класифікації встановлюється спільним голосуванням. Серед переваг цього алгоритму варто відзначити його простоту, швидкість порівняно з іншими алгоритмами бегінга та бустингу та високу точність [20] .

AdaBoost – ансамблевий алгоритм машинного навчання категорії boosting. Схема його є ітераційний алгоритм, у якому після кожної епохи навчання ваги

«слабких учнів» індивідуально перераховуються. Ваги збільшуються для некоректно передбачених екземплярів та зменшуються для коректно передбачених. На першій ітерації слабкі учні навчаються на оригінальному НД, із кожною наступною ітерацією НД змінюється. Схема алгоритму [22] представлена на рис. 2.2. Результат класифікації визначається загальним голосуванням навчених «слабких учнів» [21] .


Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathcal{X}$, $y_i \in \{-1, +1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1, \ldots, m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : \mathcal{X} \to \{-1, +1\}$.
- Aim: select h_t with low weighted error:

$$\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right)$.
- Update, for $i = 1, \ldots, m$:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Рисунок 2.2 – Схема алгоритму AdaBoost

2.3 Будова мережі

В даний час найбільш поширена мережева модель TCP/IP що складається з 4 рівнів: прикладний; транспортний; міжмережевий; мережевий.

Мережевий рівень забезпечує адресацію обладнання та надає протоколи для фізичного транспортування даних.

Міжмережевий рівень визначає такі протоколи, котрі є відповідальними за передавання даних по всій мережі. Основними протоколами цього рівня є протоколи IP та ICMP. IP - протокол, котрий відповідає за доставленнф пакетів від джерела до одержувача, сканує IP- адреси в заголовках пакетів. ICMP – протокол, котрий повідомляє щодо мережевих помилок. ARP – відповідальний

за пошук адреси пристрою хоста для існуючих IP адрес.

Транспортний рівень відповідальний за зв'язок кінцевих пристроїв та передачу даних без помилок. Він захищає програми вищого рівня від заплутаності інформації. Двома головними протоколами цього рівня є TCP та UDP. Перший з них відповідальний за безпомилковий зв'язок між кінцевими системами, проводить сегментацію і послідовне опрацювання даних. Протокол має функцію підтвердження та управляє потоками даних при допомозі механізму контролю потоків. Протокол володіє значним обсягом інформації, тому його реалізації потребує більше ресурсів. Другий є протоколом, котрий не надає таких можливостей, проте дозволяє передавати дані без підтвердження, саме тому вважається менш надійним.

Прикладний рівень забезпечує виконання функцій комунікації кінцевих мережевих пристроїв у контролює специфікації юзера. Містить у собі сукупність протоколів, зокрема: HTTP, HTTPS, FTP, TFTP, Telnet, SSH та інші [23].

2.4 Аналізатор пакетів

Аналізатор пакетів (сніфер пакетів) - частина апаратного чи програмного забезпечення, котра застосовується для спостереження за трафіком у мережі. Робота аналізатора пакетів – це дослідження потоків даних між комп'ютерами в одній мережі чи між мережевими пристроями та інтернетом. Аналізатори можна налаштувати на дослідження визначених пакетів при допомозі фільтрів або на перехоплення усіх існуючих пакетів.

Часто застосовуються компаніями для спостереження за використанням мережі працівниками, та є частиною значної кількості антивірусних програм.

Правильне використання аналізатора пакетів здатне допомогти в очищенні мережного трафіку та обмеженню впливу вірусних інфекцій для захисту від неавторизованого доступу [24] .

2.5 Вибір НД

НД для навчання та тестування було обрано CIC-IDS2017 [25]. Даний НД складається з нормального трафіку та найсучасніших атак, які зустрічаються в наш час. НД також містить згенерований реалістичний сторонній трафік, отриманий за допомогою системи B-Profile.

Ця система дозволяє скласти абстрактну поведінку користувачів та генерує фоновий трафік. Для цього НД було побудовано поведінку 25 користувачів на основі HTTP, HTTPS, FTP та SSH протоколів, а також протоколів електронної пошти.

Для отримання характеристик трафіку використовувався інструмент CICFlowMeter [6]. Ця програма використовується як генератор потоків трафіку та їх аналізатор. Застосунок здатний виділяти понад 80 статичних параметрів трафіку, серед них тривалість, кількість пакетів, кількість байтів, довжина пакетів та інші.

У програмі також можна вибрати окремі існуючі параметри, додавати нові і контролювати затримку потоків. Програма виводить результат як файл формату CSV.

Кожен запис у наборі CIC-IDS2017 характеризується 77 параметрами. Параметри наведені на рис. 2.3 [26].

No.	Feat. ID	Feature Names	Weight	No.	Feat. ID	Feature Names	Weight
1	41	Packet Length Std	0,638	40	17	Fwd Packet Length Std	0,280
2	13	Total Length of Bwd Packets	0,612	41	29	Bwd IAT Mean	0,271
3	65	Subflow Bwd Bytes	0,612	42	5	Fwd IAT Std	0,268
4	8	Destination Port	0,609	43	15	Fwd Packet Length Min	0,234
5	42	Packet Length Variance	0,577	44	38	Min Packet Length	0,231
6	20	Bwd Packet Length Mean	0,567	45	70	Active Mean	0,231
7	54	Avg Bwd Segment Size	0,567	46	27	Fwd IAT Min	0,229
8	18	Bwd Packet Length Max	0,560	47	73	Active Min	0,228
9	67	Init_Win_bytes_backward	0,554	48	69	min_seg_size_forward	0,227
10	12	Total Length of Fwd Packets	0,546	49	72	Active Max	0,226
11	63	Subflow Fwd Bytes	0,546	50	31	Bwd IAT Min	0,226
12	66	Init_Win_bytes_forward	0,542	51	23	Flow IAT Min	0,216
13	52	Average Packet Size	0,535	52	76	Idle Max	0,205
14	40	Packet Length Mean	0,526	53	74	Idle Mean	0,197
15	39	Max Packet Length	0,512	54	77	Idle Min	0,195
16	14	Fwd Packet Length Max	0,495	55	68	act_data_pkt_fwd	0,186
17	22	Flow IAT Max	0,467	56	6	Bwd IAT Std	0,179
18	36	Bwd Header Length	0,448	57	46	PSH Flag Count	0,106
19	9	Flow Duration	0,443	58	51	Down/Up Ratio	0,088
20	26	Fwd IAT Max	0,438	59	47	ACK Flag Count	0,069
21	55	Fwd Header Length	0,431	60	75	Idle Std	0,036
22	24	Fwd IAT Total	0,415	61	43	FIN Flag Count	0,033
23	25	Fwd IAT Mean	0,390	62	48	URG Flag Count	0,028
24	21	Flow IAT Mean	0,379	63	71	Active Std	0,025
25	2	Flow Bytes/s	0,379	64	44	SYN Flag Count	0,012
26	1	Bwd Packet Length Std	0,360	65	32	Fwd PSH Flags	0,012
27	64	Subflow Bwd Packets	0,355	66	45	RST Flag Count	0
28	11	Total Backward Packets	0,355	67	50	ECE Flag Count	0
29	16	Fwd Packet Length Mean	0,351	68	61	Bwd Avg Bulk Rate	0
30	53	Avg Fwd Segment Size	0,351	69	49	CWE Flag Count	0
31	19	Bwd Packet Length Min	0,324	70	57	Fwd Avg Packets/Bulk	0
32	3	Flow Packets/s	0,311	71	56	Fwd Avg Bytes/Bulk	0
33	37	Fwd Packets/s	0,309	72	34	Fwd URG Flags	0
34	30	Bwd IAT Max	0,306	73	33	Bwd PSH Flags	0
35	7	Bwd Packets/s	0,304	74	35	Bwd URG Flags	0
36	10	Total Fwd Packets	0,291	75	60	Bwd Avg Packets/Bulk	0
37	62	Subflow Fwd Packets	0,291	76	58	Fwd Avg Bulk Rate	0
38	28	Bwd IAT Total	0,287	77	59	Bwd Avg Bytes/Bulk	0
39	4	Flow IAT Std	0,281				

Рисунок 2.3 – Параметри трафіку, отримані за допомогою CICFLOWMETER

Набір містить інформацію про множину типів атак, наведених нижче.

1. DoS Hulk - 230 124 записів. Атака проводиться за допомогою програми Hulk DDoS Tool, що полягає в генерації трафіку у величезних обсягах. Згенерований трафік також проходить рушії кешування і атакує ресурси сервера безпосередньо [27] .

2. PortScan - 158804 записів. Атака спрямована на пошук доступних портів на сервері або в додатку і може містити обчислення для оцінки захищеності до загальнодоступних уразливостей або помилок конфігурації. Це досягається відправкою наперед визначеного трафіку до мети, і, на основі відповіді або його відсутності, сканер портів робить висновки [28].

3. DDoS – 128 025 записів.

4. DoS GoldenEye - 10293 записів. GoldenEye – інструмент HTTP

тестування. Може бути використаний для перевірки захищеності сайту до атак Deny of Service. Програма тестує захищеність за допомогою атак "HTTP Keep Alive + NoCache" [29] .

5. FTP-Patator - 7935 записів. Brute-Force атака за протоколом FTP [30] .

6. SSH-Patator - 5897 записів. Brute-Force атака за протоколом SSH [30] .

7. DoS slowloris - 5796 записів. Атака типу "відмова сервісу" в якій атакуюча програма дозволяє підтримувати безліч HTTP підключень між атакуючим і метою атаки [31] .

8. DoS Slowhttptest - 5499 записів. Атака, що щдійснюється за допомогою інструменту slowhttptest, що включає найбільш відомі низькорівневі атаки «відмова сервісу»: Slowloris; Slow HTTP Post; Slow Read Attack; Apache Range Header attack.

9. Bot - 1956 записів. Botnet атака за допомогою програмного пакету Ares, інструменту віддаленого доступу, написаного на Python [32] .

10. Web Attack Brute Force - 1507 записів.

11. Web Attack XSS – 652 записів.

12. Infiltration - 36 записів. Атака, спрямована на заповнення кінцевих ресурсів сервера, таких як місце на диску та пам'ять. Виконується за допомогою відкриття множинних підключень та ініціації запитів транзакції [33] .

13. Web Attack Sql Injection – 21 записів. Атака з використанням уразливостей, що дозволяють відправляти сторонні запити до баз даних програми [34] .

14. Heartbleed - 11 записів. Атака із використанням уразливості криптографічного алгоритму OpenSSL. Вразливість дозволяє читати пам'ять систем, що використовують вразливі версії програмного пакету OpenSSL, у тому числі секретні ключі, які застосовуються для ідентифікації провідників сервісу, трафік, що шифрують, імена і паролі користувачів, а також вміст пакетів [35] .

А також 2271320 записів нормального трафіку.

НД містить вісім csv файлах, поділених за днями та типами трафіку:

– понеділок; нормальна активність;

– вівторок, атаки та нормальна активність;

– середа, атаки та нормальна активність;

– четвер, атаки та нормальна активність;

– п'ятниця, атаки та нормальна активність

2.6 Вимоги до проектованого застосунку

2.6.1 Функціональні вимоги

Застосунок, котрий створюється, має задовольняти наступним функціональним вимогам:

– користувачу має надатися можливість вибору даних у вигляді файлу pcap , який буде аналізуватися;

– користуватись повинен мати можливість вибрати інтерфейс для аналізу пакетів у реальному часі;

– користувач повинен мати можливість вибрати навчену модель для аналізу трафіку;

– затосунок повинен класифікувати вхідні дані на 9 класів (DoS Slowhttptest, DoS slowloris, SSH-Patator, FTP-Patator, DoS GlodenEye, DDoS, PortScan, DosHulk та BENIGN);

– програма повинна виводити інформацію про результат класифікації та помилки користувача.

2.6.2 Нефункціональні вимоги

Застосунок, що розробляється, повинен задовольняти наступним нефункціональним вимогам:

– програма має бути розроблена мовою Python;

– у застосунку мають бути реалізовані два ансамблеві алгоритми машинного навчання;

– інтерфейс програми має бути консольним.

2.7 Варіанти використання

Було сформовано діаграму варіантів використання, наведену на рис. 2.4.

На ній представлений головний актор - «Користувач», який взаємодіє із системою. Користувач має два основні варіанти використання системи (вибору режиму роботи): "в реальному часі" – режим 1, " з локальними файлами" – режим 2..

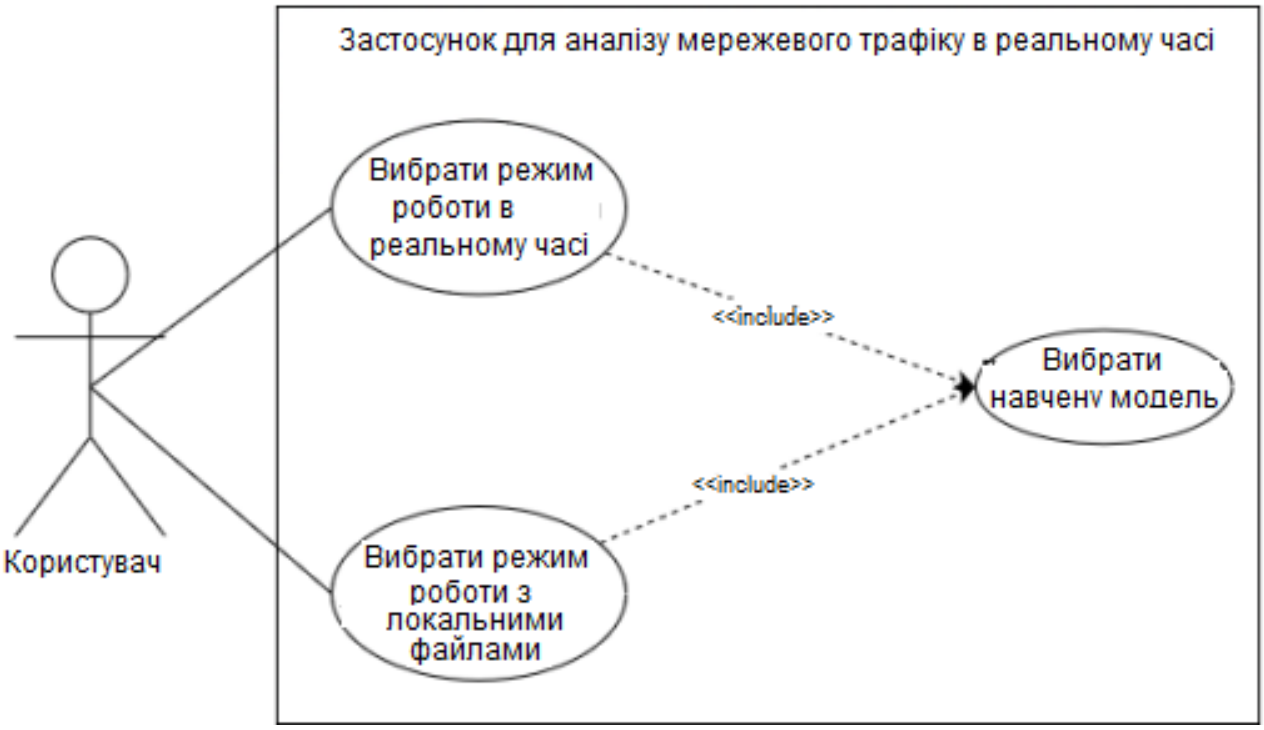


Рисунок 2.4 – Діаграма варіантів використання

У табл. 2.1. представлена специфікації варіанту використання «Вибрати роботу в реальному часі»

Таблиця 2.1 - Специфікація варіанту використання режиму 1

Прецедент: Вибрати роботу в реальному часі
ID: 1
Короткий опис: Користувач вибирає режим роботи програми у реальному часі
Головні актори: Користувач
Другорядні актори: Немає
Передумови: Застосунок запущено
Основний потік: 1. Прецедент починається, коли користувач запускає програму. 2. Програма виводить меню з варіантами вибору. 3. Користувач вибирає режим роботи у реальному часі. 4. Користувач вибирає інтерфейс для роботи
Післяумови: 1. Програма переходить до меню вибору моделі
Альтернативні потоки: Немає

Специфікації варіанту використання «Вибрати режим роботи з локальними файлами» наведена відповідно у табл. 2.1.

Таблиця 2.2 - Специфікація варіанту використання режиму 2

Прецедент: Вибрати режим роботи з локальними файлами
ID: 2
Короткий опис: Користувач вибирає режим роботи з локальними файлами
Головні актори: Користувач
Другорядні актори: Немає
Передумови: Застосунок запущено
Основний потік: 1. Прецедент починається, коли користувач запускає програму. 2. Програма виводить меню з варіантами вибору. 3. Користувач вибирає режим роботи з локальними файлами. 4. Користувач вибирає файл
Післяумови: 1. Програма переходить до меню вибору моделі
Альтернативні потоки: Немає

2.8 Діаграма компонентів

Також було розроблено діаграму компонентів, що відображає розбиття об'єктів системи та взаємозв'язку між компонентами. Розроблена діаграма представлена на рис. 2.5.

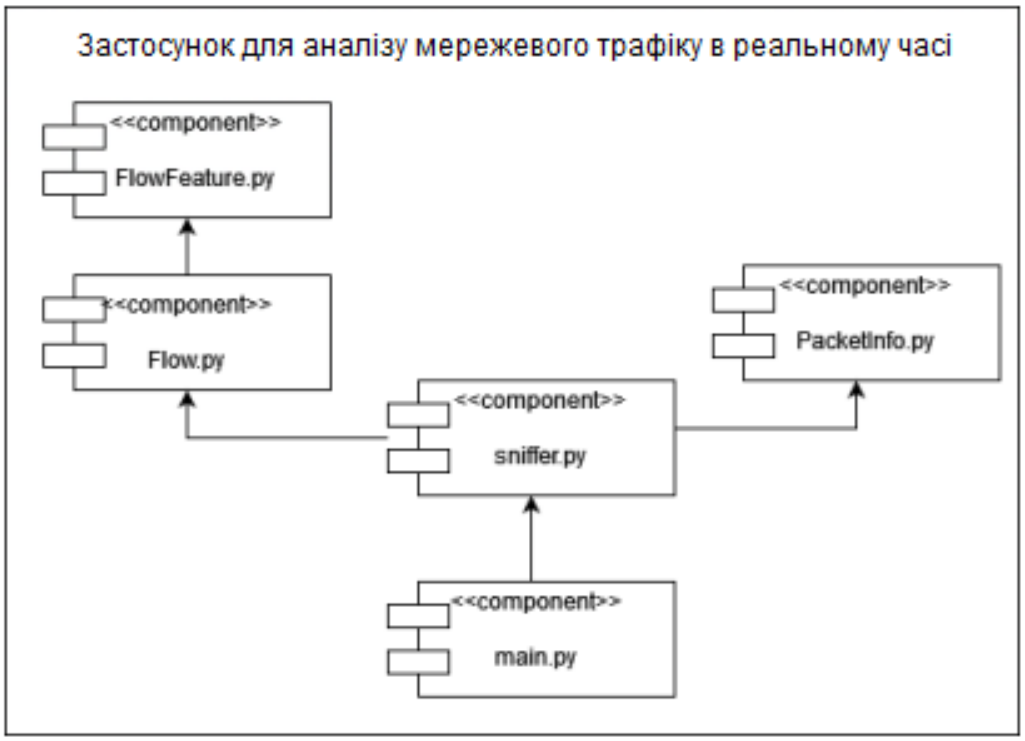


Рисунок 2.5 – Діаграма компонентів застосунку

Система складається з наступних компонентів (програмних модулів):

– main.py - відповідає за вибір параметрів і запуск процесу аналізу трафіку;

– sniffer.py - реалізовані компоненти програми, що відповідають за процес аналізу трафіку, виведення результатів у файл та в консоль користувача;

– Flow.py - реалізовано структуру мережного потоку та методи обчислення параметрів трафіку, необхідних для класифікації;

– FlowFeature.py - описана структура даних мережевих потоків та методи встановлення та отримання цих даних;

– PacketInfo.py - описана структура даних мережевих пакетів та методи встановлення та отримання цих даних.

2.9 Висновки до другого розділу

У другому розділі були описані такі пункти, як постановка задачі класифікації трафіку, пропоновані до розробки ансамблеві методи машинного

навчання, розгляд мережевої моделі TCP/IP, визначення аналізатора пакетів, а також проведено аналіз вихідного НД.

Також було визначено вимоги до застосунку (як функціональні, так і нефункціональні). На основі цих вимог було розроблено діаграму варіантів використання. Також було розроблено діаграма компонентів, що відображає архітектуру розроблюваного застосунку.

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ РОЗРОБКИ

3.1 Середовище виконання та програмні засоби розробки

Як основна мова для програмної частини була обрана високорівнева мова Python версії 3.9.1. Ця мова має багато бібліотек, створених для спрощення роботи з машинним навчанням.

Для реалізації ансамблевих моделей було обрано бібліотеку scikit-learn [36]. Ця бібліотека містить готові рішення щодо реалізації алгоритмів машинного навчання, а також безліч допоміжних функцій та інструментів для обробки даних та результатів моделювання. Також було використано бібліотеку pandas [37] для спрощення роботи з таблицями.

Як середовище розробки, навчання та налагодження моделі машинного навчання було застосовано сервіс Google Colaboratory.

Google Colaboratory – хмарний сервіс компанії Google, що надає середовище виконання Python коду у вигляді Jupyter Notebook. Дозволяє запускати код в окремий осередок. Сервіс спрямований на спрощення досліджень у галузі машинного навчання. Багато бібліотек, спрямованих на розробку в цій галузі, вже встановлено. Також сервіс надає великі обчислювальні можливості виконання коду [38] .

Розробка програми аналізатора пакетів велася у середовищі розробки PyCharm версії 2021.2.1 [39]. Аналізатор пакетів реалізований за допомогою бібліотеки Scapy [40] версії 2.4.5.

3.2 Передобробка НД

Для більш ефективної роботи моделей було проведено попередню обробку вихідного НД, що містить декілька етапів.

З вихідного НД було прибрано записи, що містять поля без значень. Потім, з набору були видалені атрибути, що повторюються, і атрибути, що містять єдине значення (рис. 3.1). Таким чином кількість атрибутів було скорочено до 65

параметрів.

```
column_names = np.array(list(X))
to_drop = []
for x in column_names:
    size = X.groupby([x]).size()
    if (len(size.unique()) == 1):
        to_drop.append(x)
to_drop
```

Рисунок 3.1 – Фрагмент лістингу функції відбору параметрів

Вихідний НД містить різну кількість записів для класів (рис. 3.2), тому для НД було проведено балансування. Балансування полягає у вибірці однакової чи схожої кількості записів для кожного класу. Попередньо з набору було прибрано класи, що містять малу кількість записів: Heartbleed; Web Attack Sql Injection; Infiltration; Web Attack XSS; Web Attack Brute Force; Bot.

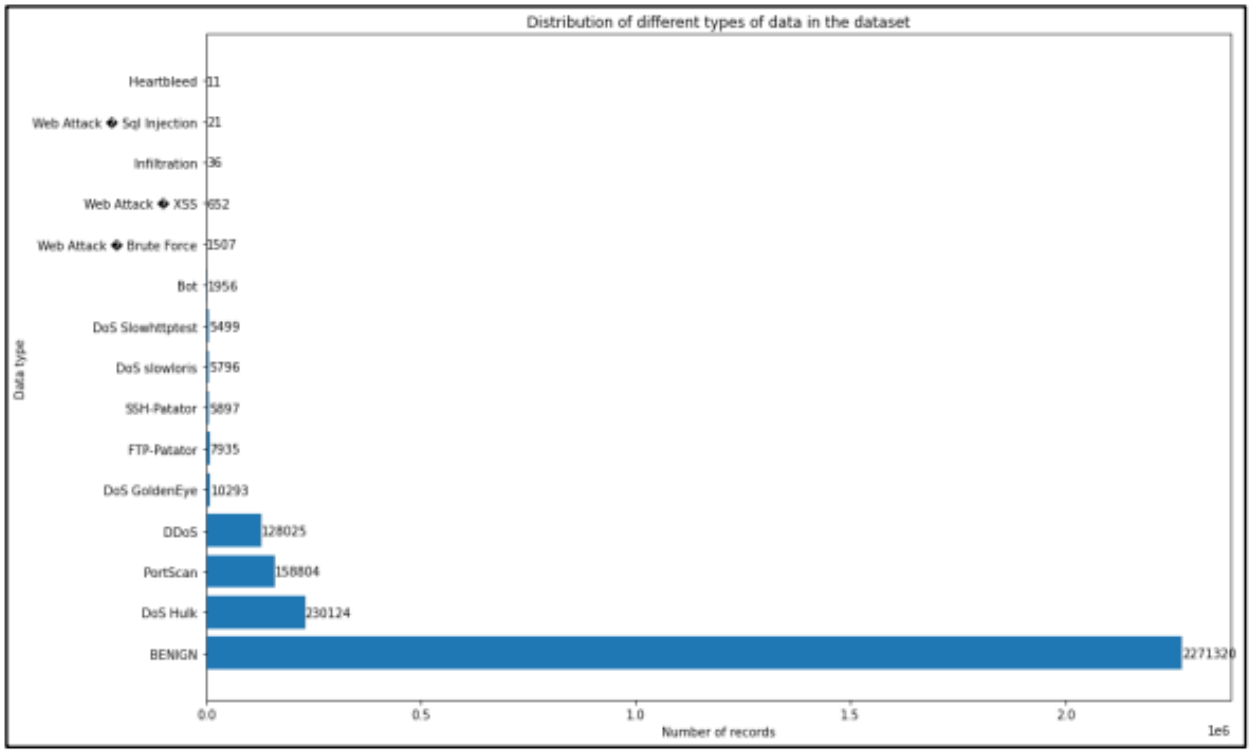


Рисунок 3.2 – Розподіл записів у НД

У ході балансування було обрано 5499 записів для наступних класів:

– DoS Slowhttptest;
– DoS slowloris;
– SSH-Patator;
– FTP-Patator;
– DoS GlodenEye;
– DDoS;
– PortScan;
– DosHulk;
– BENIGN.

Подальша передобробка полягала у виділенні корисної інформації із загальної кількості параметрів. Виділення ознак проводилося за допомогою методу SelectKBest [41] із пакету scikit-learn. Як функцію відбору параметрів була використана chi2. НД був нормалізований за допомогою функції MinMaxScaler [42] , яка перетворює дані для кожного стовпця в проміжок від 0 до 1. Процес відбору ознак представлений на рис. 3.3. Результати відбору параметрів наведені на рис. 3.4.

```
min_max = MinMaxScaler().fit(x_train)
x_train_copy_test = min_max.transform(x_train)
x_test_copy_test = min_max.transform(x_test)
features = SelectKBest(score_func=chi2, k=x_train_copy_test.shape[1])
fit = features.fit(x_train_copy_test, y_train)
feature_importances = zip(dataset_copy.columns[:-1], features.scores_)
feature_importances = sorted(feature_importances, key = lambda x: x[1],
reverse = True)
sorted_importances = [importance[1] for importance in feature_importances]
sorted_features = [importance[0] for importance in feature_importances]
x_values = list(range(len(feature_importances)))
cumulative_importances = np.cumsum(sorted_importances)
value99 = cumulative_importances[-1]*0.99
plt.figure(figsize=(30,15))
plt.plot(x_values, cumulative_importances)
plt.hlines(y = value99, xmin=0, xmax=len(sorted_importances), color = 'r',
linestyles = 'dotted')
plt.xticks(x_values, sorted_features, rotation = 'vertical', fontsize=17)
plt.yticks([], [])
```

Рисунок 3.3 – Фрагмент лістингу коду відбору ознак та побудови графіку

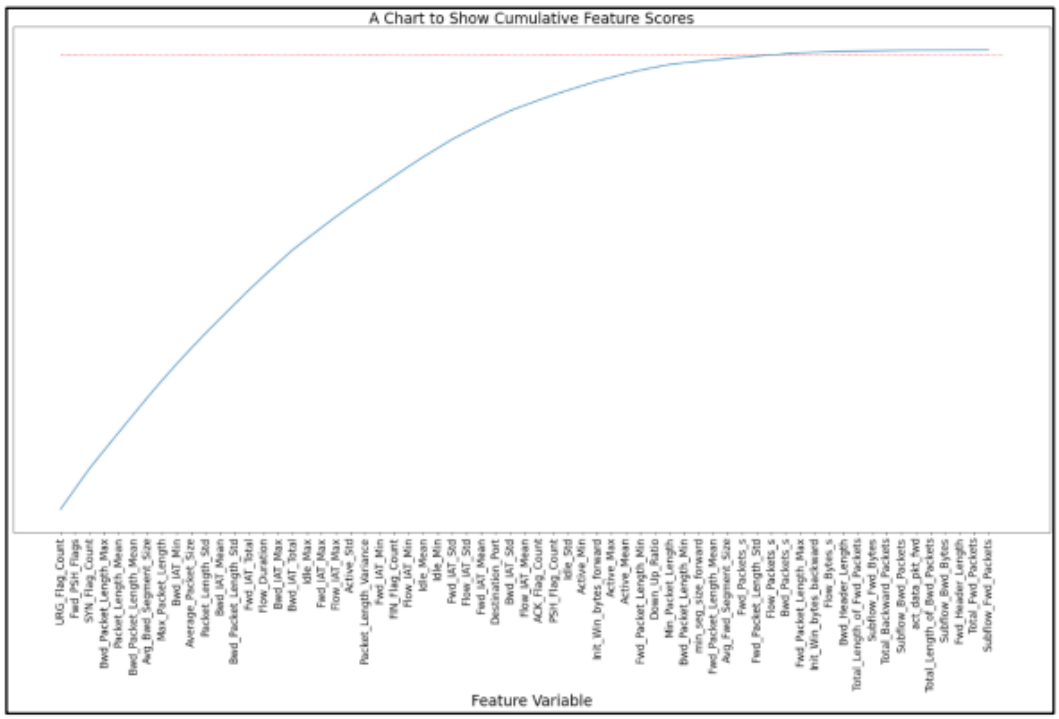


Рисунок 3.4 – Графік накопичувальної суми важливості параметрів

Червона лінія на рис. 3.4 означає 99% залишкової важливості, параметри вище цієї лінії не дають значної інформації. Виходячи з даних графіка, для класифікації було виділено 38 найбільш значних параметрів інтернет-трафіку. Список отриманих параметрів представлено на рис. 3.5.

```
Index(['Destination_Port', 'Flow_Duration', 'Bwd_Packet_Length_Max',
       'Bwd_Packet_Length_Mean', 'Bwd_Packet_Length_Std', 'Flow_IAT_Mean',
       'Flow_IAT_Std', 'Flow_IAT_Max', 'Flow_IAT_Min', 'Fwd_IAT_Total',
       'Fwd_IAT_Mean', 'Fwd_IAT_Std', 'Fwd_IAT_Max', 'Fwd_IAT_Min',
       'Bwd_IAT_Total', 'Bwd_IAT_Mean', 'Bwd_IAT_Std', 'Bwd_IAT_Max',
       'Bwd_IAT_Min', 'Fwd_PSH_Flags', 'Max_Packet_Length',
       'Packet_Length_Mean', 'Packet_Length_Std', 'Packet_Length_Variance',
       'FIN_Flag_Count', 'SYN_Flag_Count', 'PSH_Flag_Count', 'ACK_Flag_Count',
       'URG_Flag_Count', 'Average_Packet_Size', 'Avg_Bwd_Segment_Size',
       'Init_Win_bytes_forward', 'Active_Std', 'Active_Min', 'Idle_Mean',
       'Idle_Std', 'Idle_Max', 'Idle_Min'],
      dtype='object')
```

Рисунок 3.5 – Отримання списку нових параметрів

3.3 Реалізація ансамблевих алгоритмів

3.3.1 Реалізація Random Forest

Модель вдалося втілити при допомозі класу Random Forest Classifier [20],

котрий входить до бібліотеки scikit-learn. Вхідними параметрами для ініціалізації моделі було вибрано наступні:

– n_estimators – число випадкових дерев, що створюються;

– random_state – параметр контролю ймовірності bootstrap вибірки і ймовірність поділу параметрів коли дерева формуються.

Для підбору значення n_estimators необхідно було провести тестування моделі з різними значеннями, у ході якого навчалася з різним значенням параметра. За наслідком виконаного тестування було сформовано графік помилки моделей, показаний на рис. 3.6.

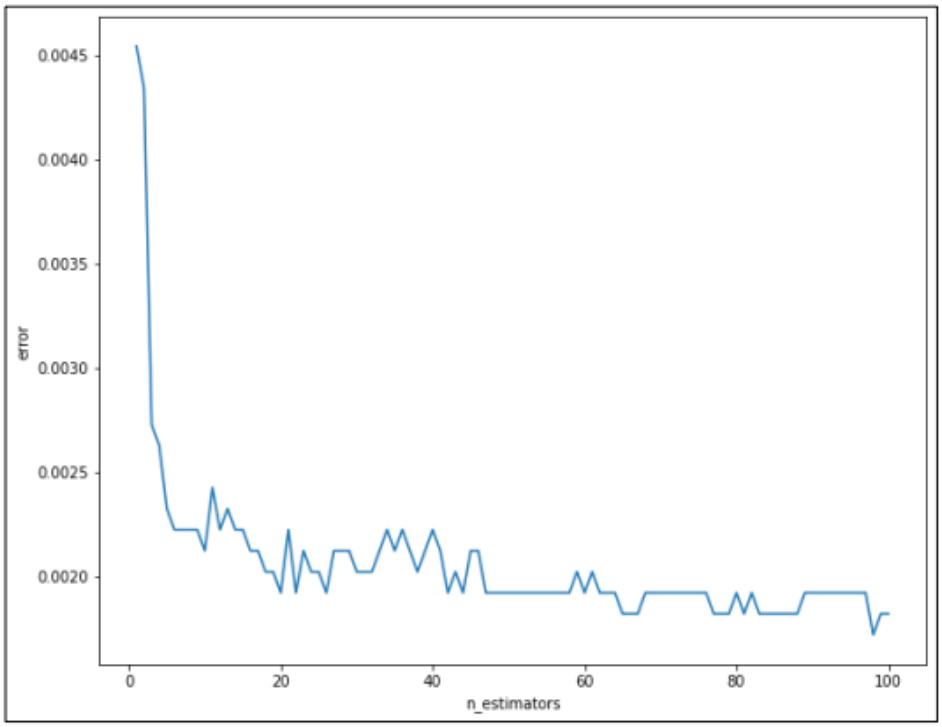


Рисунок 3.6 – Графік помилки Random Forest

За підсумками тестування для параметра вибрали значення 98, при цій кількості дерев модель має мінімальну помилку класифікації. Для можливості відтворення експериментів та співставлення якості наведеної моделі із іншими, потрібно було використати параметр random_state із значенням 42. Ті параметри, котрі залишилися, розглянуті в документації [20] . Їх було залишено як є і надано їм стандартних значень.

На рис. 3.7 наведено код реалізації моделі, навченої за допомогою

RandomForest.

```
model = RandomForestClassifier(n_estimators=45, random_state=42)
x_train_copy, x_test_copy, feature = preproc(x_train, x_test, y_train, 38)
#print(dataset_copy.columns[feature.get_support(indices=True)])
model.fit(x_train_copy, y_train)
y pred rf = model.predict(x test copy)
results rf = model.score(x test copy, y test)
print("Random Forest results", results rf)
print("Confusion matrix for Random Forest\n", confusion matrix(y test,
y_pred_rf))
```

Рисунок 3.7 – Лістинг коду моделі RandomForest

3.3.2 Реалізація AdaBoost

Модель була реалізована за допомогою класу AdaBoost Classifier [21] із бібліотеки scikit-learn. Як вхідні параметри для ініціалізації моделі використовуються такі:

– n_estimators – кількість створюваних випадкових дерев;

– random_state - параметр, що контролює випадковість bootstrap вибірки та випадковість поділу атрибутів при побудові дерев;

– base_estimator - параметр, який передається базовий класифікатор для побудови ансамблевої моделі.

Для вибору значення вхідного параметра n_estimators було проведено тестування моделі з різним значенням вхідного параметра. За результатами тестування було побудовано графік залежності помилки моделі, представлений на рис. 3.8. Для параметра встановлено значення 38, при цьому значенні модель має мінімальну помилку. Для можливості відтворення експериментів та порівняння якості представленої моделі з іншими параметру random_state було присвоєно значення 42.

Базовим класифікатором для моделі був вказаний DecisionTreeClassifier [43] з максимальною глибиною дерева 5. Параметри, що залишилися, описані в документації [21] , залишені без змін і приймають стандартні значення.

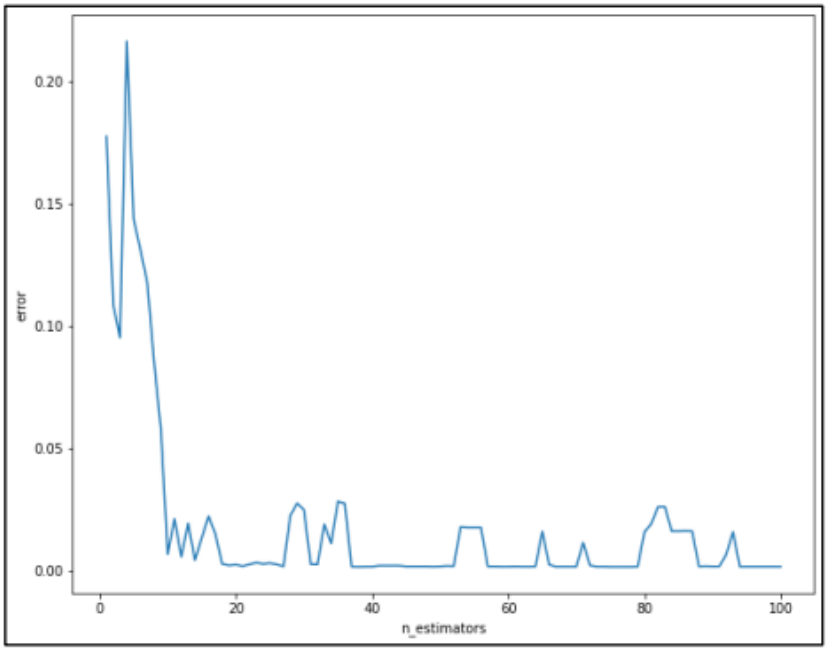


Рисунок 3.8 – Графік помилки моделі AdaBoost

На рис. 3.9 представлено лістинг коду реалізації моделі, навченої за допомогою алгоритму AdaBoost.

```
model_ab = AdaBoostClassifier(n_estimators=42, random_state=42,
base_estimator=DecisionTreeClassifier(max_depth=5))
x_train_copy, x_test_copy, feature = preproc(x_train, x_test, y_train, 38)
#print(dataset copy.columns[feature.get support(indices=True)])
model ab.fit(x train copy, y train)
y pred ab = model ab.predict(x test copy)
results ab = model ab.score(x test copy, y test)
print("AdaBoost results", results_ab)
print("Confusion matrix for AdaBoost\n", confusion_matrix(y_test,
y_pred_ab))
```

Рисунок 3.9 – Лістинг коду моделі AdaBoost

3.4 Оцінка якості алгоритмів

Для навчання та подальшої оцінки якості вихідний НД був поділений на дві частини: навчальну вибірку; тестову вибірку. У навчальній вибірці міститься 80% НД, тестова вибірка містить 20% набору.

За наслідками навчання можна отримати метрики якості навчання. Як метрик навчання були вибрані score та confusion_matrix. Дані метрики були імпортовані з бібліотеки scikit-learn.

Score є вбудованим методом класифікаторів, який повертає середнє значення точності передбачення, порівнюючи передбачені моделями мітки та дійсними мітками даних.

Confusion_matrix - функція, яка будує двовимірну матрицю *C*, в якій кожен елемент C_{ij} відображає кількість записів, для яких відомий клас *i*, та яким був присвоєний клас *j*. Коректно передбачені елементи знаходяться на перетині номера рядка та номера стовпця. Таким чином, чітко передбачені записи знаходяться на діагоналі матриці. За матрицею збурень можна оцінити у яких випадках модель працює коректно, і з якими класами помиляється.

Результати навчання та тестування моделей представлені в табл. 3.1.

Таблиця 3.1 – Результати навчання моделей

Алгоритм	Результат	Матриця збурень
RandomForest	99,79	[[1117 1 0 0 0 0 0 0 1] [2 1143 1 0 0 0 0 0 0] [0 0 1086 0 1 0 0 0 0] [0 0 0 1094 0 0 0 2 0] [2 0 1 0 1109 5 0 0 0] [2 0 0 0 1 1103 0 1 0] [0 0 0 0 0 0 1111 0 0] [0 0 0 1 0 0 0 1065 0] [0 0 0 0 0 0 0 0 1050]]
AdaBoost	99,81	[[1117 1 1 0 0 0 0 0 0] [1 1145 0 0 0 0 0 0 0] [0 0 1086 0 1 0 0 0 0] [0 0 0 1096 0 0 0 0 0] [0 0 1 0 1107 9 0 0 0] [1 0 0 0 3 1102 0 1 0] [0 0 0 0 0 0 1111 0 0] [0 0 0 0 0 0 0 1066 0] [0 0 0 0 0 0 0 0 1050]]

3.5 Реалізація аналізатора пакетів

Для реалізації аналізатора пакетів було використано бібліотеку Scapy [40] . Отримання інформації з необроблених пакетів було реалізовано за допомогою функції обробки пакетів new_packet. При отриманні нового пакета функція-обробник намагається отримати його дані. а потім надає його певному існуючому мережному потоку. Якщо потоку з ID пакета, що обробляється, не існує, то функція-обробник створює новий потік. У програмі заданий час життя мережного потоку, якщо потік не діє більше заданого часу, то потік знищується і створюється новий на основі останнього отриманого пакета. Якщо пакет містить прапор закінчення потоку, потік видаляється і створюється новий потік на основі останнього отриманого пакета. Цей процес виконується як для вхідних, так вихідних потоків.

Обробник для опису пакетів та потоків використовує класи PacketInfo та Flow.

Клас PacketInfo описує структуру зберігання параметрів пакетів і містить у собі методи встановлення та отримання різних параметрів пакета. Параметри встановлюються шляхом перетворення інформації з рівнів TCP/IP архітектури, отриманої методом sniff з бібліотеки Scapy. Клас Flow описує методи збирання та обробки статистичної інформації мережевих потоків, які необхідні для класифікації трафіку. Для зберігання інформації про потік використовують клас FlowFeature. Фрагмент вихідного код класу представлений на рис. 3.10.

3.6 Реалізація інтерфейсу взаємодії

Було реалізовано інтерфейс взаємодії користувача із застосунком. На рис. 3.12 продемонстровано лістинг вихідного модуля main. У цьому модулі користувач може самостійно вказати режим роботи програми та вибрати параметри, котрі необхідні для подальшої роботи програми.

При виборі режиму роботи з локальними файлами користувачеві буде доступний список доступних файлів для аналізу, котрі зберігаються в робочій

папці програми. А при вказанні режиму роботи в реальному часі користувачеві виводиться список доступних для прослуховування інтерфейсів. Вибір файлу або інтерфейсу проходить за допомогою цифр з клавіатури.

```
def main():
    mode = input("Enter s to sniff, or p for pcap analysis. Default will be
to sniff traffic.")
    if mode == 'p':
        right = False
        filelist = [f for f in glob.glob('*.pcap*')]
        print(*[str(j + 1) + ' - ' + f for j, f in enumerate(filelist)],
sep='\n')
        while not right:
            try:
                a = int(input('Choose file - '))
                if a <= len(filelist):
                    right = True
                    continue
                else:
                    raise ValueError
            except ValueError:
                print('Wrong Input')
        f = filelist[a-1]
        sniffer.main(1, f, ' ')
    else:
        print("Choose interface for sniffing".center(40, ' '))
        ifaces = list(psutil.net_if_addrs().keys())
```

Рисунок 3.12 – Лістинг фрагменту коду функції main

Далі з файлу sniffer.py викликається функція main, в котрій здійснюється завантаження моделі, перетворювача вхідних даних та виклик функції аналізатора. На рис. 3.13 – 3.15 представлений вихідний код функцій.

```
def main(mode, pcap_file, iface):
    global classifier
    global normalization
    print("Loading model ".center(20, '~'))
    classifier = load_model()
    normalization = load_scalar()
    print(" Sniffing ".center(20, '*'))
    if mode == 0:
        live(iface)
    else:
        pcap(pcap_file)
```

Рисунок 3.13 – Код функції main (оновленої)

```
def load model():
    filelist = [f for f in glob.glob('* model.sav')]
    print(*[str(j + 1) + ' - ' + m for j, m in enumerate(filelist)], sep='\n')
    right = False
    while not right:
        try:
            ch_f = int(input("Choose model - "))
            if ch_f <= len(filelist):
                right = True
                break
            else:
                raise ValueError
        except ValueError:
            print('Wrong Input')
    model = filelist[ch_f-1]
    with open(model, 'rb') as m:
        return pickle.load(m)
```

Рисунок 3.14 – Завантаження навченої моделі

```
def load scalar():
    with open('scalar.sav', 'rb') as s:
        return pickle.load(s)
```

Рисунок 3.15 – Код функції scalar

Як видно з рис. 3.14, перед завантаженням моделі проходить перевірка введення даних користувача.

У випадку неправильно введених даних програма продемонструє повідомлення про помилку і порекомендує користувачеві знову їх ввести. На рис. 3.16 показаний процес введення невірних даних та реакція застосунку на введені дані.

```
Enter s to sniff, or p for pcap analysis. Default will be to sniff traffic.s
Choose interface for sniffing
1 -  Ethernet
2 -  Під’єднання по локальній мережі*  1
3 -  Під’єднання по локальній мережі * 2
4 – Безпровідна мережа
5 - Loopback Pseudo-Interface 1
Choose interface - s
Wrong input
Choose interface – б
 Wrong input
Choose interface - 4
– Loading model –
1 -  ab_model.sav
2 –  rf_model.sav
3 –  test_model.sav
Choose model - s
Wrong Input
Choose model - l
***** Sniffing *****
```

Рисунок 3.16 – Ввід неправильних даних

3.7 Тестування розробки

3.7.1 Функціональне тестування

Є тестуванням ПЗ з метою перевірки втіленості функціональних вимог, чи по іншому, можливості програми за визначених умов розв’язувати завдання, котрі потрібні користувачам.

У табл. 3.2. наведено набір тестів на функціональність.

Таблиця 3.2 – Функціональне тестування

Назва тесту	Кроки	Очікуваний результат	Тест пройдено?
Вибір файлу	1. У меню ввести літеру p. 2. У меню ввести цифру необхідного файлу.	Програма повинна перейти до меню вибору	Так
Вибір інтерфейсу	1. У меню введіть літеру s. 2. У новому меню введіть цифру з номером необхідного інтерфейсу.	Програма повинна перейти до меню вибору моделі.	Так
Вибір моделі	1. У меню виберіть спосіб роботи. 2. У новому меню виберіть інтерфейс або файл. 3. У новому меню введіть цифру з номером моделі.	Програма має запустити процес аналізу та класифікації.	Так
Введення неправильних параметрів	1. У меню вибору інтерфейсу ввести значення не зі списку доступних інтерфейсів.	Програма має вивести повідомлення про помилку Wrong input.	Так

3.7.2 A/B тестування

Такий вид тестування дозволяє оцінити якісні показники функціонування кількох варіантів моделі, а також співставити їх між собою [44] .

Під час A/B тестування було виконано кілька експериментів щодо навчання моделей за різної кількості даних із вихідного НД. Експерименти проводилися для визначення найкращого балансування для обраних класів атак. Також моделі навчалися на незбалансованій вибірці, ці експерименти були здійснені з метою підтвердження потреби у балансуванні. Результати експериментів поазані у табл. 3.3.

Таблиця 3.3 – Результати A/B тестування

Кількість даних (записів)	Алгоритм	Score
1000	RandomForest	99,28
	AdaBoost	99,44
2000	RandomForest	99,67
	AdaBoost	99,67
3000	RandomForest	99,74
	AdaBoost	99,35
4000	RandomForest	99,82
	AdaBoost	98,55
5499	RandomForest	99,81
	AdaBoost	99,86
Немає балансування	RandomForest	99,77
	AdaBoost	93,99

При проведенні експериментів було встановлено, що балансування потрібне для збільшення точності прогнозування. На НД, який є незбалансованим, показники метрики score були нижчими, аніж на збалансованій вибірці. Матриці збурення для НД без балансування свідчать, що через різну кількість вихідних даних модель починає дуже помилятися у деяких класах.

3.7.3 Підбір параметрів моделі

Було досліджено залежність величини помилки від числа параметрів і числа слабких учнів. На рис. 3.17 наведено графік залежності помилки моделі, котра була навчена алгоритмом Random Forest.

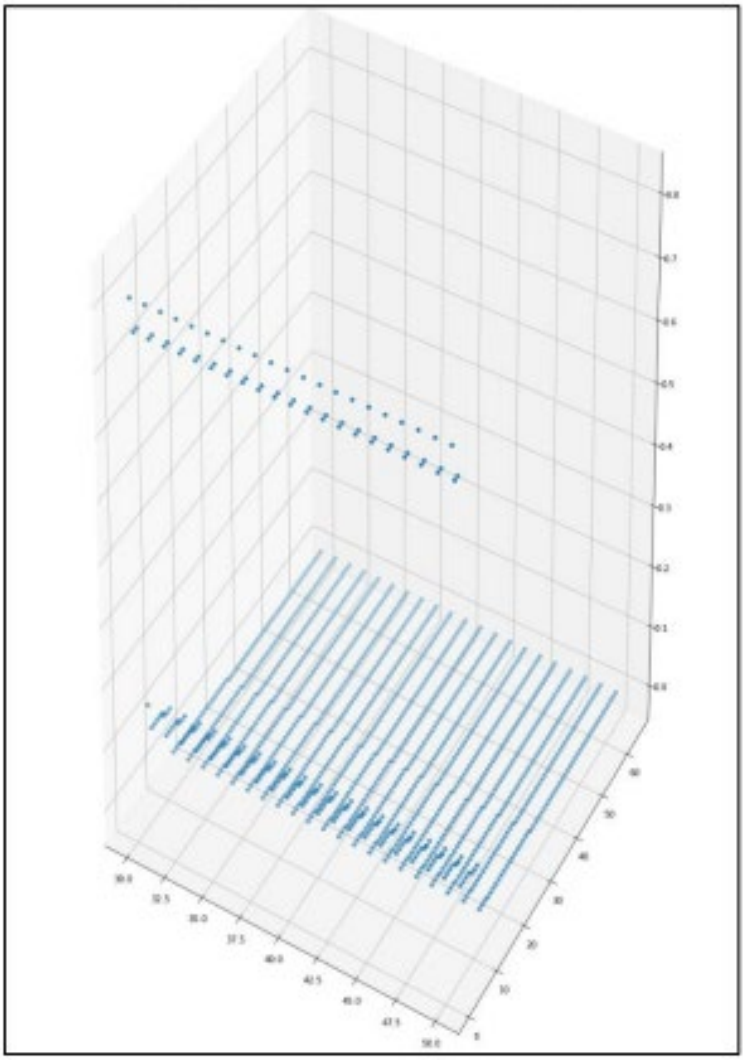


Рисунок 3.17 – Графік залежності помилки класифікації від параметрів моделі

На горизонтальних осях графіка відображено параметри n_estimators, котрі налаштовуються, і кількість параметрів трафіку, котрі відбираються. Вертикальна вісь показує помилку класифікації за заданих параметрів. За побудованим графіком було зроблено висновок, що кількість дерев істотно впливає на помилку класифікації. Кількість параметрів, котрі відбираються, істотно впливає на кількість помилки в діапазоні від 1 до 37 параметрів, при кількості параметрів > 38 значення помилки виходить на плато. Беручи це до уваги можна дійти до висновку, що вибрані параметри ініціалізації моделі на чолі реалізації, є оптимальними.

3.7.4 Тестування аналізатора пакетів

Для тестування роботи сніффера був використаний НД HTTP DoS Dataset

в форматі PCAP for Wireshark [45] , в котрому міститься інтернет-трафік, одержаний під час атак. НД складається з таких атак,я к Slow Read, Slow Post, Slow Get, HULK, Goldeneye та звичайного трафіку. Це збігається з атаками, котрим навчена модель. На рис. 3.18 наведено результат роботи аналізатора пакетів.

Як видно із рис. 3.18 застосунок детектує та класифікує підозрілий трафік, що свідчить про його успішне функціонуванн.

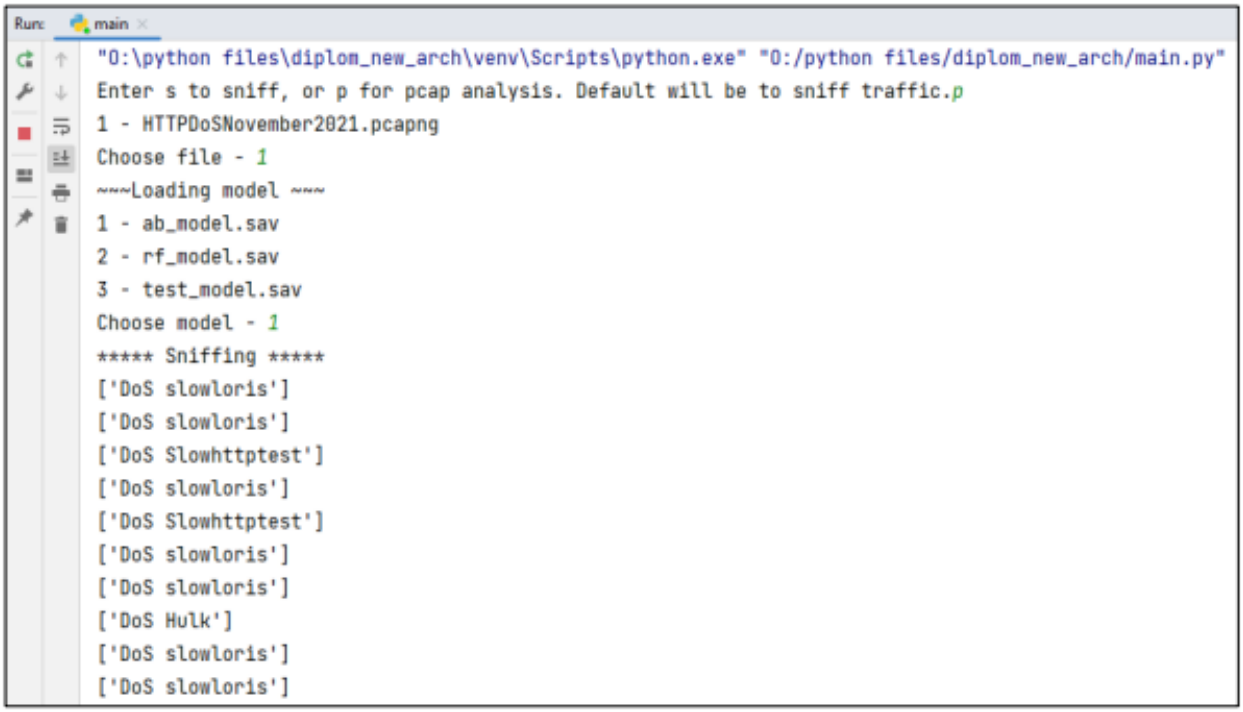


Рисунок 3.18 – Робота аналізатора пакетів

Розроблений застосунок було протестовано в режимі реального часу на локальному інтерфейсі. Було відкрито декілька сайтів у браузері, з яких надсилався нормальний трафік. Результати проведеного тестування відображені на рис. 3.19.

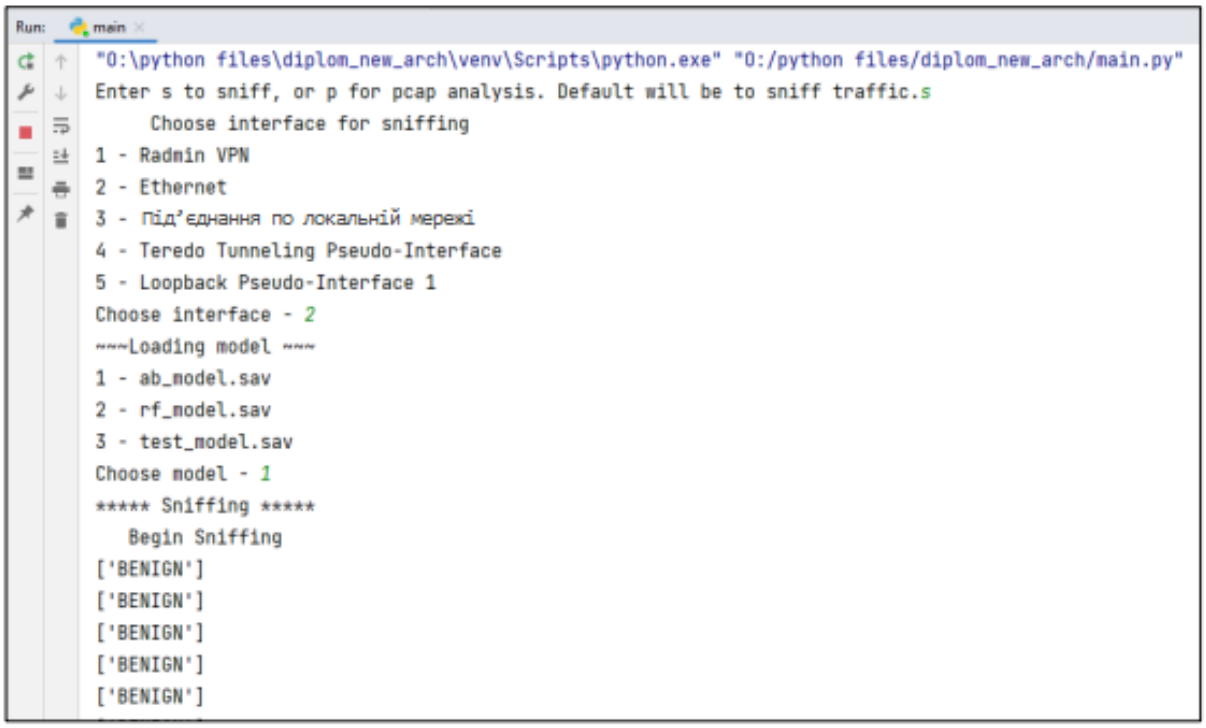


Рисунок 3.19 – Робота застосунку на звичайному трафіку

Як видно із рис. 3.19, застосунок успішно зчитує інформацію, котра надходить в реальному часі з інтерфейсів і здатен достовірно класифікувати звичайний трафік.

3.8 Висновки до третього розділу

Відповідно до поставлених завдань було реалізовано компоненти застосунку, а також проведено передобробку вихідного НД. Були реалізовані ансамблеві моделі, аналізатор пакетів, а також користувальницький інтерфейс програми.

Було проведено функціональне тестування застосунку, під час якого вся функціональність системи була успішно протестовано. Було виконано A/B тестування, в якому співставлялася точність класифікації моделей за різної кількості записів з НД. Це дало змогу що виявити найкраще балансування для обраних класів атак. Також у ході A/B тестування доведено потребу у балансуванні даних, за відсутності балансування модель значно гірше змогла класифікувати окремі класи. Було проведено тестування залежності помилки

класифікації моделі від кількості вхідних параметрів трафіку, за результатами виконаного аналізу зроблено висновки щодо вибраних параметрів ініціалізації моделей. Було проведено тестування роботи програми в режимі реального часу та при опрацюванні локальних файлів, під час якого програма успішно класифікувала як звичайний трафік із сайтів у реальному часі, так само і шкідливий трафік під час зчитування інформації із файлу.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Охорона праці

Метою кваліфікаційної роботи магістра є проведення ґрунтовного аналізу мережного трафіку в режимі реального часу на основі ансамблевих методів машинного навчання та створення програмного застосунку для цього. Оскільки, проведення такого виду робіт передбачає використання комп'ютерної техніки, зокрема ПК та периферійних пристроїв, то обов'язковим є дотримання вимог з охорони праці і техніки безпеки.

Для ефективної і безпечної роботи колективу працівників для проведення ґрунтовного аналізу мережного трафіку в режимі реального часу на основі ансамблевих методів машинного навчання та створення програмного застосунку для цього, необхідно організувати безпечні умови праці. При цьому керівник організації несе безпосередню відповідальність за порушення нормативно-правових актів з охорони праці [34]. Окрім цього, на робочих місцях працівників необхідно забезпечити дотримання вимог, затверджених Наказом Мінсоцполітики від 14.02.2018 за № 207 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями». Згідно Вимог приміщення, де розміщені робочі місця операторів, крім приміщень, у яких розміщені робочі місця операторів великих ЕОМ загального призначення (сервер), мають бути оснащені системою автоматичної пожежної сигналізації відповідно до цих вимог;

– переліку однотипних за призначенням об'єктів, які підлягають обладнанню автоматичними установками пожежогасіння та пожежної сигналізації, затвердженого наказом Міністерства України з питань надзвичайних ситуацій та у справах захисту населення від наслідків Чорнобильської катастрофи від 22.08.2005 N 161, зареєстрованого в Міністерстві юстиції України 05.09.2005 за N 990/11270 (НАПБ Б.06.004-2005);

– Державних будівельних норм "Інженерне обладнання будинків і споруд. Пожежна автоматика будинків і споруд", затверджених наказом Держбуду

України від 28.10.98 N 247 (далі - ДБН В.2.5-56:2014, з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками.

В інших приміщеннях допускається встановлювати теплові пожежні сповіщувачі. Приміщення, де розміщені робочі місця операторів, мають бути оснащені вогнегасниками, кількість яких визначається згідно з вимогами ДСТУ 4297:2004 «Пожежна техніка. Технічне обслуговування вогнегасників». Загальні технічні вимоги і з урахуванням граничнодопустимих концентрацій вогнегасної рідини відповідно до вимог НАПБ А.01.001-2014. Приміщення, в яких розміщуються робочі місця операторів сервера загального призначення, обладнуються системою автоматичної пожежної сигналізації та засобами пожежогасіння відповідно до вимог ДБН В.2.5-56:2014, ДБН В.2.5-56:2010, НАПБ А.01.001-2014 і вимог нормативно-технічної та експлуатаційної документації виробника. Проходи до засобів пожежогасіння мають бути вільними.

Лінія електромережі для живлення комп'ютера та периферійних пристроїв повинні бути виконаними як окрема групова трипровідна мережа шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Не допускається використовувати нульовий робочий провідник як нульовий захисний провідник. Нульовий захисний провідник прокладається від стійки групового розподільного щита, розподільного пункту до розеток електроживлення. Не допускається підключати на щиті до одного контактного затискача нульовий робочий та нульовий захисний провідники.

Площа перерізу нульового робочого та нульового захисного провідника в груповій трипровідній мережі має бути не менше площі перерізу фазового провідника. Усі провідники мають відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу провідників, температурному режиму та типам апаратури захисту, вимогам НПАОП 40.1-1.01-97.

У приміщенні, де одночасно експлуатуються понад п'ять комп'ютерів, на помітному, доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім

освітлення. Комп’ютери повинні підключатися до електромережі тільки за допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення.

У штепсельних з'єднаннях та електророзетках, крім контактів фазового та нульового робочого провідників, мають бути спеціальні контакти для підключення нульового захисного провідника. Їхня конструкція має бути такою, щоб приєднання нульового захисного провідника відбувалося раніше, ніж приєднання фазового та нульового робочого провідників. Порядок роз’єднання при відключенні має бути зворотним. Не допускається підключати комп’ютери до звичайної двопровідної електромережі, в тому числі – з використанням перехідних пристроїв. Електромережі штепсельних з'єднань та електророзеток для живлення комп’ютерної техніки повинні бути виконаними за магістральною схемою, по 3-6 з'єднань або електророзеток в одному колі. Штепсельні з'єднання та електророзетки для напруги 12 В та 42 В за своєю конструкцією мають відрізнятися від штепсельних з'єднань для напруги 127 В та 220 В. Штепсельні з'єднання та електророзетки, розраховані на напругу 12 В та 42 В, мають візуально (за кольором) відрізнятися від кольору штепсельних з'єднань, розрахованих на напругу 127 В та 220 В.

При підвищенні ефективності контролю доступу в приміщення, де для забезпечення безпеки мешканців, співробітників і збереження майна використовуються ДС, важливим, з точки зору охорони праці, є забезпечення достатньої величини природного та штучного освітлення, які визначені у НПАОП 0.00-7.15-18. Організація робочого місця фахівця із дослідження методів та програмно-апаратних засобів оптимізаційних процесів на основі ГА повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ДСТУ 8604:2015 «Дизайн і ергономіка. Робоче місце для виконання робіт у положенні сидячи. Загальні ергономічні вимоги». Відстань від екрана до ока фахівців, які працюють за комп’ютером визначається згідно з вимогами ДСанПіН 3.3.2.007-98.

Розміщення принтера або іншого пристрою введення-виведення інформації на робочому місці має забезпечувати добру видимість екрана

комп’ютера, зручність ручного керування пристроєм введення-виведення інформації в зоні досяжності моторного поля згідно з вимогами ДСанПіН 3.3.2.007-98.

Таким чином, у результаті аналізу вимог щодо охорони праці користувачів комп’ютерів, визначено особливості організації робочих місць, вимог з електробезпеки, природного та штучного освітлення для ефективної і безпечної роботи фахівців з проведення ґрунтовного аналізу мережного трафіку в режимі реального часу на основі ансамблевих методів машинного навчання та створення програмного застосунку для цього.

4.2. Вплив електромагнітного імпульсу ядерного вибуху на елементи виробництва та заходи захисту

У воєнний час при застосуванні ядерної зброї проти України на електронно-обчислювальне обладнання в першу чергу буде впливати електромагнітний імпульс (ЕМІ) ядерного вибуху у вигляді короткого імпульсу, який вражає головним чином електричну та електронну апаратуру. ЕМІ виникають в основному в результаті взаємодії гамма-випромінювання з атомами навколишнього середовища. На утворення ЕМІ йде невелика кількість ядерної енергії, але він здатен викликати високі імпульси струмів та напруг в кабелях повітряних і підземних ліній зв'язку, сигналізації, управління, електропередачі, в антенах радіостанцій. Вплив ЕМІ може привести до згортання чутливих електронних та електричних елементів, зв'язаних з великими антенами чи відкритими дротами, а також до порушень в обчислювальних пристроях. Вплив ЕМІ необхідно враховувати для всіх електричних та електронних систем [27].

Особливістю ЕМІ, як вражаючого фактору є його здатність розповсюджуватись на десятки і сотні кілометрів в оточуючому середовищі. Тому ЕМІ може вплинути своєю дією на об'єкти, там де вибухова хвиля, світлове випромінювання, проникаюча радіація втрачають своє значення, як вражаючі фактори. При наземних та низьких повітряних вибухах в лініях зв'язку та електрозабезпечення виникають напруги, які можуть викликати пробій ізоляції

провідників та кабелів відносно землі, пробій ізоляції елементів приладів підключених до повітряних і підземних ліній..

Найбільш піддані впливу ЕМІ системи зв'язку, сигналізації, управління. Використані в цих системах кабелі та апаратура мають обмежену електричну міцність не більше 10кВ імпульсної напруги, тоді як наведені імпульси напруги від ЕМІ можуть перевищувати ці значення. Найбільш піддана впливу ЕМІ апаратура виконана на напівпровідниках та інтегральних схемах, працюючих на малих струмах і напругах, і значить відчутних до впливу зовнішніх електричних і магнітних кіл, в тому числі і елементи програмного засобу для управління процесом міграції віртуальних машин в обчислювальній хмарі. ЕМІ пробиває ізоляцію, спалює елементи електричних схем радіоапаратури, викликає коротке замикання в радіопристроях, іонізацію діелектриків, змінює або повністю стирає магнітний запис. ЕМІ пошкоджує також резистори, викликає іскріння в їх міжконтактних з'єднаннях і деяких областях провідної поверхні. Найбільшу небезпеку ЕМІ представляє для апаратури, яка встановлена в особливо міцних спорудах, які витримують великі тиски ударної хвилі. В цих спорудах апаратура не виходить з ладу від механічних пошкоджень, але ЕМІ може вивести з ладу всю незахищену апаратуру системи зв'язку, сигналізації і керування. Найбільших значень досягають напруги, які наводяться між кабелем і землею.

Розглянемо можливі шляхи рішення задачі захисту від ЕМІ. Ідеальним захистом від ЕМІ виявилося б повне укриття приміщення, в якому розміщена радіоелектронна апаратура, металевим екраном [31]. Водночас зрозуміло, що практично забезпечити такий захист у ряді випадків неможливо, тому що для роботи апаратури часто потрібно забезпечити її електричний зв'язок із зовнішніми пристроями. Тому використовуються менш надійні засоби захисту, такі, як струмопровідні сітки, або плівкові покриття для вікон, щільникові металеві конструкції для повітрезабірників і вентиляційних отворів і контактні пружинні прокладки, розміщувані по периметру дверей і люків.

Більш складною технічною проблемою рахується захист від проникнення ЕМІ в апаратуру через різноманітні кабельні входи. Радикальним рішенням даної проблеми міг би стати перехід від електричних мереж зв'язку до практично

не схильних до впливу ЕМІ волоконно-оптичних. Проте заміна напівпровідникових приладів у всьому спектрі виконуваних ними функцій електронно-оптичними пристроями можлива тільки у віддаленому майбутньому. Тому в даний час в якості засобів захисту кабельних входів найбільші широко використовуються фільтри, у тому числі волоконні, а також іскрові розрядники, металлоокисні варистори і високошвидкісні зенеровські діоди [31].

Всі ці засоби мають як переваги, так і недоліки. Так, ємнісно-індуктивні фільтри достатньо ефективні для захисту від ЕМІ малої інтенсивності, волоконні фільтри захищають у відносно вузькому діапазоні надвисоких частот. Іскрові розрядники мають значну інерційність й в основному придатні для захисту від перевантажень, що виникають під впливом напруг і струмів, що наводяться в обшивці літака, кожусі апаратури й оплетенні кабеля.

Металоокисні варистори є напівпровідниковими приладами, що різко підвищують свою провідність при високій напрузі. Проте, при застосуванні цих приладів у якості засобів захисту від ЕМІ варто враховувати їх недостатньо високу швидкодію і погіршення характеристик при кількаразовому впливі навантажень. Ці недоліки відсутні у високошвидкісних зенеровських діодах, дія яких заснована на різкій лавиноподібній зміні опору від високого значення практично до нуля, при перевищенні прикладеної до них напруги граничного розміру. Крім того на відміну від варисторів характеристики зенеровських діодів після багатократних впливів високих напруг і переключень режимів не погіршуються.

Найбільш раціональним підходом до проектування засобів захисту від ЕМІ кабельних входів є створення таких роз'ємів у конструкції яких передбачені спеціальні заходи, що забезпечують формування елементів фільтрів і установку вмонтованих зенеровських діодів. Подібне рішення сприяє одержанню дуже малих значень ємності й індуктивності, що необхідно для забезпечення захисту від імпульсів, що мають незначну тривалість і, отже, потужну високочастотну складову. Використання роз'ємів подібної конструкції дозволить вирішити проблему обмеження малогабаритних характеристик

пристрою захисту. Складність рішення задачі захисту від ЕМІ і висока вартість розроблених для цього засобів і методів змушують піти по шляху їхнього вибіркового застосування в особливо важливих системах зброї і військової техніки. Такий же шлях обраний і для захисту систем, що мають велику протяжність, керування і зв'язку. Проте основним методом вирішення проблеми спеціалісти вважають створення так званих розподілених мереж зв'язку [27].

4.3. Висновки до четвертого розділу

В цьому розділі проаналізовано важливі питання охорони праці та безпеки в надзвичайних ситуаціях, висвітлено питання впливу електромагнітного імпульсу ядерного вибуху на елементи виробництва та заходи захисту.

ВИСНОВКИ

У рамках цієї роботи було створено програму для аналізу мережного трафіку в режимі реального часу на базі ансамблевих методів машинного навчання.

У ході роботи було вирішено такі завдання:

– проведено огляд наукових джерел за тематикою дослідження;

– втілено обрані ансамблеві методи машинного навчання (AdaBoost, Random Forest);

– розраховано оцінку якості втілених алгоритмів (на базі метрик якості навчання);

– реалізовано сніффер;

– виконано оцінку якості розробленого застосунку.

Реалізовано компоненти застосунку, а також проведено передобробку вихідного НД. Виконано функціональне та A/B тестування, доведено потребу балансування даних, так як цього модель значно гірше класифікувала окремі класи. Виконано тестування роботи застосунку в онлайн режимі, а також при роботі із локальними файлами. Розроблений застосунок успішно класифікував як звичайний, так і шкідливий трафік.

У майбутньому планується вдосконалення розробленого застосунку, покращення якості класифікації, розширення функціональності.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Machine Learning, ML. IT-Enterprise – your one-stop platform for digital transformation | www.it.ua. URL: https://www.it.ua/knowledge-base/technology-innovation/machine-learning (date of access: 15.12.2024).

2. Rocca J. Ensemble methods: bagging, boosting and stacking. Medium. URL: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205 (date of access: 15.12.2024).

3. Розподілені мережеві атаки | ERC Ukraine. ERC Ukraine. URL: https://erc.ua/erc-reviews/20759/rozpodileni-merezhevi-ataki (дата звернення: 15.12.2024).

4. Tymoshchuk, V., Vorona, M., Dolinskyi, A., Shymanska, V., & Tymoshchuk, D. (2024). SECURITY ONION PLATFORM AS A TOOL FOR DETECTING AND ANALYSING CYBER THREATS. Collection of scientific papers «ΛΌГОΣ», (December 13, 2024; Zurich, Switzerland), 232-237.

5. Tymoshchuk, V., Mykhailovskyi, O., Dolinskyi, A., Orlovska, A., & Tymoshchuk, D. (2024). OPTIMISING IPS RULES FOR EFFECTIVE DETECTION OF MULTI-VECTOR DDOS ATTACKS. Матеріали конференцій МЦНД, (22.11. 2024; Біла Церква, Україна), 295-300.

6. Applications | Research | Canadian Institute for Cybersecurity | UNB. University of New Brunswick | UNB. URL: https://www.unb.ca/cic/research/applications.html#CICFlowMeter (date of access: 15.12.2024).

7. Tymoshchuk, V., Vantsa, V., Karnaukhov, A., Orlovska, A., & Tymoshchuk, D. (2024). COMPARATIVE ANALYSIS OF INTRUSION DETECTION APPROACHES BASED ON SIGNATURES AND ANOMALIES. Матеріали конференцій МЦНД, (29.11. 2024; Житомир, Україна), 328-332.

8. Ukrinform. Кількість зареєстрованих кіберінцидентів минулого року зросла на 62,5% - Держспецзв'язку. Укрінформ - актуальні новини України та світу. URL: https://www.ukrinform.ua/rubric-technology/3812394-kilkist-

zareestrovanih-kiberincidentiv-minulogo-roku-zrosla-na-625-derzspeczvazku.html (дата звернення: 15.12.2024).

9. Tymoshchuk, V., Pakhoda, V., Dolinskyi, A., Karnaukhov, A., & Tymoshchuk, D. (2024). MODELLING CYBER THREATS AND EVALUATING THE PERFORMANCE OF INTRUSION DETECTION SYSTEMS. Grail of Science, (46), 636–641. https://doi.org/10.36074/grail-of-science.29.11.2024.081

10. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.

11. The Use of Ensemble Models for Multiple Class and Binary Class Classification for Improving Intrusion Detection Systems / C. Iwendi et al. Sensors. 2020. Vol. 20, no. 9. P. 2559. URL: https://doi.org/10.3390/s20092559 (date of access: 15.12.2024).

12. Shahraki A., Abbasi M., Haugen Ø. Boosting algorithms for network intrusion detection: A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost. Engineering Applications of Artificial Intelligence. 2020. Vol. 94. P. 103770. URL: https://doi.org/10.1016/j.engappai.2020.103770 (date of access: 15.12.2024).

13. Юречко О.С. Ансамблеві методи машинного навчання // Інформаційні моделі, системи та технології: Праці XII наук.-техн. конф. (Тернопіль, ТНТУ ім. І. Пулюя, 18-19 грудня 2024 р.) С. 126.

14. Stanko, A., Wieczorek, W., Mykytyshyn, A., Holotenko, O., & Lechachenko, T. (2024). Realtime air quality management: Integrating IoT and Fog computing for effective urban monitoring. CITI, 2024, 2nd.

15. Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T., (2024). Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 3896, pp. 1-11.

16. Sen S., Gupta K. D., Ahsan M. M. Leveraging Machine Learning Approach to Setup Software-Defined Network(SDN) Controller Rules During DDoS Attack. SpringerLink. URL: https://link.springer.com/chapter/10.1007/978-981-13-7564-4_5 (date of access: 15.12.2024)

17. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56.

18. Глоба Л. С., Астраханцев А.А., Цуканов С.О. Класифікація мережного трафіку методами машинного навчання. URL: https://pt.nure.ua/wp-content/uploads/2023/12/223_globa_traffic_.pdf (дата звернення: 29.11.2024).

19. What is Boosting? - Boosting in Machine Learning Explained - AWS. Amazon Web Services, Inc. URL: https://aws.amazon.com/what-is/boosting/ (date of access: 15.12.2024).

20. RandomForestClassifier. scikit-learn. URL: https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html (date of access: 15.12.2024).

21. AdaBoostClassifier. scikit-learn. URL: https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.AdaBoostClassifier.html (date of access: 15.12.2024).

22. Ревнюк, О. А., Загородна, Н. В., Козак, Р. О., Карпінський, М. П., & Флуд, Л. О. (2024). The improvement of web-application SDL process to prevent Insecure Design vulnerabilities. Прикладні аспекти інформаційних технологій, 7(2), 162-174.

23. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220.

24. Mishko, O., Matiuk, D., & Derkach, M. (2024). Security of remote iot system management by integrating firewall configuration into tunneled traffic. Вісник Тернопільського національного технічного університету, 115(3), 122-129.

25. IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB. University of New Brunswick | UNB. URL: https://www.unb.ca/cic/datasets/ids-2017.html (date of access: 15.12.2024).

26. CICIDS-2017 Dataset Feature Analysis With Information Gain for Anomaly Detection / Kurniabudi et al. IEEE Access. 2020. Vol. 8. P. 132911–132921. URL: https://doi.org/10.1109/access.2020.3009843 (date of access: 15.12.2024).

27. Hulk DDoS Tool : Complete Installation & Usage with Examples. URL: https://allabouttesting.org/hulk-ddos-toolcomplete-installation-usage-with-examples/ (te of access: 15.12.2024).

28. Zagorodna, N., Skorenkyy, Y., Kunanets, N., Baran, I., & Stadnyk, M. (2022). Augmented Reality Enhanced Learning Tools Development for Cybersecurity Major. In ITTAP (pp. 25-32).

29. Ubuntu Manpage: goldeneye - HTTP DoS test tool. Ubuntu Manpage:Welcome. URL: http://manpages.ubuntu.com/manpages/bionic/man1/goldeneye.1.html (date of access: 15.12.2024).

30. patator | Kali Linux Tools. Kali Linux. URL: https://www.kali.org/tools/patator (date of access: 15.12.2024).

31. Tymoshchuk, D., & Yatskiv, V. (2024). Slowloris ddos detection and prevention in real-time. Collection of scientific papers «ΛΌГОΣ», (August 16, 2024; Oxford, UK), 171-176.

32. Bomba, A., Lechachenko, T., & Nazaruk, M. (2021). Modeling the Dynamics of “Knowledge Potentials” of Agents Including the Stakeholder Requests. In Advances in Computer Science for Engineering and Education IV (pp. 75-88). Springer International Publishing.

33. What is DDoS Attack | Types, Examples, Detection & Prevention. SSLInsights. URL: https://sslinsights.com/what-is-ddos-attack/ (date of access: 15.12.2024).

34. What is SQL Injection? Tutorial & Examples | Web Security Academy. Web Application Security, Testing, & Scanning - PortSwigger. URL: https://portswigger.net/web-security/sql-injection (date of access: 15.12.2024).

35. Derkach, M., Skarga-Bandurova, I., Matiuk, D., & Zagorodna, N. (2022, December). Autonomous quadrotor flight stabilisation based on a complementary filter and a PID controller. In 2022 12th International Conference on Dependable Systems, Services and Technologies (DESSERT) (pp. 1-7). IEEE.

36. scikit-learn: machine learning in Python – scikit-learn 1.6.0 documentation. scikit-learn: machine learning in Python – scikit-learn 0.16.1 documentation. URL: https://scikit-learn.org/stable/ (date of access: 15.12.2024).

37. pandas - Python Data Analysis Library. pandas - Python Data Analysis Library. URL: https://pandas.pydata.org (date of access: 15.12.2024).

38. Google Colab. Google Colab. URL: https://colab.research.google.com/?hl=ua (date of access: 15.12.2024).

39. JetBrains. PyCharm: the Python IDE for data science and web development. JetBrains. URL: https://www.jetbrains.com/pycharm (date of access: 15.12.2024).

40. Derkach, M., Lysak, V., Skarga-Bandurova, I., & Kotsiuba, I. (2019, September). Parking Guide Service for Large Urban Areas. In 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (Vol. 1, pp. 567-571). IEEE.

41. SelectKBest. scikit-learn. URL: https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.SelectKBest.html (date of access: 15.12.2024).

42. MinMaxScaler. scikit-learn. URL: https://scikit-learn.org/1.5/modules/generated/sklearn.preprocessing.MinMaxScaler.html (date of access: 15.12.2024).

43. Yesin, V., Karpinski, M., Yesina, M., Vilihura, V., Kozak, R., & Shevchuk, R. (2023). Technique for Searching Data in a Cryptographically Protected SQL Database. Applied Sciences, 13(20), 11525.

44. Довідник по Machine Learning – Machine Learning Lifecycle | База знань IT. База знань IT технологій. URL: https://itwiki.dev/data-science/ml-reference/ml-glossary/machine-learning-lifecycle (дата звернення: 15.12.2024).

45. HTTP DoS Dataset in PCAP format for Wireshark. figshare. URL: https://ordo.open.ac.uk/articles/dataset/HTTP_DoS_Dataset_in_PCAP_format_for_Wireshark/17206289 (date of access: 15.12.2024).

46. Muzh, V., & Lechachenko, T. (2024). Computer technologies as an object and source of forensic knowledge: challenges and prospects of development. Вісник Тернопільського національного технічного університету, 115(3), 17-22

47. Кучма М.М. Цивільна оборона (цивільний захист). Навчальний посібник. Львів : Магнолія 2006 . 2024. 296 с.

ЗАСТОСУНОК А

Тези конференції