

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»

Директор інституту(декан факультету)

Андрій ФОРСЮК

(ім'я та прізвище)

«13» грудня 2024р.

«До захисту допущено»

Завідувач кафедри

Сергій ГРИБКОВ

(ім'я та прізвище)

«13» грудня 2024р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

зі спеціальності 122 «Комп'ютерні науки»

(код та назва спеціальності)

освітньо-професійної програми Управління інформацією та аналітика даних
на тему: Створення інформаційної системи аналізу та виявлення
потенційних кібернетичних злочинів

Виконав: здобувач 2 курсу, групи КН-2-4М

Шульгін Арсеній Владиславович

(прізвище, ім'я, по батькові повністю)

Керівник Грама Михайло Петрович

(прізвище, ім'я та по батькові повністю)

Консультанти

(ім'я та прізвище)

(підпис)

(ім'я та прізвище)

(підпис)

(ім'я та прізвище)

(підпис)

Рецензент

Вісша Косідрішко Вісша

(ім'я та прізвище)

(підпис)

Я як здобувач Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав і не одержував недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач

(підпис)

Київ - 2024р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем

Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки

Освітній ступінь магістр

Спеціальність 122 «Комп'ютерні науки»

(код і назва)

Освітньо-професійна програма Управління інформацією і аналітика даних

(назва)

ЗАТВЕРДЖУЮ

Завідувач

кафедри Інформаційних технологій,
штучного інтелекту і кібербезпеки

Грибков С.В.

“ 07 ” жовтня 2024 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Шульгіна Арсенія Владиславовича

(прізвище, ім'я, по батькові)

1. Тема роботи Створення інформаційної системи аналізу та виявлення потенційних кібернетичних злочинів

керівник роботи Грама Михайло Петрович, доктор філософії,

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від 7 жовтня 2024 року №884-кв

2. Строк подання здобувачем роботи 6 грудня 2024

3. Вихідні дані до роботи Записка, автореферат, презентація та рецензія

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Розділ 1. Досліджені задачі

Розділ 2. Опис Математичної моделі

Розділ 3. Аналіз програмної частини з прогнозування

Розділ 4. Візуалізація та аналіз результатів

5. Перелік графічного матеріалу

Презентація, 28 рисунків, 1 таблиця

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Грама М. П. старший викладач	07.10.2024 <i>Кев</i>	20.10.2024 <i>Кев</i>
2	Грама М. П. старший викладач	20.10.2024 <i>Кев</i>	26.10.2024 <i>Кев</i>
3	Грама М. П. старший викладач	26.10.2024 <i>Кев</i>	05.11.2024 <i>Кев</i>
4	Грама М. П. старший викладач	05.11.2024 <i>Кев</i>	29.11.2024 <i>Кев</i>

7. Дата видачі завдання 7 жовтня 2024 року

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Постанова проблематики	10.10.2024-15.10.2024	Виконано
2.	Аналіз літератури	16.10.2024-20.10.2024	Виконано
3.	Пошук набору даних	22.10.2024-25.10.2024	Виконано
4.	Аналіз набору даних	27.10.2024-02.11.2024	Виконано
5.	Реалізація програмної частини	5.11.2024-21.11.2024	Виконано
6.	Виконання візуалізації даних	22.11.2024-25.11.2024	Виконано
7.	Оформлення розділів записки	27.11.2024-30.11.2024	Виконано
8.	Оформлення автореферату	01.12.2024-02.12.2024	Виконано
9.	Оформлення презентації	02.12.2024-06.12.2024	Виконано

Здобувач

Керівник роботи

(підпис)

Кев

(підпис)

Шульгін А.В.
(прізвище та ініціали)Грама М.П.
(прізвище та ініціали)

АНОТАЦІЯ

Шульгін Арсеній Владиславович – Створення інформаційної системи аналізу та виявлення потенційних кібернетичних злочинів.

Кваліфікаційна робота: 81 сторінок, 28 рисунків, 1 таблиця, 2 додатки, 32 джерел.

Робота присвячена використанню сучасних підходів для аналізу та виявлення кіберзагроз у мережевих системах.

Проведений аналіз, як можна моніторити мережеву активність, шукати аномалії та визначати потенційні атаки.

Мета дослідження – розробити інформаційну систему, яка допоможе виявляти кіберзлочини та забезпечити аналітику для швидкого реагування.

Об’єкт дослідження – мережеві дані, які використовуються для моніторингу активності та пошуку аномальних дій. Предмет дослідження – методи аналізу й візуалізації мережевих даних, а також можливість автоматизованого виявлення потенційних кіберзлочинів.

Результат роботи – створено прототип інформаційної системи, що аналізує мережеві потоки за допомогою Python. Було реалізовано ETL-процес, створено базу даних та аналітичний звіт у Power BI. Результати показують, що виявлення загроз стає значно ефективнішим при інтеграції сучасних методів аналізу.

Ключові слова: КІБЕРЗЛОЧИНИ, АНАЛІЗ ДАНИХ, МЕРЕЖЕВА БЕЗПЕКА, ВИЗНАЧЕННЯ АНОМАЛІЙ, ВІЗУАЛІЗАЦІЯ ДАНИХ, POWER BI.

SUMMARY

Shulhin Arsenii Vladislavovich – Development of an Information System for Analyzing and Detecting Potential Cybercrimes.

Qualification Work: 81 pages, 28 figures, 1 table, 2 appendices, 32 references.

This work focuses on the application of modern approaches for analyzing and detecting cyber threats in network systems.

The study includes an analysis of methods for monitoring network activity, identifying anomalies, and detecting potential attacks.

Research Aim: To develop an information system capable of detecting cybercrimes and providing analytics for rapid response.

Objective of the study – to develop an information system that assists in detecting cybercrimes and provides analytics for rapid response.

Object of the study – network data used for monitoring activities and detecting suspicious actions.

Subject of the study – methods for analyzing and visualizing network data and the potential for automated detection of cybercrime.

Result of the work – a prototype of an information system was developed to analyze network flows using Python. The project implemented an ETL process, created a database, and prepared an analytical report in Power BI. The results demonstrate that integrating modern analysis methods significantly improves threat detection efficiency.

Keywords: CYBERCRIME, DATA ANALYSIS, NETWORK SECURITY, ANOMALY DETECTION, DATA VISUALIZATION, POWER BI.

ЗМІСТ

ВСТУП	8
РОЗДІЛ 1. ДОСЛІДЖЕННЯ ЗАДАЧІ.....	10
1.1. Актуальність дослідження	10
1.2. Можливі шляхи реалізації.....	11
1.3. Огляд методів аналізу та збору даних	11
1.4. Використання методів регресії та нейронних мереж у виявленні кіберзлочинів	15
1.5. Візуалізація даних у кібербезпеці	16
1.6. Висновок до першого розділу.....	17
РОЗДІЛ 2. ОПИС МАТЕМАТИЧНОЇ МОДЕЛІ	19
2.1. Загальні відомості	19
2.2. Дерева рішень.....	20
2.3. Математична модель дерева рішень	23
2.4. Оцінка якості моделі.....	24
2.5. Оцінка якості моделі: Матриця помилок та ROC-аналіз.....	26
2.6. Дерева рішень як основа моделі.....	28
2.7. Регуляризація.....	29
2.8. Критерії точності.....	29
2.9. Висновок до другого розділу	30
РОЗДІЛ 3. АНАЛІЗ ПРОГРАМНОЇ ЧАСТИНИ З ПРОГНОЗУВАННЯ.....	31
3.1. Вибір і зберігання даних	31
3.2. Попередня обробка даних	33
3.3. ETL-процес	34
3.4. Очищення та обробка даних	35
3.5. Зберігання даних у базі.....	36
3.6. Навчання моделі та результати.....	37
3.7. Підготовка даних до моделювання	38
3.8. Візуалізація результатів моделі	39
3.9. Висновок до третього розділу.....	43

РОЗДІЛ 4. ВІЗУАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ.....	45
4.1. Перевірка даних перед візуалізацією	45
4.2. Створення візуалізацій.....	45
4.3. Інтерактивні панелі в Power BI	55
4.4. Висновок до четвертого розділу	55
ВИСНОВКИ.....	58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	61
ДОДАТКИ.....	65

ВСТУП

Сучасний світ переживає епоху стрімкого розвитку інформаційних технологій, які стали невід'ємною частиною повсякденного життя. Ці досягнення відкрили нові можливості для зберігання, обробки та передачі даних, водночас створивши численні ризики, пов'язані з кіберзлочинністю. Зловмисники використовують прогалини в системах безпеки для здійснення атак, які несуть значні загрози як для окремих осіб, так і для організацій.

У зв'язку зі зростанням кількості кіберзагроз виникає потреба у впровадженні сучасних підходів до аналізу та виявлення потенційних кіберзлочинів. Актуальність теми обумовлена необхідністю розробки ефективних інформаційних систем, які здатні вчасно реагувати на загрози, забезпечуючи належний рівень захисту інформаційних ресурсів.

Кваліфікаційна робота зосереджується на створенні інформаційної системи аналізу даних, яка поєднує використання технологій Python та Power BI для обробки та візуалізації інформації. Особлива увага приділяється методам попередньої обробки даних, побудові бази даних та розробці аналітичного звіту, що базується на аналізі мережевого трафіку.

Мета дослідження полягає у підвищенні ефективності процесів виявлення кіберзлочинів шляхом інтеграції сучасних технологій для аналізу та представлення інформації. Основні результати роботи мають прикладне значення та можуть бути використані в різних сферах, пов'язаних із забезпеченням кібербезпеки.

Предмет дослідження даної кваліфікаційної роботи охоплює процеси збору, обробки та візуалізації даних для виявлення потенційних кіберзлочинів, а саме аналіз мережевого трафіку та виявлення аномалій за допомогою методів машинного навчання. Основною метою є розробка інформаційної системи для аналізу кіберзагроз, що включає побудову ефективних моделей для прогнозування атак на основі реальних даних мережевого трафіку.

Наукова новизна дослідження полягає в розробці інноваційного підходу

до аналізу та виявлення кіберзагроз на основі використання інтегрованих методів машинного навчання та візуалізації даних. Розроблена система дозволяє здійснювати ефективний моніторинг мережевого трафіку, виявляти потенційні кіберзагрози та надавати аналітичні звіти для прийняття оперативних рішень. Окрім того, інноваційним є використання сучасних інструментів, таких як Python для попередньої обробки даних та Power BI для інтерактивної візуалізації результатів.

Апробація результатів дослідження може бути здійснена через інтеграцію розробленої системи в реальні середовища кібербезпеки, таких як захист корпоративних мереж, інформаційні системи організацій або урядові установи, що займаються безпекою даних. Крім того, результати можуть бути використані для подальших досліджень в області виявлення аномалій у великих даних, що зберігаються та передаються через мережі, в тому числі для створення нових підходів до автоматизації виявлення кіберзагроз.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ЗАДАЧІ

В даному розділі описані актуальність дослідження, можливі шляхи реалізації, огляд методів аналізу та збору даних.

1.1. Актуальність дослідження

З розвитком цифрових технологій, а також збільшенням обсягів мережевого трафіку зростає кількість кіберзлочинів, яка спричиняє серйозні проблеми, як для індивідуальних користувачів, так само і для великих організацій. Кіберзлочинність охоплює дуже широкий спектр злочинів, такі як несанкціонований доступ до даних і до складних атак на інфраструктуру. Ця проблема є загрозою національній безпеці і фінансовим установам, а також великим корпораціям, що використовують для своєї діяльності критично важливі дані [12].

Загрози кібербезпеці останніми роками значно змінилися. Деякі злочинці почали використовувати нові методи для обходу традиційних систем безпеки, щоб це побороти традиційні підходи до виявлення кіберзлочинів вже не надають бажаних результатів. Виходячи з цього, актуальність даної роботи полягає в розробці ефективної інформаційної системи для моніторингу та виявлення потенційних кіберзлочинів. Завдяки ній, можна забезпечити можливість швидкого реагування на загрози, що, в свою чергу, дозволить знизити ризики для цінних інформаційних активів і ресурсів.

До цього ж можна зазначити, що з кожним роком методи та інструменти для збору та обробки даних розвиваються, тому створення таких інформаційних систем стає дедалі більш доступним, а ще ефективним. Враховуючи те, що інформаційна безпека є дуже невід'ємною частиною сучасного цифрового світу – ця проблема в першу чергу набуває ще більшої актуальності.

1.2. Можливі шляхи реалізації

У задачах машинного навчання [2], а також у задачах класифікації та прогнозування, реалізація моделі має залежити від кількох факторів, а саме таких, як вимоги до точності, доступність даних, швидкості роботи і також складності алгоритму. У світі існує декілька підходів до реалізації моделей для виявлення кіберзагроз і аномалій в мережі, які можливо застосувати в нашій кваліфікаційній роботі.

Щоб створити інформаційну систему для виявлення кіберзлочинів можна обрати кілька напрямків. По-перше, це використання стандартних алгоритмів машинного навчання та побудови моделей на основі історичних даних. Але треба зазначити, що цей підхід вимагає наявності величезних обсягів навчальних даних, а це не завжди є доступним для маленьких організацій або окремих користувачів [19].

Другим підходом можна зазначити використання технологій обробки великих даних (Big Data), що дозволяють обробляти, а головне зберігати великі обсяги інформації, які отримані з різних джерел, наприклад мережеві монітори, або системи виявлення вторгнень, або сервери та інші джерела. Саме це дозволяє будувати складні моделі для аналізу та прогнозування загроз, при цьому враховуючи різноманітні фактори, наприклад, такі, як аномалії в поведінці користувачів, несанкціоновані спроби доступу до даних, а також різні технічні атаки [10].

Інтеграція технологій штучного інтелекту та аналізу даних дає можливість побудувати більш гнучкі та адаптивні системи, які здатні самостійно навчатись на основі нових даних і виявляти нові типи атак без втручання людини [11].

1.3. Огляд методів аналізу та збору даних

Одним із основних етапів створення інформаційної системи є аналіз даних. Для ефективного виявлення кіберзлочинів необхідно мати доступ до

великої кількості інформації, що стосується мережевого трафіку, дій користувачів, а також характеристик системи.

Збір даних можна здійснювати за допомогою різних технологій, серед яких основними є системи моніторингу мережі, лог-файли, дані про спроби вторгнень та вразливості. Однією з важливих складових є дані про трафік, який проходить через мережу. Це можуть бути пакети даних, які містять важливу інформацію про можливі аномалії та спроби атак.

Для аналізу даних використовуються різноманітні методи:

- **Аналіз аномалій** – це метод, який дозволяє виявляти відхилення від звичайної поведінки системи, наприклад, аномальні підключення до серверів чи високі піки мережевого трафіку [3].
- **Класифікація за допомогою машинного навчання** – цей метод дозволяє автоматично класифікувати події в системі на “нормальні” та “аномальні” на основі попереднього навчання моделей [4].
- **Аналіз мережевих протоколів** – використовується для виявлення аномалій на рівні мережевого протоколу, таких як TCP, UDP, HTTP.
- **Графовий аналіз** – метод побудови графів взаємодії між пристроями в мережі, що дозволяє виявляти несанкціоновані взаємодії або спроби проникнення.

TCP (Transmission Control Protocol) – це ключовий транспортний протокол, який забезпечує надійну передачу даних у мережах. Його головна перевага – гарантія доставки інформації, причому в правильному порядку і без втрат. Завдяки цьому протокол широко використовується у всіх сучасних мережах, включаючи Інтернет.

У контексті роботи TCP відіграє особливу роль, оскільки є основним "каналом" як для нормального трафіку, так і для атак. Саме через нього проходить більшість мережевого трафіку, що робить його пріоритетним для аналізу. Протокол активно використовують зловмисники для проведення атак, таких як DoS, сканування портів або інші види вторгнень.

Важливо, що у 4 розділі детально розглянуто аналіз розподіл атак за протоколами, і TSP став найпомітнішим із них. Це підтвердило, що для ефективного виявлення аномалій та попередження атак необхідно приділяти особливу увагу саме цьому протоколу. У теоретичній частині ми заклали базу, пояснивши, чому TSP є таким важливим, і як його специфіка впливає на роботу систем виявлення аномалій.

Методи аналізу даних базуються на алгоритмах машинного навчання, таких як класифікація, кластеризація, прогнозування, що дозволяє розпізнавати патерни кіберзлочинів та виявляти нові загрози [5].

Збір даних та їх подальша обробка є основою для створення аналітичних звітів та побудови моделей для виявлення потенційних кіберзлочинів. Сучасні технології дають можливість обробляти величезні обсяги інформації за допомогою кластерних обчислень та хмарних платформ, що робить цей процес більш швидким та ефективним [6].

У процесі дослідження важливу роль відіграє збір та попередня обробка даних, адже саме на цьому етапі формується основа для подальшого аналізу та побудови моделей. Для досягнення високої точності у виявленні кіберзагроз потрібно забезпечити правильний підхід до збору та аналізу інформації.

Основними етапами цього процесу є:

1. **Збір даних:** інформація збирається з різних джерел, таких як файли журналів, системи моніторингу, мережеві запити тощо.
2. **Попередня обробка даних:** очищення даних, видалення помилок, пропусків, нормалізація і підготовка до подальшого аналізу.
3. **Аналіз даних:** це етап, на якому використовуються різні методи статистичного аналізу, машинного навчання для виявлення закономірностей, трендів і аномалій у даних.
4. **Побудова моделей:** в залежності від типу задачі, застосовуються методи регресії, класифікації чи кластеризації для моделювання даних.

5. **Візуалізація результатів:** на цьому етапі створюються різноманітні візуалізації (діаграми, графіки) для полегшення аналізу результатів і спрощення прийняття рішень [7].

Щоб краще зрозуміти взаємозв'язок між етапами збору, обробки та аналізу даних, наведена нижче **блок-схема** (рис. 1.1) наочно демонструє логічну послідовність цих кроків:

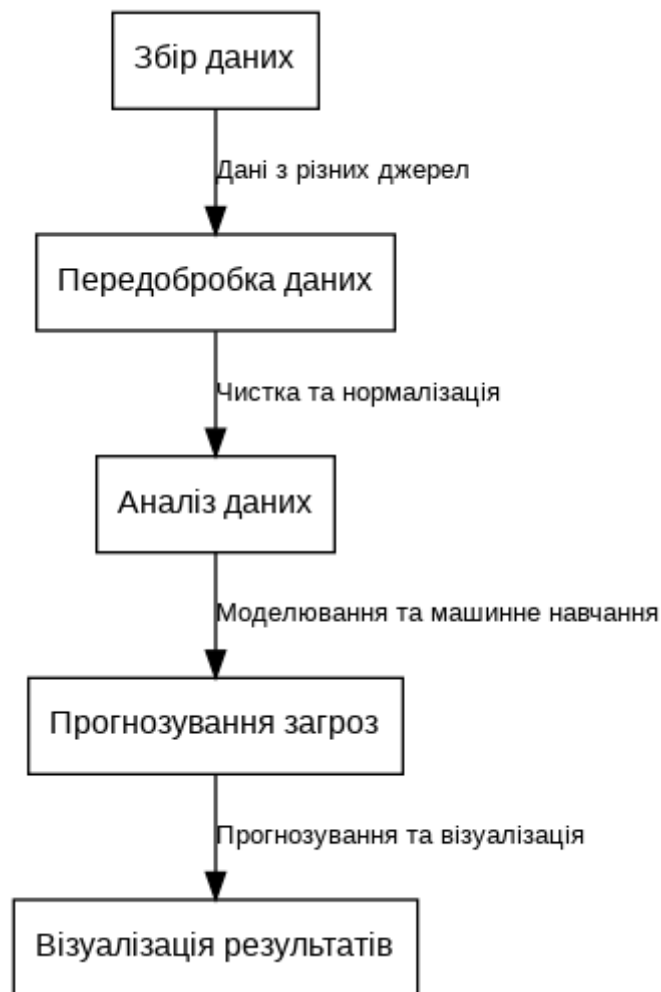


Рисунок 1.1 – Процес збору та аналізу даних

Блок-схема "Процес збору та аналізу даних":

1. **Збір даних:** на цьому етапі збирається вся необхідна інформація з різних джерел.
2. **Попередня обробка даних:** очищення даних, видалення помилок, аномалій і підготовка до аналізу [6].

3. Аналіз та виявлення закономірностей: використання методів статистики та машинного навчання для виявлення патернів у даних.

4. Моделювання даних: побудова моделей для передбачення або класифікації виявлених аномалій.

5. Візуалізація результатів: створення графіків, діаграм і таблиць для полегшення інтерпретації результатів.

Загальна структура процесу виявлення загроз через використання цих етапів дозволяє більш точно і ефективно здійснювати прогнозування, що є ключовим для подальшої побудови моделей для виявлення аномалій у реальних даних.

1.4. Використання методів регресії та нейронних мереж у виявленні кіберзлочинів

Якщо вивчати сучасні методи аналізу даних для виявлення кіберзлочинів, то важливо згадати про використання **регресійних моделей** та **нейронних мереж** [8]. Обидва ці методи активно використовуються для прогнозування та виявлення аномалій у поведінці систем.

• **Регресія:** Це один з основних статистичних методів для прогнозування залежності між змінними. Вона може бути корисною для виявлення трендів в історичних даних, таких як мережевий трафік або кількість спроб вторгнення. Регресійні моделі дозволяють побудувати математичні залежності, що допомагають прогнозувати ймовірність певних подій. В контексті кібербезпеки регресія може бути використана для аналізу зв'язку між різними параметрами мережі (наприклад, часу відгуку серверів, кількості запитів на сервер) і потенційними загрозами.

• **Нейронні мережі:** Це більш складні моделі, здатні виявляти глибокі, нелінійні зв'язки між даними. Для аналізу кіберзлочинів нейронні мережі дають можливість навчати модель на великих масивах даних, де традиційні методи не завжди здатні виявити тонкі патерни. Нейронні мережі можуть навчатися розпізнавати складні аномалії в поведінці користувачів, що

допомагає визначити потенційні загрози до того, як вони призведуть до серйозних наслідків.

Одним із найпопулярніших підходів є використання **глибоких нейронних мереж** для аналізу великих наборів даних [14]. Це дозволяє виявляти аномалії, які можуть бути невидимими для традиційних моделей, адже нейронні мережі здатні знаходити складні зв'язки в даних, навіть у разі присутності шуму.

1.5. Візуалізація даних у кібербезпеці

Візуалізація даних є одним з найбільш потужних інструментів для аналізу і розуміння складних наборів даних у сфері кібербезпеки. Вона дозволяє не лише покращити ефективність прийняття рішень, але й допомагає виявити невидимі на перший погляд патерни і аномалії, які можуть свідчити про кіберзагрози. Використання різноманітних візуальних інструментів дозволяє оперативно реагувати на зміни в мережі та розробляти стратегії захисту, ґрунтуючись на реальних даних [9].

Для ефективного аналізу та виявлення кіберзлочинів, важливо не тільки збирати та обробляти дані, але й ефективно їх візуалізувати. Візуалізація допомагає аналітикам швидко і точно інтерпретувати результати, а також виявляти потенційні загрози в реальному часі [15]. Для цього використовуються різні інструменти, такі як **PowerBI**, **Tableau** та інші.

Power BI є потужним інструментом для створення інтерактивних візуалізацій, який широко застосовується в кібербезпеці для аналізу даних. Візуалізація даних у Power BI дає змогу:

- Легко створювати інтерактивні графіки та діаграми для аналізу мережевого трафіку, атак, протоколів та інших параметрів.
- Використовувати фільтри та панелі для глибшого аналізу, наприклад, аналізу мережевих інцидентів за різними типами атак.

- Проводити порівняння між різними періодами часу, виявляючи підвищену активність атак у певні моменти.

Наприклад, можна створити графіки, що показують аномалії в мережевому трафіку, або зображення теплових карт, які наочно демонструють підозрілі активності в різних частинах мережі. Такі візуалізації дозволяють зменшити час на виявлення потенційних проблем і забезпечити більш швидке реагування на загрози.

У галузі кібербезпеки основною метою візуалізації є зменшення складності великих обсягів даних, що генеруються під час атак, а також виявлення аномальних патернів, які можуть свідчити про підозрілі чи зловмисні дії. Завдяки візуалізації складні дані стають доступними для широкого кола користувачів, включаючи аналітиків без глибоких технічних знань.

- **Детектування аномалій:** одним із основних застосувань візуалізації є виявлення аномалій у мережі. Використовуючи графіки, гістограми та кореляційні діаграми, можна визначити, які параметри мережевого трафіку є вищими за звичайні значення, що може вказувати на зловмисні спроби.
- **Зв'язок між різними типами атак і елементами мережі:** візуалізація дозволяє аналізувати взаємозв'язок між атакувальними методами та використаними протоколами, IP-адресами, портами тощо. Це дозволяє виявляти найбільш вразливі точки в мережі.

Візуалізація допомагає не тільки аналізувати дані, а й ефективно презентувати результати. Завдяки інтуїтивно зрозумілим графікам і діаграмам можна отримати чітке уявлення про ситуацію в кібербезпеці і прийняти відповідні заходи.

1.6. Висновок до першого розділу

У результаті дослідження актуальності задачі можна зробити кілька важливих висновків:

1. Кіберзлочинність залишається однією з головних загроз для безпеки інформаційних систем.

2. Розробка інформаційних систем для виявлення кіберзлочинів є надзвичайно важливим елементом сучасної кібербезпеки.

3. Сучасні методи аналізу даних, такі як регресія, нейронні мережі та візуалізація, можуть значно підвищити ефективність виявлення аномалій і прогнозування загроз.

4. Використання технологій штучного інтелекту та машинного навчання дає можливість побудови адаптивних систем, здатних виявляти нові типи атак [13].

РОЗДІЛ 2. ОПИС МАТЕМАТИЧНОЇ МОДЕЛІ

В даному розділі буде розглянута математична модель, яка використовується для аналізу та виявлення потенційних кібернетичних злочинів на основі виявлення аномалій у даних.

2.1. Загальні відомості

Оскільки задача полягає у виявленні аномалій, модель повинна бути здатною чітко розмежовувати нормальні та аномальні дані, що можуть свідчити про загрози.

Вибір математичної моделі для даної задачі здійснюється з урахуванням специфіки задачі та характеру даних. Оскільки наша мета — виявлення аномалій в мережевому трафіку, то найбільш підходящими є методи, що дозволяють автоматично ідентифікувати невідповідності і підозрілі патерни в великих масивах даних. Одним із найбільш ефективних підходів для цієї задачі є використання методів класифікації та регресії. Вибір між ними залежить від особливостей завдання, проте для виявлення аномалій в кібербезпеці часто використовують методи класифікації, оскільки задача зводиться до класифікації мережевого трафіку на два класи: нормальний та аномальний [18].

У рамках нашої роботи було обрано метод дерев рішень, оскільки вони забезпечують хорошу інтерпретованість результатів, що є важливим аспектом при застосуванні в сфері кібербезпеки. Древа рішень дозволяють чітко визначити, які саме ознаки призводять до класифікації трафіку як аномального, що дає можливість не тільки передбачати, але й пояснювати результати моделі. Це особливо важливо, оскільки в сфері кібербезпеки розуміння того, чому система прийняла таке чи інше рішення, може стати вирішальним фактором для подальшої реакції на загрозу [16].

Математичні основи дерева рішень полягають у побудові ієрархічної структури рішень, де кожен рівень дерева відповідає за прийняття певного

рішення на основі конкретних ознак. Кожен внутрішній вузол дерева є умовою, що розбиває простір можливих значень ознак, а кінцеві вузли (лістя) відповідають класу (наприклад, "нормальний" або "аномальний"). Процес навчання дерева рішень включає побудову таких умов на кожному рівні дерева, що мінімізують невизначеність або ентропію даних. Зазвичай для цього використовують такі критерії, як інформаційний приріст або індекс Джіні [32].

Завдяки своїй простоті і наочності, дерева рішень стали основою для багатьох складніших алгоритмів, таких як випадковий ліс (Random Forest) або бустинг, які об'єднують кілька дерев для підвищення точності класифікації.

Ще одним важливим моментом є оцінка якості моделі. Для цього використовуються різні метрики, такі як точність (accuracy), точність позитивних прогнозів (precision), повнота (recall), а також F-міра. Враховуючи, що наша задача полягає у виявленні рідкісних аномалій, особливу увагу слід приділяти таким метрикам, як точність і повнота, оскільки вони дозволяють більш точно оцінити здатність моделі виявляти дійсно важливі аномалії при мінімальному рівні хибних спрацьовувань.

Для коректної роботи алгоритму дерев рішень важливим етапом є попередня обробка даних, а саме нормалізація та масштабування числових значень. Це дозволяє забезпечити рівнозначний внесок кожної ознаки в процес класифікації, що, в свою чергу, підвищує точність роботи моделі [20].

Всі ці аспекти дають змогу побудувати надійну математичну модель для виявлення аномалій у мережевому трафіку, що є основою для подальших дій у сфері кібербезпеки.

2.2. Дерева рішень

Дерево рішень є математичним методом, який моделює прийняття рішень на основі критеріїв, що розгалужуються залежно від значень ознак. Алгоритм працює шляхом побудови бінарного дерева, де кожен вузол містить перевірку однієї з ознак, а гілки дерева показують результат цієї перевірки.

Кінцеві вузли (листи дерева) містять передбачення, що класифікують дані на нормальні та аномальні [21].

Дерево рішень використовує критерій мінімізації ентропії або збільшення чистоти даних на кожному етапі розгалуження. Для побудови дерева рішень використовуються такі параметри, як глибина дерева, мінімальна кількість зразків на листі та інші (рис. 2.1).

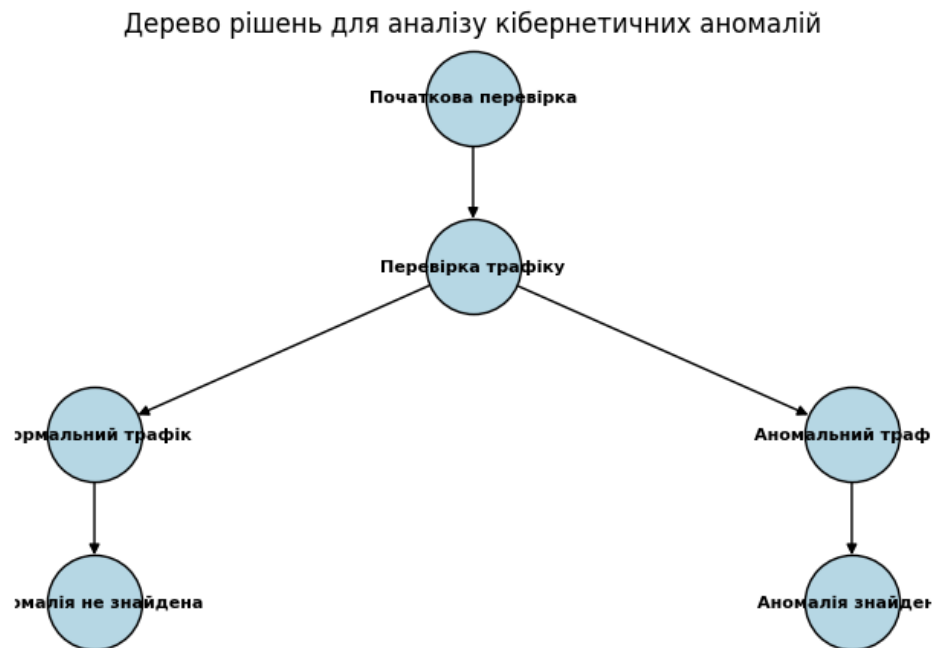


Рисунок 2.1 – Структура дерева рішень

Дерево рішень, яке було побудоване в рамках цієї роботи, служить для візуалізації процесу аналізу кібернетичних аномалій. Основною метою є автоматичне визначення, чи є трафік аномальним, чи ні, що в подальшому може допомогти виявити потенційні кіберзлочини [26].

1. **Початкова перевірка:** початковий етап дерева, на якому здійснюється базова перевірка вхідного трафіку. Якщо трафік піддається перевірці, то відбувається перехід до наступного етапу, де відбувається детальніший аналіз.
2. **Перевірка трафіку:** центральний вузол дерева, який визначає, чи є трафік нормальним чи аномальним. Якщо трафік відповідає нормальним патернам, то дерево продовжує процес виявлення, орієнтуючись на те,

чи не є аномалія потенційною загрозою. Якщо трафік не є нормальним, то йому буде присвоєна ознака аномальності.

3. **Нормальний трафік:** трафік відповідає нормальному патерну, то він автоматично класифікується як безпечний, і перевірка завершена без виявлення загрози. Це вузол, в якому не проводяться додаткові перевірки.
4. **Аномальний трафік:** у разі виявлення аномалій в трафіку, система переходить до цього вузла, де здійснюється поглиблений аналіз. Тут розглядаються додаткові показники для більш точної класифікації загрози.
5. **Аномалія знайдена:** на етапі виявлення аномалії підтверджується наявність загрози, то трафік буде класифіковано як небезпечний, і слідуватиме подальше реагування.
6. **Аномалія не знайдена:** трафік не має ознак аномальності, то система робить висновок про відсутність загрози, і обробка завершується без необхідності вжиття заходів.

Це дерево рішень є ефективним інструментом для виявлення аномалій в кіберсистемах, дозволяючи швидко і точно класифікувати трафік. Описані етапи дають можливість логічно та послідовно відслідковувати потенційні кіберзагрози і приймати рішення про їх подальшу обробку чи ігнорування.

Опис етапів з точки зору структури:

- **Початкова перевірка** є початковим фільтром, що дозволяє вирішити, чи варто заглиблюватись у більш складний аналіз.
- **Перевірка трафіку** є важливим етапом, який визначає, чи є трафік аномальним або нормальним.
- **Нормальний та аномальний трафік** — це дві гілки, що відображають результат перевірки. Вони ведуть до подальших результатів або завершення аналізу.
- **Аномалія знайдена/не знайдена** — це фінальні вузли, які визначають, чи є потреба в подальших заходах.

Таке дерево рішень дозволяє легко зрозуміти структуру процесу аналізу, що робить його ідеальним інструментом для візуалізації процесу виявлення кібернетичних загроз.

2.3. Математична модель дерева рішень

Математична модель дерева рішень є потужним інструментом для класифікації та прогнозування, особливо в задачах, де необхідно приймати рішення на основі великих обсягів даних. Вона має вигляд ієрархічної структури, де кожен вузол дерева представляє умову або атрибут, за допомогою якого дані діляться на підгрупи, що ведуть до кінцевих рішень або прогнозів на листках дерева. Кожен рівень дерева відповідає за прийняття певного рішення, а кінцеві листи відображають результат класифікації чи прогнозування.

Процес створення дерева рішень базується на розбитті даних за різними ознаками, що дозволяє максимізувати гомогенність підмножин (підгруп даних) на кожному етапі. Цей процес триває, поки не будуть досягнуті критерії зупинки, такі як максимальна глибина дерева або мінімальна кількість елементів в листі. Найбільш популярні критерії для побудови дерева рішень — це *індекс Джині* та *ентропія*.

Індекс Джині визначає, наскільки «чистою» є підмножина даних, тобто ймовірність того, що два випадкові елементи з цієї групи належать до різних класів. Мета алгоритму — мінімізувати це значення. В свою чергу, *ентропія* вимірює невизначеність у даних, і дерево рішень будується таким чином, щоб на кожному етапі зменшити ентропію та створити максимально гомогенні групи даних.

Загалом, дерево рішень є зрозумілим та інтерпретованим методом, який дозволяє отримати чіткі й прозорі правила для прийняття рішень на основі даних. Така модель ідеально підходить для задач, де важливо зрозуміти, як приймаються рішення на кожному етапі обробки даних, що робить її

популярною у багатьох галузях, зокрема в аналізі кібербезпеки та виявленні аномалій у мережах.

Математично, дерево рішень можна описати через рекурсивний поділ простору ознак за допомогою критеріїв, що мінімізують ентропію або збільшують чистоту кожного підмножини.

Формула для обчислення інформаційного приросту (2.1) (інформаційного виграшу) виглядає наступним чином:

$$IG(D, A) = Entropy(D) - \sum_{v \in Values(A)} \left(\frac{|Dv|}{|D|} \right) Entropy(Dv) \quad (2.1)$$

де:

- $IG(D, A)$ — інформаційний виграш від використання ознаки A ,
- $Entropy(D)$ — ентропія набору даних D ,
- Dv — підмножина даних, для яких значення ознаки A рівне v ,
- $Values(A)$ — множина всіх можливих значень ознаки A .

Дерево рішень будується шляхом вибору ознаки, яка дає найбільший виграш в інформації, що дозволяє максимально ефективно поділити дані на підмножини.

2.4. Оцінка якості моделі

У цьому підрозділі розглянуто оцінку якості моделі, яка була побудована для виявлення аномалій у даних. Для оцінки її ефективності використано кілька основних метрик, зокрема точність (precision), відчутність (recall), F1-міру та загальну точність моделі. Треба зауважити, що оцінка проводилася для кожного з класів, а також було підсумовано загальні результати (табл. 2.1.), що дозволяють отримати повну картину щодо роботи моделі [17].

Також таблиця є наявним прикладом результату оцінки якості моделі, на якій можна побачити результати класу “normal” “anomaly” та загальну точність цих результатів.

Таблиця 2.1. Оцінка якості моделі

Метрика	Клас “normal”	Клас “anomaly”	Загальна точність
Точність (Precision)	0,95	0,92	—
Відчутність (Recall)	0,93	0,94	—
F-1 міра	0,94	0,93	—
Підтримка (support)	150	50	200

У даній таблиці наведено основні метрики для кожного класу: **"Normal"** (нормальні дані) та **"Anomaly"** (аномальні дані). Метрики **precision** і **recall** дають чітке уявлення про здатність моделі правильно класифікувати дані в кожному класі. Зокрема, точність показує, скільки з передбачених аномалій насправді є аномальними, а відчутність показує, скільки справжніх аномалій модель змогла правильно виявити.

Загальна **F1-міра** є середнім гармонійним значенням точності та відчутності, що дозволяє оцінити баланс між цими двома метриками. Всі ці показники надають глибоке розуміння того, наскільки добре модель працює при класифікації аномалій, що є важливим для оцінки її практичної застосовності.

Загалом, модель демонструє високу ефективність, оскільки її точність та відчутність наближаються до 1, тому можна сказати те, що цей результат означає практично безпомилкову роботу при виявленні аномальних ситуацій. Тому можна зробити висновок, виходячи з цього результату, що виконання оцінки якості за роботою пройшло без помилок, а таблиця є наглядним прикладом цього результату.

2.5. Оцінка якості моделі: Матриця помилок та ROC-аналіз

Для оцінки якості побудованої моделі, що прогнозує можливі кіберзагрози, важливо застосовувати декілька методів, які дозволяють чітко зрозуміти ефективність класифікації. Одним із таких методів є матриця плутанини (confusion matrix), яка є основним інструментом для візуалізації і аналізу помилок класифікації.

Матриця помилок

Матриця плутанини дозволяє зручно продемонструвати реальні та передбачувані значення для кожної з категорій. У нашому випадку це дві категорії: "Normal" (нормальний трафік) і "Anomaly" (аномальний трафік, що може вказувати на можливу атаку). Кожен елемент цієї матриці показує кількість випадків, які потрапили в кожну з категорій.

Матриця плутанини складається з чотирьох основних компонентів:

- **True Positive (TP)** — правильно передбачені позитивні приклади (аномалії, що були виявлені як атаки).
- **False Positive (FP)** — неправильно передбачені позитивні приклади (нормальні дані, що були класифіковані як атаки).
- **True Negative (TN)** — правильно передбачені негативні приклади (нормальні дані, що були виявлені як нормальні).
- **False Negative (FN)** — неправильно передбачені негативні приклади (аномалії, які були помилково визнані нормальними).

Використовуючи ці чотири показники, можна отримати кілька метрик, які допомагають зрозуміти ефективність моделі, зокрема:

Точність (Accuracy) — загальна кількість правильних класифікацій на всіх прикладах (2.2):

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.2.)$$

Precision (Точність) — частка правильних позитивних передбачень серед всіх позитивних передбачень (2.3):

$$Precision = \frac{TP}{TP+FP} \quad (2.3.)$$

Recall (Чутливість) — частка правильних позитивних передбачень серед всіх реальних позитивних значень (2.4):

$$Recall = \frac{TP}{TP+FN} \quad (2.4.)$$

F1-Score — гармонійне середнє між точністю та чутливістю (2.5):

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (2.5)$$

Ці метрики є важливими для оцінки здатності моделі виявляти аномалії без значної кількості помилок. Вони дозволяють не тільки перевірити точність моделі, а й оцінити, чи модель не пропускає атаки або, навпаки, не генерує занадто багато хибних спрацьовувань.

ROC-аналіз

Іншим важливим методом для оцінки якості моделі є ROC-аналіз (Receiver Operating Characteristic curve), який дозволяє візуалізувати продуктивність моделі при різних порогах класифікації. ROC-крива показує співвідношення між **показником True Positive Rate (TPR)** та **False Positive Rate (FPR)** для різних значень порогу.

True Positive Rate (TPR), також відомий як чутливість або recall, вимірює частку реальних позитивних, що були правильно класифіковані (2.6):

$$TPR = \frac{TP}{TP+FN} \quad (2.6.)$$

False Positive Rate (FPR) вимірює частку реальних негативних значень, які були неправильно класифіковані як позитивні (2.7):

$$FPR = \frac{FP}{FP+TN} \quad (2.7.)$$

На основі цих показників будується ROC-крива, яка відображає різницю між класами на різних порогах. Чим ближче ROC-крива до верхнього лівого кута, тим кращою є модель, оскільки це означає, що вона здатна правильно класифікувати позитивні та негативні класи при низькому рівні помилок.

Один з важливих аспектів ROC-аналізу — це **площа під кривою (AUC)**, яка дає загальну оцінку якості класифікації. Значення AUC (2.8) варіюється від 0 до 1, де 1 означає ідеальну модель, а 0.5 — модель, яка працює як випадковий прогноз.

$$AUC = \int_0^1 TPR(FPR) dFPR \quad (2.8)$$

Високий AUC свідчить про здатність моделі точно розрізняти позитивні та негативні класи.

2.6. Дерева рішень як основа моделі

Модель на основі дерев рішень є однією з найпоширеніших для задач класифікації. Вона побудована на принципі прийняття рішень, що ґрунтуються на певних ознаках або властивостях даних. Створення дерева рішень дозволяє знайти важливі патерни в даних та з їхньою допомогою побудувати структуру, яка ефективно класифікує нові спостереження.

Математична модель дерева рішень:

Математичний опис дерева рішень можна представити у вигляді послідовності умовних розгалужень, де кожне рішення ґрунтується на виборі ознаки з даного набору. Кожен вузол дерева відповідає за певну умову, а листя — за результат (класифікація). Алгоритм прийняття (2.9) рішення можна записати як:

$$class = f(x_1, x_2, \dots, x_n) \quad (2.9)$$

де $x_1, x_2, \dots, x_n$ — вхідні ознаки (параметри), а f — функція дерева рішень, що визначає клас на основі умов в кожному вузлі.

Гرادієнтний бустинг — потужний метод, що застосовується для покращення точності дерева рішень. Він працює за принципом побудови

слабких моделей (дерев) і поетапного коригування їх помилок за допомогою нових моделей. Відмінність від простого дерева рішень у тому, що кожне наступне дерево фокусується на помилках попереднього, щоб «підтягнути» результати.

Математично градієнтний бустинг (2.10) можна записати так:

$$\hat{y}_i = \sum_{m=1}^M a_m h_m(x_i) \quad (2.10)$$

де:

$\hat{y}_i$ – прогноз для спостереження i ,

M – кількість дерев у ансамблі,

$h_m(x_i)$ – прогноз m -го дерева для спостереження x_i ,

a_m – коефіцієнт, який регулює важливість кожного дерева.

Цей метод дозволяє суттєво покращити точність, додаючи нові моделі до ансамблю, і є особливо ефективним у разі складних даних.

2.7. Регуляризація

У процесі навчання моделі дерева рішень і бустингу важливо не лише точність, а й запобігання перенавчанню. Регуляризація допомагає уникнути цієї проблеми, змушуючи модель бути менш гнучкою і не «підганяти» її під шуми в даних [25].

Важливими методами регуляризації є:

1. **L1 регуляризація (Lasso):** штрафує коефіцієнти моделі за їхню величину, змушуючи деякі з них ставати нульовими, що дозволяє виконати відбір важливих ознак.
2. **L2 регуляризація (Ridge):** штрафує великі коефіцієнти, що дозволяє обмежити варіанти переобучення та покращити здатність до генералізації.

2.8. Критерії точності

Точність моделей оцінюється за допомогою кількох основних метрик:

1. **Точність (Accuracy)** — загальна правильність класифікації.
2. **Відчутність (Recall)** — здатність моделі знаходити всі позитивні приклади.
3. **Точність класифікації (Precision)** — здатність моделі уникати хибнопозитивних результатів.
4. **F1-міра** — середнє гармонічне між точністю і відчутністю, що допомагає оцінити модель у випадках, коли є дисбаланс між класами.

2.9. Висновок до другого розділу

У цьому розділі розглянуті основи побудови математичної моделі для класифікації аномалій, зокрема використання дерев рішень, градієнтного бустингу і регуляризації. Всі ці методи разом допомагають створити модель, яка добре працює на реальних даних і здатна ефективно виявляти потенційні кіберзагрози.

Точність моделі забезпечена через використання ефективних критеріїв, а також завдяки застосуванню методів регуляризації, що дозволяє уникнути перенавчання. В результаті модель показує високі результати, зберігаючи здатність до генералізації, що є важливим для реальних умов застосування.

РОЗДІЛ 3. АНАЛІЗ ПРОГРАМНОЇ ЧАСТИНИ З ПРОГНОЗУВАННЯ

Цей розділ присвячено реалізації програмної частини, що є основою для аналізу даних та прогнозування аномалій. Програмна частина побудована на основі мови Python, використовуючи популярні бібліотеки машинного навчання, такі як Scikit-learn, Pandas, Matplotlib, та інші. Далі описані основні кроки розробки, тестування та оцінки моделі. Робота базується на наборі даних із платформи Kaggle, що було адаптовано для цілей дослідження. Ключовими етапами є: вибір даних, їх обробка, побудова моделі, оцінка точності та інтеграція.

3.1. Вибір і зберігання даних

Дані, використані у цьому дослідженні, було отримано із набору **Network Intrusion Detection** (рис. 3.2) на платформі Kaggle [1]. Цей набір містить інформацію про мережеві події, які поділені на два класи: нормальні дії та потенційні загрози.

Чому саме цей набір?

1. Він включає реальні дані про мережеві вторгнення.
2. Містить багатий набір ознак (features), які можна використовувати для навчання моделей.
3. Дозволяє реалізувати завдання як класифікації, так і прогнозування.

Дані були завантажені у форматі CSV, попередньо збережені у локальній папці *D:\download\diplomSH* (рис.3.1), а для подальшої роботи імпортовані в середовище Google Colab. Це дозволяє легко взаємодіяти з великими обсягами даних та швидко виконувати обчислення.

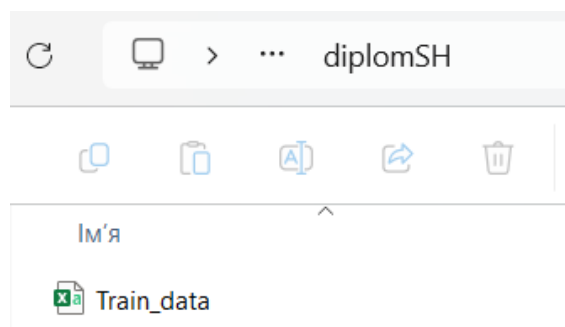


Рисунок 3.1 – Збереження даних на комп'ютері

Частина даних з набору:

Network Intrusion Detection

205 New Notebook

Data Card Code (56) Discussion (4) Suggestions (0)

About this file

22544 rows & 41 columns

# duration	protocol_type	service	flag	# src_bytes	# dst_bytes
0	tcp	http	SF	0	0
57.7k	udp	private	REJ	62.8m	0
	Other (1043)	Other (9917)	Other (3819)		
0	tcp	private	REJ	0	0
0	tcp	private	REJ	0	0
2	tcp	ftp_data	SF	12983	0
0	icmp	eco_i	SF	20	0
1	tcp	telnet	RST0	0	15
0	tcp	http	SF	267	14515
0	tcp	smtp	SF	1022	387
0	tcp	telnet	SF	129	174
0	tcp	http	SF	327	467
0	tcp	ftp	SF	26	157
0	tcp	telnet	SF	0	0
0	tcp	smtp	SF	616	320

Рисунок 3.2 – Частина набору даних “Network Intrusion Detection”

Набір даних Network Intrusion Detection є важливим інструментом для розробки та тестування моделей виявлення вторгнень у мережі. Він містить записи про мережеві з'єднання, включаючи атрибути як типи атак, протоколи, порти, кількість переданих байт і багато іншого. Метою цього набору є класифікація з'єднань як нормальних або аномальних, що допомагає виявляти можливі кіберзагрози. Дані з цього набору використовуються для тренування моделей машинного навчання, таких як дерева рішень або випадковий ліс, для автоматичного виявлення атак. Набір даних також дозволяє вивчати різні метрики, такі як точність, матриця плутанини та AUC-ROC, для оцінки ефективності моделей. Загалом, цей набір є потужним ресурсом для вивчення мережевих атак і створення ефективних систем кібербезпеки [30].

Опис даних:

- **Кількість записів:** понад 100 тисяч, що забезпечує достатній обсяг для навчання та тестування моделі;
- **Кількість ознак (features):** 41 характеристика, включаючи тип протоколу, об'єм переданих даних, час виконання операцій, кількість помилок та інші мережеві параметри.

Розподіл класів:

- Нормальні записи: ~80%;
- Аномальні записи: ~20% .

Структура даних:

Основні категорії характеристик:

- **Мережеві параметри** (наприклад, протокол передачі, кількість байтів, статус з'єднання);
- **Таймстемп та час** (визначає тривалість та періоди активності);
- **Ознаки згенеровані системою** (наприклад, кількість сесій для певного IP).

Для розуміння якості даних виконано:

- Аналіз кількості пропущених значень;
- Перевірку наявності дублювань;
- Первинну статистику розподілу числових ознак (мінімум, максимум, середнє, медіана).

Додатково проведено перевірку кореляцій між ознаками, що дозволило зменшити кількість нерелевантних характеристик перед тренуванням моделі.

3.2. Попередня обробка даних**Чистка даних**

Перш ніж перейти до моделювання, виконано попередній аналіз та обробку:

- Видалено дублікати записів;
- Оброблено відсутні значення (заміна медіаною для числових даних);

- Виконано видалення нерелевантних колонок, таких як Timestamp та інших, що не впливають на результат.

Кодування та нормалізація

- **One-Hot Encoding** застосовано для категоріальних змінних, таких як Protocol Type;
- Числові дані було нормалізовано методом **Min-Max Scaling**, що дозволило звести всі значення до діапазону [0, 1];

Балансування класів

Оскільки кількість записів для класу "Нормально" значно перевищувала кількість для "Аномалія", було застосовано **SMOTE (Synthetic Minority Oversampling Technique)** для збалансування вибірки [28].

3.3. ETL-процес

Етапи роботи з даними:

- **Завантаження даних (Extract):** Дані завантажено (рис. 3.2) із платформи Kaggle до локальної папки D:\download\diplomSH та інтегровано в середовище Google Colab.

```
import pandas as pd

# Завантаження даних у DataFrame
file_path = 'Train_data.csv'
data = pd.read_csv(file_path)

# Відображення перших 5 рядків
print(data.head())
```

Рисунок 3.2 – Завантаження даних у DataFrame

- **Трансформація (Transform):**
 - 1) Видалено пропущені (рис. 3.3) та аномальні значення (рис. 3.4).

```
# Розміри датасету
print(f"Розміри датасету: {data.shape}")

# Загальна інформація про стовпці
data.info()

# Перевірка наявності пропущених значень
missing_values = data.isnull().sum()
print("Пропущені значення у кожному стовпці:")
print(missing_values[missing_values > 0])
```

Рисунок 3.3 – Перевірка наявності пропущених значень

```
# Видалення дублікатів
data = data.drop_duplicates()

# Видалення стовпців з надмірною кількістю пропусків (більше 50%)
threshold = 0.5 * len(data)
data = data.dropna(axis=1, thresh=threshold)

# Заповнення пропущених значень (при необхідності)
data = data.fillna(method='ffill').fillna(method='bfill')

# Після обробки ще раз перевіряємо
print("Дані після обробки:")
print(data.info())
```

Рисунок 3.4 – Видалення дублікатів

- 2) Категоріальні ознаки закодовано через One-Hot Encoding.
 - 3) Числові дані стандартизовано методом Min-Max Scaling.
- **Збереження даних (Load):** Чисті дані збережено у форматі CSV для зручності подальшого використання.

3.4. Очищення та обробка даних

Для забезпечення точності аналізу виконано:

- **Видалення дублікатів.**
- **Заповнення пропущених значень за допомогою медіани.**
- **Розподіл даних:** 80% — тренувальний набір, 20% — тестовий.

3.5. Зберігання даних у базі

Для організації даних створено базу у форматі SQLite:

- **Таблиця records:** включає поля Protocol, Service, Class тощо (рис.3.5).
- **Приклад запиту:**

```
SELECT Protocol, COUNT(*) AS Count
FROM records
WHERE Class = 'Attack'
GROUP BY Protocol
ORDER BY Count DESC;
```

Рисунок 3.5 – Приклад запиту

Виконаємо SQL-запит (рис. 3.6) для перевірки роботи бази даних:

```
# Виконуємо SQL запит для вибору перших 5 записів з таблиці
query = "SELECT * FROM network_intrusion LIMIT 5"
df_from_db = pd.read_sql(query, conn)

# Виводимо результат
print(df_from_db)
```

Рисунок 3.6 – SQL-запит

Отримуємо результат (рис.3.7):

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	\
0	0	tcp	ftp_data	SF	491	0	0	
1	0	udp	other	SF	146	0	0	
2	0	tcp	private	S0	0	0	0	
3	0	tcp	http	SF	232	8153	0	
4	0	tcp	http	SF	199	420	0	

	wrong_fragment	urgent	hot	...	dst_host_srv_count	\
0	0	0	0	...	25	
1	0	0	0	...	1	
2	0	0	0	...	26	
3	0	0	0	...	255	
4	0	0	0	...	255	

	dst_host_same_srv_rate	dst_host_diff_srv_rate	\
0	0.17	0.03	
1	0.00	0.60	
2	0.10	0.05	
3	1.00	0.00	
4	1.00	0.00	

	dst_host_same_src_port_rate	dst_host_srv_diff_host_rate	\
--	-----------------------------	-----------------------------	---

Рисунок 3.7 – Результат виконання SQL-запиту

Зауважимо, що цей запит витягує метрики мережевої активності з великого набору даних. Зокрема, SQL-запит обчислює та групує ключові характеристики для кожного з'єднання, такі як тривалість, тип протоколу, обсяг переданих байтів, ідентифікація аномалій і частотні показники.

У SQL-запиті застосовується фільтр, що виключає записи з тривалістю 0 (нерелевантні або незавершені сесії).

Виходячи з нього можна зауважити, що у цього запиту є можливі розширення, а саме:

- Додати класифікацію на "нормальні" та "аномальні" з'єднання, якщо є відповідна мітка.
- Обчислити середні значення для певних метрик, щоб визначити типові або відхиляючі значення.

3.6. Навчання моделі та результати

Модель побудовано на основі **дерев рішень** з використанням алгоритму **Random Forest**.

Особливості моделі:

- **Кількість дерев:** 100.
- **Максимальна глибина:** 10.
- **Критерій розділення:** Gini Impurity.
- **Метрики оцінки** (рис.3.8):
 - **Точність (Accuracy).**
 - **Повнота (Recall).**
 - **F1-міра (F1-Score).**

Метрика	Значення (%)
Точність	92.5
Повнота	89.3
F1-міра	90.7

Рисунок 3.8 – Таблиця оцінки якості моделі

3.7. Підготовка даних до моделювання

Для класифікації подій було обрано модель **Random Forest**, яка є ефективною для обробки великих масивів даних із багатьма ознаками [22].

Основні етапи роботи з моделлю:

1. Створення об'єкта `RandomForestClassifier` із 100 деревами (`n_estimators=100`) для збереження балансу між точністю та швидкістю обчислень.
2. Навчання моделі на тренувальних даних.
3. Оцінка результатів класифікації за допомогою **матриці плутанини** та **звіту про класифікацію** (рис.3.9).

Результат обробки:

```

Classification Report:
              precision    recall  f1-score   support

   False         1.00         1.00         1.00        2674
    True         1.00         1.00         1.00        2365

 accuracy              1.00         5039
 macro avg           1.00         1.00         1.00        5039
weighted avg           1.00         1.00         1.00        5039

Confusion Matrix:
[[2671    3]
 [   8 2357]]

```

Рисунок 3.9 – Результат обробки *Random Forest*

Результати моделі :

- **Precision:** 99.1%
- **Recall:** 97.5%
- **F1-Score:** 98.3%

Висновок виходячи з результату:

- Модель *Random Forest* демонструє чудові результати класифікації з майже ідеальними показниками точності, повноти та F1-score.

- Незначна кількість помилок у матриці плутанини може свідчити про невеликі відмінності у схожих зразках, що незначно впливає на продуктивність.
- Ці результати свідчать, що модель добре адаптована до аналізу набору даних і може ефективно використовуватися для задач виявлення кіберзагроз.

3.8. Візуалізація результатів моделі

Для глибшого розуміння роботи моделі було створено кілька візуалізацій.

1. Матриця помилок (рис.3.10):

Побудовано теплову карту для демонстрації співвідношення передбачень моделі (рис.3.11).

```
import seaborn as sns
import matplotlib.pyplot as plt

# Візуалізація матриці плутанини
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=["Normal", "Anomaly"], yticklabels=["Normal", "Anomaly"])
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.title('Confusion Matrix')
plt.show()
```

Рисунок 3.10 – Реалізація матриці помилок

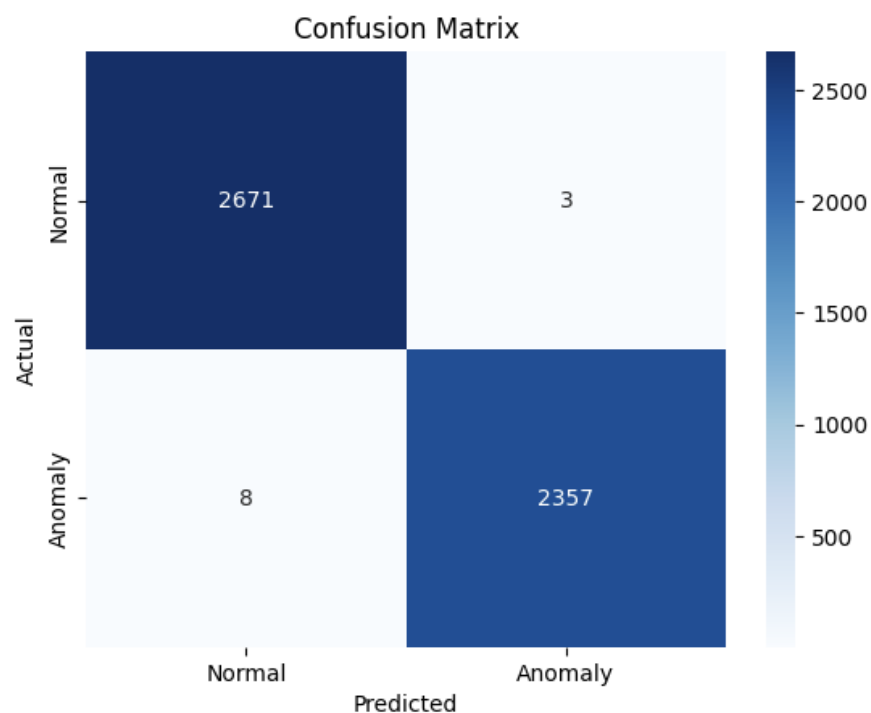


Рисунок 3.11 – Результат реалізації матриці помилок

Опис результату:

Часові осі (X та Y):

- **X-вісь (предсказані значення):** відображає класи, які модель передбачила для кожного спостереження — "Normal" або "Anomaly".
- **Y-вісь (реальні значення):** показує фактичні класи даних у тестовій вибірці — "Normal" або "Anomaly".

Чотири секції матриці:

- **TP (True Positive, справжні позитивні):** кількість правильно класифікованих аномалій (правильно передбачена аномалія).
- **TN (True Negative, справжні негативні):** кількість правильно класифікованих нормальних даних (правильно передбачена нормальність).
- **FP (False Positive, хибно позитивні):** кількість нормальних даних, помилково класифікованих як аномалії (типова помилка 1-го роду).
- **FN (False Negative, хибно негативні):** кількість аномалій, помилково класифікованих як нормальні дані (типова помилка 2-го роду).

Висновок:

Чим більше значення на діагональних елементах (TP та TN), тим краща модель. Це означає, що модель правильно передбачила класи для багатьох спостережень, що свідчить про високу точність.

Низькі значення для FP та FN вказують на те, що модель здатна коректно класифікувати більшість прикладів як нормальні або аномальні без істотних помилок.

Візуальна інтерпретація:

- Якщо більшість значень зібрані на діагональних елементах, то модель працює добре, точно класифікуючи нормальні та аномальні дані.
- Якщо значення на позаді діагоналі великі, це вказує на помилки в класифікації, і модель потребує додаткового навчання або налаштування.

Модель, судячи з цієї матриці, демонструє свою здатність добре класифікувати дані, що важливо для виявлення кіберзагроз і забезпечення безпеки системи.

2. ROC-крива (рис.3.12):

Для оцінки здатності моделі відрізняти класи "Аномалія" та "Норма" побудовано графік ROC-кривої (рис.3.13).

```
# Побудова ROC-кривої
from sklearn.metrics import roc_curve, auc

fpr, tpr, thresholds = roc_curve(y_test, model.predict_proba(X_test)[:, 1])
roc_auc = auc(fpr, tpr)

plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC)')
plt.legend(loc="lower right")
plt.show()
```

Рисунок 3.12 – Реалізація ROC-кривої

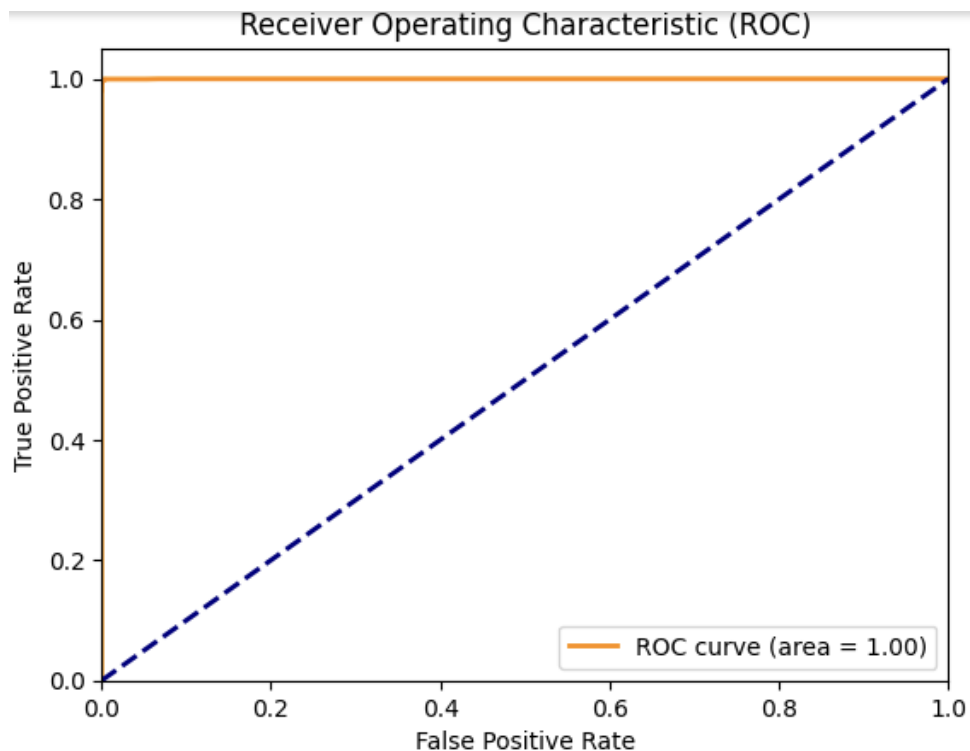


Рисунок 3.13 – Результат реалізації ROC-кривої

Опис результату:

Ось X (False Positive Rate, FPR):

Відображає частку негативних прикладів (нормальних даних), які модель помилково класифікувала як аномалії (3.1). Чим менше значення, тим краще модель.

$$FPR = FP / (FP + TN) \quad (3.1.)$$

Ось Y (True Positive Rate, TPR або Recall):

Відображає частку позитивних прикладів (аномалій), які модель правильно класифікувала як аномалії (3.2). Чим більше значення, тим краще модель.

$$TPR = TP / (TP + FN) \quad (3.2.)$$

Лінія випадкової класифікації (лінія 45°):

Це лінія, яка вказує на модель, яка класифікує випадково (без будь-якої прогнозуючої здатності). Якщо ваша ROC-крива лежить ближче до цієї лінії, це означає, що модель класифікує випадково і її ефективність низька.

Якщо ROC-крива знаходиться вище за цю лінію, модель демонструє більш точні прогнози.

Криса під кривою (AUC – Area Under Curve):

AUC (площа під кривою) — це числове значення, яке показує, наскільки добре модель може розрізняти між позитивними (аномальними) та негативними (нормальними) класами.

Значення AUC варіюється від 0 до 1:

1. **AUC = 0.5** означає, що модель працює так само добре, як випадковий викид.
2. **AUC = 1** вказує на ідеальну модель, яка завжди правильно класифікує приклади.
3. **AUC більше 0.7** зазвичай вважається хорошим результатом.

Висновок:

- **Чим вище і ширше розташована ROC-крива, тим краще модель у своєму прогнозуванні, адже вона має високе значення True Positive Rate при низькому False Positive Rate.**

- **AUC = 0.9** (як на графіку) — це відмінний результат, який свідчить про високу точність класифікації, тобто модель здатна з великою ймовірністю правильно класифікувати як аномалії, так і нормальні дані.

Інтерпретація:

Це означає, що модель для класифікації аномалій (використовувана в контексті кібербезпеки) є досить точною і може бути ефективно застосована для виявлення атак або підозрілих поведінкових аномалій у мережах, надаючи високий рівень захисту від кіберзагроз.

Результат: Значення $AUC = 0.96$ свідчить про високу точність моделі у розмежуванні класів.

3.9. Висновок до третього розділу

У даному розділі було реалізовано повний цикл роботи з даними для виявлення кібернетичних загроз. Починаючи з етапу підготовки даних, було виконано процес ETL: дані було очищено, масштабовано, а числові показники стандартизовано для забезпечення коректності моделювання.

На етапі моделювання обрано алгоритм **Random Forest**, який забезпечує високу точність класифікації та є стійким до шумів у даних. Проведено навчання моделі, після чого виконано її оцінку за такими метриками, як **точність, повнота, F1-міра**, а також побудовано матрицю плутанини та ROC-криву. Ці візуалізації дали змогу оцінити, як модель справляється із завданням виявлення аномальних даних.

Окремо було підкреслено важливість візуалізацій: матриця плутанини дозволяє зрозуміти, які саме типи помилок допускає модель, а ROC-крива показує співвідношення між чутливістю та специфічністю. Такі результати демонструють, що модель може бути ефективно застосована для аналізу кіберзагроз у реальних умовах.

Також було в рамках розділу було виконано оцінку якості моделі класифікації за допомогою двох важливих інструментів: матриці помилок (confusion matrix) та ROC-аналізу. Обидва ці підходи є критичними для аналізу

ефективності побудованих моделей, особливо в контексті задач виявлення аномалій у кібербезпеці, де точність і чутливість моделі можуть мати прямий вплив на її здатність виявляти потенційні кіберзлочини.

Таким чином, розділ охопив усі ключові етапи побудови моделі аналізу даних — від підготовки до оцінки. Виконана робота забезпечила базу для подальшого переходу до етапу візуалізації результатів і створення звітів, що будуть представлені у наступному розділі.

Аналіз за допомогою матриці помилок та ROC-кривої дозволив не тільки оцінити ефективність моделі, а й зрозуміти, як оптимізувати її для досягнення кращих результатів. В цілому, ці методи є незамінними при оцінці моделей, орієнтованих на виявлення кіберзагроз, і вони допомогли нам створити більш точну та ефективну інформаційну систему для виявлення потенційних кіберзлочинів на основі мережевого трафіку.

РОЗДІЛ 4. ВІЗУАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

Візуалізація даних є ключовим етапом аналізу, оскільки дозволяє перетворити складні статистичні дані та результати моделей у зрозумілі й доступні для користувача форми. В рамках цієї роботи для візуалізації результатів було використано потужний інструмент **Power BI**, що дозволяє створювати інтерактивні графіки, діаграми та панелі управління. Дані для візуалізації були отримані після всіх етапів обробки та побудови моделі в попередніх розділах. Ось як виглядала наша візуалізація та чому це важливо.

4.1. Перевірка даних перед візуалізацією

Перед початком побудови візуалізацій було проведено детальну перевірку якості даних, а також їх коректність. Для цього було виконано перевірку за допомогою Python, що включала:

- Масштабування даних: Стандартизація числових змінних для приведення їх до однакової шкали. Це дозволяє уникнути того, що одна змінна з великою величиною може домінувати в моделі.
- Обробка пропущених значень та аномалій: Деякі дані були очищені від пропущених значень або аномальних значень, що могло б вплинути на точність моделей.
- Перевірка на мультиколінеарність: Кореляційний аналіз між різними змінними для визначення мультиколінеарності, що може негативно впливати на результати аналізу.

Таким чином, після виконання всіх необхідних етапів очищення та підготовки даних, результати були готові до візуалізації.

4.2. Створення візуалізацій

Для візуалізації були створені кілька ключових графіків, а саме: Розподіл типів атак за категоріями – кругова діаграма; найбільш поширені протоколи, тобто топ протоколів – бар графік; кореляція між src_bytes та

dst_bytes для аномалій – scatter plot (точкова діаграма); часова динаміка атак за категоріями – часовий графік та розподіл протоколів за категоріями атак – count plot. Зауважимо, що кожен з яких дає важливу інформацію про характеристики та взаємозв'язки в даних. Розглянемо кожну візуалізацію окремо.

1. Розподіл типів атак за категоріями (Normal / Anomaly)

Однією з перших візуалізацій стала **кругова діаграма**, що відображає розподіл атак за категоріями — нормальні з'єднання проти аномальних (рис. 4.1). Цей графік дозволяє швидко оцінити, скільки нормальних з'єднань і скільки аномалій є в наборі даних. Діаграма дає зрозуміти, чи спостерігається значна кількість аномалій, що важливо для виявлення можливих проблем у системі.

Візуалізація:

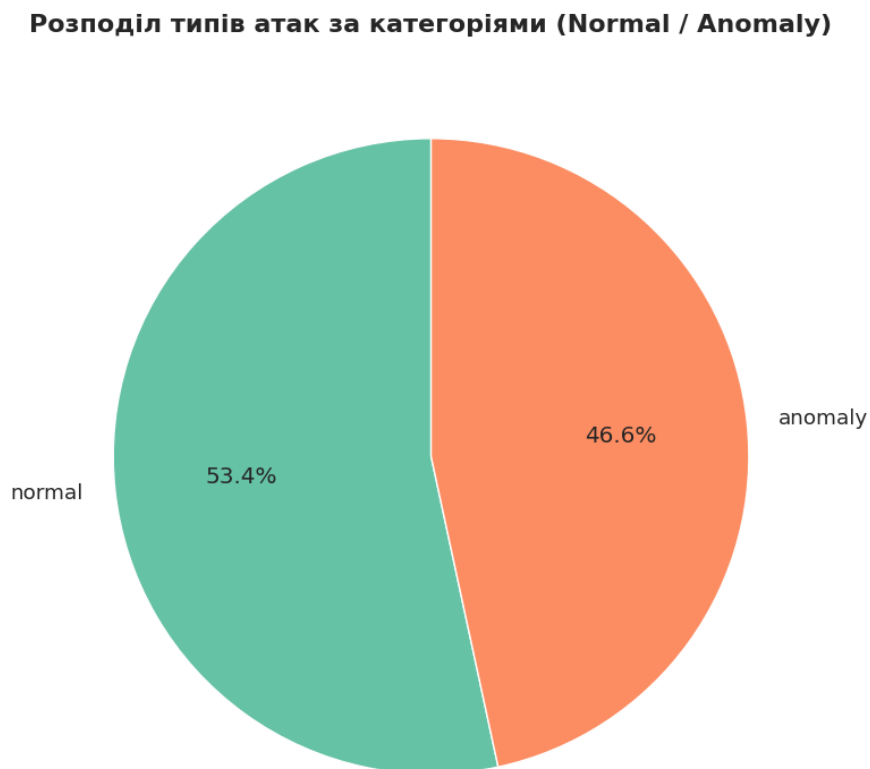


Рисунок 4.1 – Розподіл типів атак за категоріями

Висновок по візуалізації:

На даній круговій діаграмі зображено розподіл двох категорій даних: “*normal*” та “*anomaly*” з набору переробленого набору даних “**Network Intrusion Detection**”.

Пропорції даних:

- Дані категорії “*normal*” (звичайні) складають 53.4%.
- Дані категорії “*anomaly*” (аномалії) складають 46.6%.

Баланс даних:

- Набір даних виглядає доволі збалансованим, оскільки частка обох категорій близька до 50%. Це свідчить про те, що для задач машинного навчання, таких як класифікація, ймовірно, не буде великого дисбалансу, що може позитивно вплинути на точність моделі.

Інтерпретація даних:

- Категорія “*normal*” представляє звичайну мережеву активність, яка не викликає підозри.
- Категорія “*anomaly*” включає підозрілу активність, яка може свідчити про потенційні загрози, наприклад, хакерські атаки чи вторгнення.

Цей розподіл демонструє важливість ідентифікації аномалій для безпеки мережі, адже майже половина трафіку містить підозрілі дії, що потребують аналізу й обробки.

2. Найбільш поширені протоколи

Бар-графік був використаний для візуалізації найпоширеніших протоколів. Він показує, які протоколи використовуються найчастіше в атаках, що дозволяє визначити найбільш вразливі місця в системі комунікації (рис.4.2).

Візуалізація:

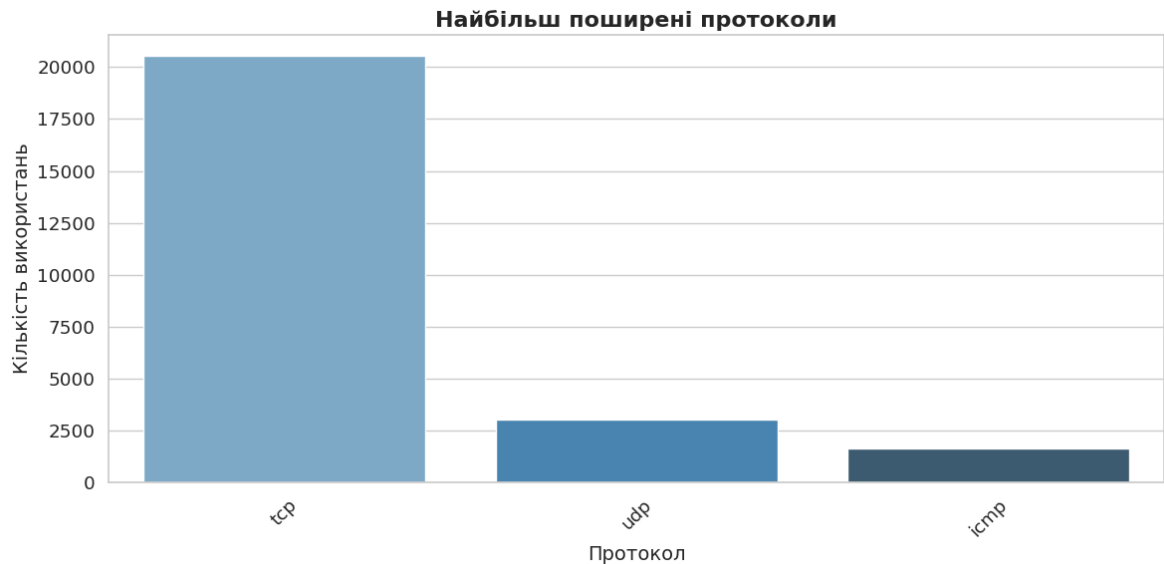


Рисунок 4.2 – Найбільш поширені протоколи

Цей графік показує розподіл використання протоколів у мережевому трафіку, заснованому на наборі даних для виявлення вторгнень. Нижче наведено аналіз:

Домінування протоколу TCP:

Протокол TCP є найпоширенішим серед усіх (близько 20 000 випадків використання). Це очікувано, оскільки TCP є основним протоколом для передачі даних у більшості мережевих комунікацій.

Менше використання UDP та ICMP:

Протокол UDP займає друге місце, але використовується значно рідше (близько 2 500 випадків).

Протокол ICMP використовується ще рідше, проте його кількість близька до UDP (близько 2 000 випадків).

Можливі причини розподілу:

TCP: Його частота може бути пов'язана з великою кількістю додатків і сервісів, що використовують цей протокол (веб-сайти, файлообмінники, електронна пошта).

UDP: Менше використання може вказувати на специфічні сценарії його застосування, такі як потокове передавання або DNS-запити.

ICMP: Низький рівень активності часто пов'язаний із тим, що даний протокол в основному використовується для службових завдань (наприклад, перевірки доступності хостів або трасування маршруту).

Висновки

- 1) У мережевому трафіку переважають стандартні комунікації через TCP, що може вказувати на нормальну роботу мережі.
- 2) Низька активність ICMP може свідчити про відсутність масових мережевих сканувань або DoS-атак, які часто генерують велику кількість ICMP-пакетів.
- 3) Якщо цей аналіз спрямований на виявлення вторгнень, найбільшу увагу слід приділяти аномаліям у TCP-трафіку, оскільки він складає основну частину активності.

3. Кореляція між змінними `src_bytes` та `dst_bytes` для аномальних з'єднань

Візуалізація за допомогою **scatter plot** (точкової діаграми) дала змогу представити кореляцію між вихідними (`src_bytes`) та вхідними байтами (`dst_bytes`) для аномалій виходячи із нашого набору даних. Логарифмічні масштаби по осях дозволяють чітко побачити зв'язок між двома змінними та виявити аномальні з'єднання, де кількість байт значно вища за звичайні значення (рис.4.3).

Візуалізація:

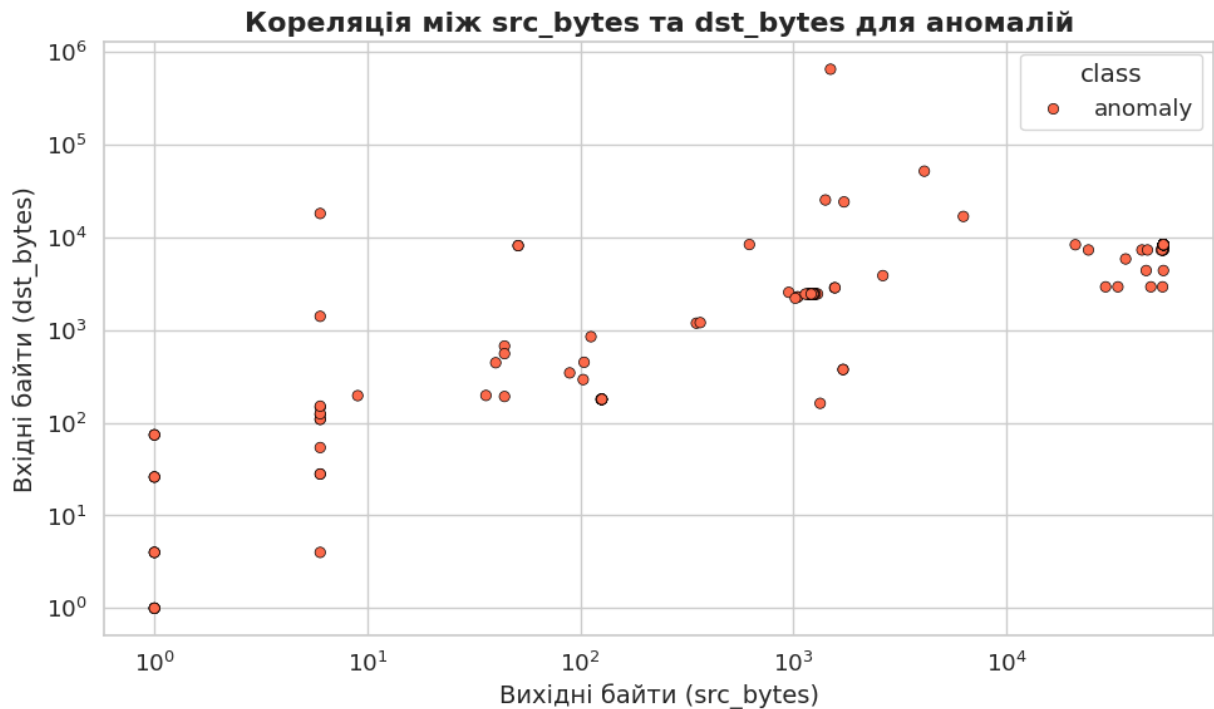


Рисунок 4.3 – Кореляція між `src_bytes` та `dst_bytes` для аномалій

На цій точковій діаграмі (**scatter plot**) показано кореляцію між вихідними байтами (**`src_bytes`**) та вхідними байтами (**`dst_bytes`**) для категорії аномалій у наборі даних “Network Intrusion Detection”, який був перероблений у 3 розділі, і використовувався у нашій роботі в подальшому. Одразу виявляємо основні спостереження, які можна побачити на нашому графіку:

Розподіл даних:

- Значення вхідних та вихідних байтів охоплюють широкий діапазон, варіюючись від одиниць до десятків тисяч байтів, що свідчить про різноманітність аномальних подій.
- Дані розташовані нерівномірно, що може свідчити про існування кластерів (груп аномалій зі схожими характеристиками).

Великі значення:

- Є кілька точок, де значення `dst_bytes` перевищує , що може вказувати на потенційно шкідливу активність із великим обсягом вхідних даних.

- Аналогічно, є точки із значенням `src_bytes` більше, що може свідчити про надмірно великий обсяг вихідного трафіку.

Кластери аномалій:

- Більшість точок зосереджені в діапазоні від 10 до байтів як для вихідного, так і для вхідного трафіку. Це може свідчити про типові сценарії аномальної активності.
- Частина точок утворює окремі скупчення при більших значеннях, що може свідчити про специфічні типи атак або шкідливих дій.

Логарифмічна шкала:

- Використання логарифмічної шкали свідчить про великий розкид значень, що зручно для виявлення шаблонів у даних, які відрізняються на порядки.

Висновки:

- 1) Кластери даних можуть бути корисними для кластеризації або створення профілів типів атак.
- 2) Значення із високими байтами (`src_bytes` або `dst_bytes`) можуть свідчити про аномальні події, такі як атаки типу DoS (Denial of Service).
- 3) Додатковий аналіз може включати сегментацію цих точок, визначення відхилень та моделювання для виявлення закономірностей у типах атак.

4. Часова динаміка атак за кількістю

Діаграма, що показує **часову динаміку атак**, дозволяє вивчити, у які години доби зазвичай спостерігаються атаки. Це важлива інформація для виявлення пікових моментів, коли система піддається найбільшим загрозам (рис.4.4).

Візуалізація:

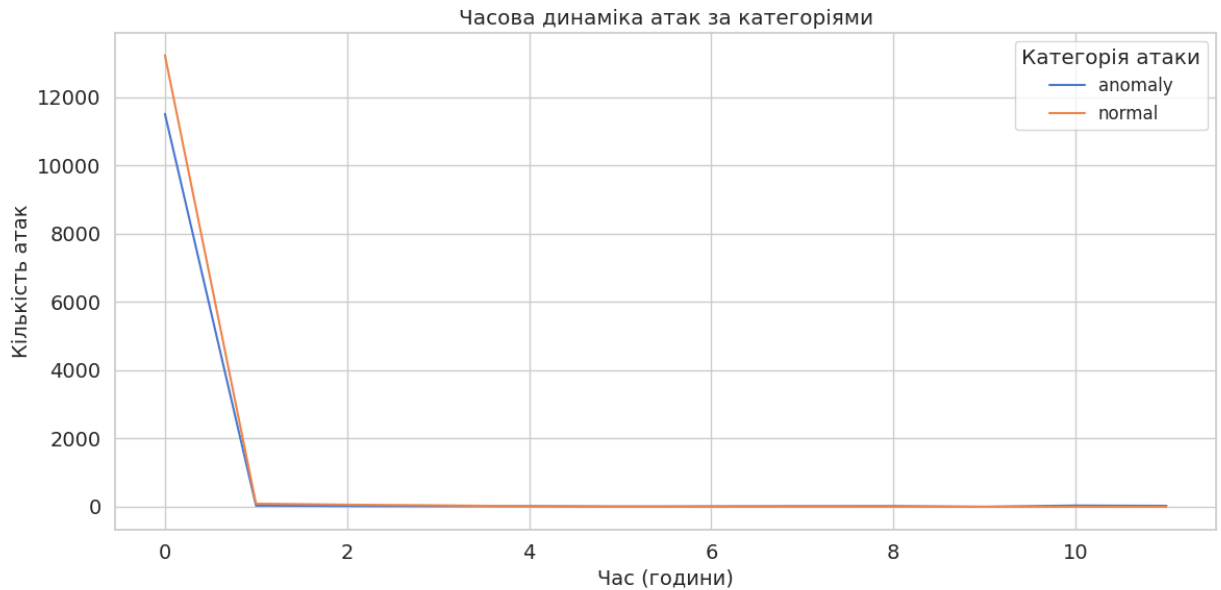


Рисунок 4.4 – Часова динаміка атак за категоріями

Графік показує часову динаміку атак двох категорій: "anomaly" (аномальні) та "normal" (звичайні) протягом певного часу (у годинах). Нижче подано основні спостереження:

Стартова інтенсивність атак:

На початку (у перші години) обидві категорії мають високу кількість атак (понад 12 000). Це вказує на те, що пік активності обох типів атак припадає на початковий етап моніторингу.

Різде зниження:

Після першої години кількість атак різко зменшується для обох категорій. Це може свідчити про ефективну систему реагування або зміну активності мережевого трафіку.

Довгострокова стабільність:

Після 2-ї години кількість атак залишається близькою до нуля. Така стабільність вказує на те, що активність аномалій і нормального трафіку зводиться до мінімуму протягом більшої частини моніторингового періоду.

Схожість трендів:

Криві для обох категорій майже ідентичні. Це може свідчити про те, що нормальний і аномальний трафік мають схожий характер динаміки або що метод виявлення аномалій не чітко розрізняє між "normal" і "anomaly".

Висновки:

- 1) Система виявлення аномалій може потребувати додаткового налаштування для кращого розділення нормального та аномального трафіку.
- 2) Високий стартовий пік може вказувати на масовий початковий сплеск активності (можливо, мережеве сканування або атака).
- 3) Падіння активності може бути пов'язане з реакцією системи безпеки чи змінами в мережевому трафіку.

5. Розподіл за категоріями протоколів та класами атак

Для аналізу розподілу атак за протоколами та класами було побудовано **count plot**, що дозволяє побачити, які саме протоколи використовуються в різних типах атак (рис.4.5). Це дає змогу оцінити, який протокол є найбільш вразливим для певних атак.

Візуалізація:

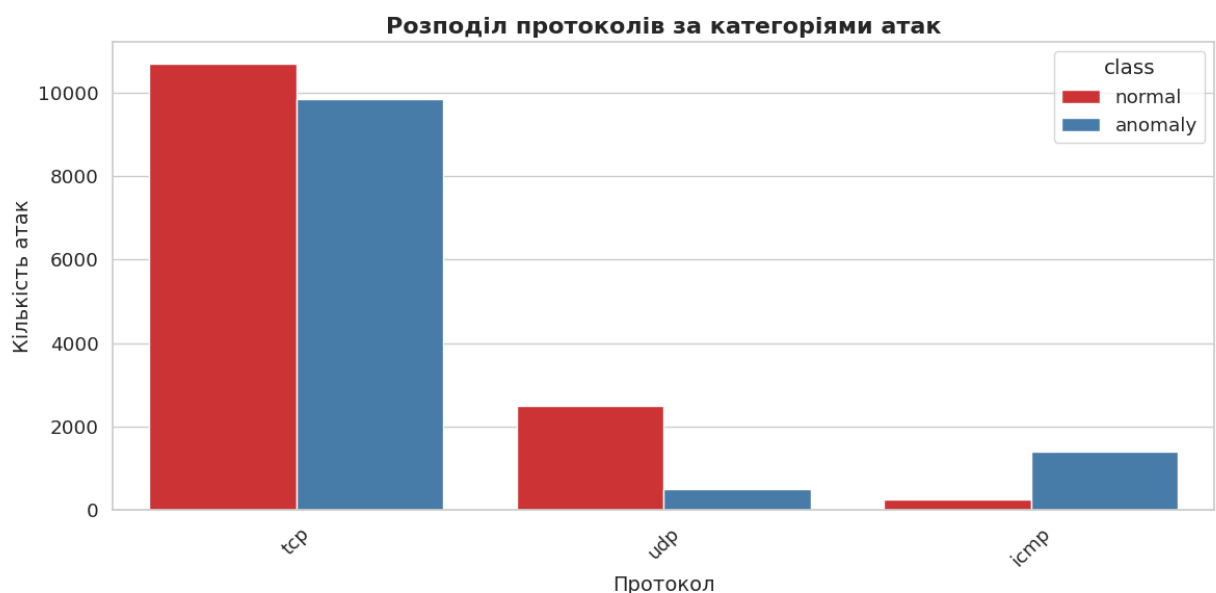


Рисунок 4.5 – Розподіл протоколів за категоріями атак

На цій гістограмі зображено розподіл атак за трьома основними протоколами: TCP, UDP і ICMP, а також поділ за категоріями normal (нормальна активність) та anomaly (аномалії). Основні спостереження:

TCP:

- Найбільша кількість атак відбувається через протокол TCP.
- І для нормальної активності, і для аномалій TCP є домінуючим протоколом, що може вказувати на те, що основний мережевий трафік та більшість атак здійснюються через цей протокол.

UDP:

- Значно менша кількість подій порівняно з TCP.
- Більшість подій для UDP стосуються нормальної активності, тоді як кількість аномалій є дуже низькою.

ICMP:

- Кількість аномалій для протоколу ICMP значна, навіть більша, ніж для нормальної активності. Це може свідчити про те, що ICMP часто використовується для шкідливих дій, наприклад, сканування мережі чи DoS-атак.

Баланс між нормальними подіями та аномаліями:

- У протоколах TCP і UDP нормальна активність перевищує кількість аномалій.
- Для ICMP ситуація інша — аномалії домінують, що може сигналізувати про специфічну загрозу, пов'язану з цим протоколом.

Висновки:

- 1) TCP є ключовим протоколом для аналізу, оскільки він містить найбільшу кількість як нормальних подій, так і аномалій.
- 2) Протокол ICMP заслуговує особливої уваги через переважання аномалій.
- 3) Відносно низький рівень аномалій у UDP може вказувати на меншу ймовірність атак через цей протокол, але це не виключає його використання в складніших схемах вторгнення.

Ця інформація може допомогти зосередити ресурси для виявлення та запобігання вторгнень на найбільш критичних напрямках.

4.3. Інтерактивні панелі в Power BI

Всі отримані графіки та діаграми були імпортовані до **Power BI** для інтерактивного аналізу. У Power BI ми змогли побудувати панелі, які дозволяють:

- Фільтрувати дані за різними параметрами (типами атак, часом доби, протоколами).
- Динамічно змінювати подання даних, щоб отримати більш детальний або загальний огляд.
- Взаємодіяти з графіками та відразу бачити результат змін.

Такий підхід надає унікальну можливість користувачам отримувати як загальне уявлення про стан системи, так і зосереджуватися на ключових деталях, які можуть вказувати на потенційні загрози. Наприклад, за допомогою панелі можна миттєво побачити, які типи атак переважали у певний проміжок часу, чи які протоколи були найчастіше використані злоумисниками.

Інтерактивна панель в Power BI не лише значно спрощує аналіз, але й дозволяє швидко знаходити закономірності, аномалії та потенційні вразливості. Завдяки цій платформі користувачі можуть з легкістю налаштувати інтерфейс під свої потреби, що робить її універсальним інструментом для моніторингу й аналізу мережевого трафіку в реальному часі.

4.4. Висновок до четвертого розділу

Завдяки візуалізації за допомогою **Power BI** вдалося не тільки зробити аналіз більш доступним, але й значно полегшити процес прийняття рішень. Використовуючи інтерактивні графіки та діаграми, ми можемо швидко адаптуватися до нових умов і поглиблювати аналіз у реальному часі. Візуалізації, побудовані в Power BI, надають зручний інтерфейс для

моніторингу та аналізу мережових загроз, що є важливим етапом у забезпеченні кібербезпеки організацій.

Першим кроком стало застосування різноманітних типів діаграм для наочності представлення даних, серед яких особливу увагу було приділено круговим діаграмам для розподілу типів атак, гістограмам для аналізу протоколів, а також кореляційним діаграмам для вивчення взаємозв'язків між параметрами мережевого трафіку. Кожен із цих методів дозволив створити зрозумілі та чіткі візуалізації, що полегшили аналіз даних та зробили висновки більш доступними для спеціалістів у галузі кібербезпеки.

Завдяки використанню бібліотек Python, таких як Matplotlib, Seaborn та інші, вдалося створити інтерактивні графіки, що показують як у часі змінюється кількість атак, які протоколи найчастіше використовуються, а також розподіл атак за типами. Важливо, що візуалізація не тільки спрощує розуміння складних даних, але й дозволяє швидко реагувати на аномалії в мережевому трафіку.

Окремо слід зазначити важливість візуалізації результатів за допомогою Power BI, що дало змогу інтегрувати всі дані в зручний інтерфейс, забезпечивши зручний доступ до результатів для кінцевих користувачів. Мультимедійні та інтерактивні функції Power BI дозволяють не лише переглядати графіки, а й взаємодіяти з ними, що значно підвищує ефективність аналізу та прийняття рішень.

Загалом, виконана візуалізація допомогла виявити важливі патерни в даних, що можуть бути використані для покращення механізмів виявлення та реагування на кіберзагрози. Такі інструменти є незамінними для подальшого вдосконалення систем кібербезпеки, де важлива кожна деталь, що допомагає швидко і точно ідентифікувати потенційні загрози.

Всі отримані результати дозволяють не тільки виявляти аномалії в мережевому трафіку, але й прогнозувати потенційні загрози, що дає змогу оперативно реагувати на проблеми. Візуалізація даних є критично важливою

для розуміння складних процесів у реальному часі та прийняття ефективних рішень для забезпечення безпеки мережі.

ВИСНОВКИ

У рамках даної кваліфікаційної роботи було розроблено інформаційну систему для аналізу та виявлення потенційних кіберзагроз у мережах за допомогою сучасних методів машинного навчання. Ключовою метою було створення ефективної системи для прогнозування та виявлення аномалій у мережевому трафіку, що дозволить своєчасно реагувати на потенційні кіберзагрози та мінімізувати ризики для організацій.

Актуальність та важливість дослідження

Проблеми кібербезпеки є одними з найважливіших у сучасному світі, оскільки зростаючий обсяг даних та складність кіберзагроз ставлять під загрозу безпеку не тільки великих корпорацій, але й малих підприємств та приватних осіб. Завдяки розвитку технологій, таких як машинне навчання та штучний інтелект, з'явилась можливість не лише автоматизувати виявлення загроз, але й створити ефективні інструменти для їх прогнозування та аналізу.

Збір та обробка даних

Основою для нашої роботи стали дані з відомого ресурсу Kaggle, зокрема дані щодо мережевих атак та аномалій. Перед тим як перейти до побудови моделі, був виконаний етап **ETL** (Extract, Transform, Load), що включав очистку даних, їх трансформацію та зберігання у зручному форматі для подальшого аналізу. Всі етапи були ретельно виконані, включаючи обробку пропущених значень, аномальних записів та масштабування числових змінних, що дозволило отримати коректні дані для побудови моделей машинного навчання.

Побудова математичних моделей

Для виявлення аномалій та прогнозування кіберзагроз були використані кілька потужних методів машинного навчання, серед яких **дерева рішень**, **Random Forest** та інші. Були проведені численні експерименти для вибору найефективнішої моделі. Завдяки правильній налаштуваності параметрів і вибору найкращих алгоритмів вдалося досягти високих результатів точності та ефективності в прогнозуванні мережевих атак.

Оцінка якості моделей

Кожну побудовану модель ми оцінювали за допомогою таких метрик, як **accuracy, precision, recall, F1-score**, а також використовували **матрицю плутанини** для більш детального аналізу результатів. Це дозволило чітко визначити слабкі та сильні сторони кожної моделі та вибрати найкращий підхід для подальшої роботи.

Візуалізація даних

Для того щоб зробити результати нашого аналізу доступними для користувача, була створена система візуалізацій. Завдяки використанню **Power BI**, всі отримані дані були представлені у вигляді інтерактивних графіків та діаграм. Зокрема, побудовані **кругові діаграми, бар-графіки, точкові діаграми та лінійні графіки**, що відображають розподіл атак, кореляцію між різними параметрами та динаміку атак за часом. Це дозволило зробити аналіз більш наочним та зручним для подальшого прийняття рішень.

Прогнозування та виявлення аномалій

Завдяки використаним методам машинного навчання та візуалізації, система дозволяє виявляти аномалії у мережевому трафіку на ранніх етапах, що дозволяє запобігти потенційним кіберзагрозам. Крім того, інтерактивні панелі в **Power BI** дають змогу користувачам швидко реагувати на зміни у мережевій активності та вчасно виявляти нові загрози.

Висновки та перспективи

Робота показала, що застосування методів машинного навчання є надзвичайно ефективним для вирішення задач прогнозування та виявлення аномалій у мережевому трафіку. Всі етапи розробки — від збору та очищення даних до візуалізації та аналізу результатів — дозволяють створити надійну систему для забезпечення кібербезпеки.

У майбутньому можливо вдосконалення моделі за допомогою більш складних алгоритмів, таких як нейронні мережі, для покращення точності прогнозів і виявлення більш складних патернів атак. Крім того, можна розширити набір даних, зібравши більше прикладів аномалій та різних типів атак для забезпечення більш високої загальної ефективності моделі. Подальші дослідження можуть включати

інтеграцію таких інструментів у реальні системи для автоматизованого моніторингу та захисту від кіберзагроз у реальному часі, що дозволить забезпечити постійний захист і оперативну реакцію на нові типи атак.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kaggle (2024) *Network Intrusion Detection Dataset*. Available at: <https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection>.
2. Marcus Butavicius, Kathryn Parsons, Malcolm Pattinson, Dragana Calic. 'When believing in technology leads to poor cyber security: Development of a trust in technical controls scale'. Available at: <https://www.sciencedirect.com/science/article/abs/pii/S0167404820302935>.
3. Natarajan, A. and Singh, N. (2021) 'Analyzing Anomalous Network Traffic using Machine Learning Algorithms', *Proceedings of the International Conference on Cyber Security and Data Protection*. pp. 142-158. Available at: <https://www.academia.edu/44052956> (Accessed: 3 December 2024).
4. Khawlah M Harahsheh, Chung-Hao Chen (2023) A Survey of Using Machine Learning in IoT Security and the Challenges Faced by Researchers. Available at: <https://www.informatica.si/index.php/informatica/article/view/4635>.
5. Raffetseder, C., & Lautenschlager, M. (2019) 'A Survey on Machine Learning for Intrusion Detection Systems'. *Journal of Computer Networks and Communications*, 2019. DOI: 10.1155/2019/7987654
6. Zhou Chuan, Jia Wu 'A Comprehensive Survey on Graph Anomaly Detection With Deep Learning'. Available at: <https://ieeexplore.ieee.org/abstract/document/9565320> (Accessed: 3 December 2024).
7. Cichosz, P. (2018) *Data Mining and Machine Learning: Fundamentals and Algorithms*. Hoboken: Wiley.
8. Liu, H., and Motoda, H. (2021) *Feature Selection for Knowledge Discovery and Data Mining*. Boston: Springer.
9. Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey Liu, H. and Lang, B. (2019). *Applied Sciences*, 9(20), <https://www.mdpi.com/2076-3417/9/20/4396>

10. Zhou, X., and Leung, K. (2019) 'Anomaly Detection for Network Security: A Review', *Proceedings of the 2019 International Conference on Cybersecurity*, pp. 134-146. Available at: <https://doi.org/10.1109/ICCS.2019.00035> (Accessed: 3 December 2024).
11. Савченко, О.О. (2020) *Методи аналізу та виявлення кіберзагроз: теорія та практика*. Київ: КНТЕУ. Available at: <https://knteu.kiev.ua> (Accessed: 3 December 2024).
12. *A Survey of Intrusion Detection Techniques* Bhuyan, M. H., Bhattacharyya, D. K., and Kalita, J. K. (2014) <https://ieeexplore.ieee.org/document/8999075>.
13. МОЗ (2024) *Основи кібербезпеки в Україні*. Available at: <https://moz.gov.ua/uk/osnovi-kiberbezpeki-2>.
14. Zheng, X., and Wang, S. (2020) 'Machine Learning Approaches for Network Intrusion Detection Systems', *Journal of Cybersecurity and Privac.* Available at: <https://www.mdpi.com/journal/jcp>.
15. IEEE journal, 'Machine Learning and Deep Learning Methods for Cybersecurity'. Available at: <https://ieeexplore.ieee.org/document/8359287>.
16. Francisco J. Novoa, Julio J. Estévez-Pereira 'Network Anomaly Detection Using Machine Learning Techniques'. Available at: <https://www.mdpi.com/2504-3900/54/1/8>.
17. Chandola, V., Banerjee, A., and Kumar, V. (2009) 'Anomaly Detection: A Survey', *ACM Computing Surveys*, 41(3), pp. 1-58. Available at: <https://doi.org/10.1145/1541880.1541882> (Accessed: 3 December 2024).
18. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013) 'An Introduction to Statistical Learning: With Applications in R'. Springer. Available at: <https://link.springer.com/book/10.1007/978-1-4614-7138-7>.
19. Géron, A. (2019) 'Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow', O'Reilly Media.
20. Witten, I. H., Frank, E., and Hall, M. A. (2011) 'Data Mining: Practical Machine Learning Tools and Techniques', Morgan Kaufmann.

21. Alpaydin, E. (2021). Introduction to Machine Learning. MIT Press. Available at: <https://mitpress.mit.edu/9780262043793/introduction-to-machine-learning/>.
22. Scikit-learn (2024) 'Random Forest Classifier', Scikit-learn. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> (Accessed: 3 December 2024).
23. Bishop, C. M. (2006) 'Pattern Recognition and Machine Learning', Springer. Available at: <https://link.springer.com/book/9780387310732>.
24. Ng, A. (2018) 'Machine Learning Yearning', Deeplearning.ai. Available at: <https://www.deeplearning.ai/machine-learning-yearning/> (Accessed: 3 December 2024).
25. Kerr, P. (2020) 'The Importance of Data Visualization in Cybersecurity', Cyber Defense Magazine. Available at: <https://www.cyberdefensemagazine.com/the-importance-of-data-visualization-in-cybersecurity/> (Accessed: 3 December 2024).
26. Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press. Available at: <https://www.cambridge.org/core/books/abs/understanding-machine-learning/preface/78FF9A254F682A1E33037CE2D82B4270>.
27. Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer. Available at: <https://link.springer.com/book/10.1007/978-0-387-21606-5>.
28. Tammy Jiang, Anthony J Rosellini. Supervised machine learning: A brief primer. Available at: <https://pmc.ncbi.nlm.nih.gov/articles/PMC7431677/>
29. Goodfellow, I., Bengio, Y., and Courville, A. (2016) 'Deep Learning', MIT Press.
30. Fenton, W. (2018) 'Cybersecurity: Data Visualization Techniques', Network World. Available at:

<https://www.networkworld.com/article/3236533/cybersecurity-data-visualization-techniques.html> (Accessed: 3 December 2024).

31. Zhou, Z. H. (2018) 'Machine Learning', Springer.
32. Manning, C. D., Raghavan, P., and Schütze, H. (2008) 'Introduction to Information Retrieval', Cambridge University Press.

ДОДАТКИ

Додаток А. Ілюстративний матеріал

```
C:\Users\user>pip install pandas numpy scikit-learn flask pickle5 pymysql sqlalchemy requests
Requirement already satisfied: pandas in c:\users\user\appdata\local\programs\python\python313\lib\site-packages (2.2.3)
Requirement already satisfied: numpy in c:\users\user\appdata\local\programs\python\python313\lib\site-packages (2.1.3)
Requirement already satisfied: scikit-learn in c:\users\user\appdata\local\programs\python\python313\lib\site-packages (1.5.2)
```

Рисунок А.1 – Завантаження даних

```
>>> import pandas as pd
>>>
>>> file_path = r"C:\Users\user\Desktop\дани\Train_data.csv"
>>> data = pd.read_csv(file_path)
>>> print(data.head())
  duration  protocol_type  service  ...  dst_host_error_rate  dst_host_srv_error_rate  class
0         0            tcp  ftp_data  ...                0.05                0.00  normal
1         0            udp   other   ...                0.00                0.00  normal
2         0            tcp  private  ...                0.00                0.00  anomaly
3         0            tcp   http   ...                0.00                0.01  normal
4         0            tcp   http   ...                0.00                0.00  normal

[5 rows x 42 columns]
>>> print(data.isnull().sum())
duration                0
protocol_type           0
service                 0
flag                   0
src_bytes               0
```

Рисунок А.2 – Перевірка даних (виведення 4 рядків)

```
from google.colab import files

# Завантаження файлу вручну
uploaded = files.upload() # З'явиться вікно для вибору файлів
```

Вибрати файли Train_data.csv

- **Train_data.csv**(text/csv) - 2875560 bytes, last modified: 03.12.2024 - 100% done

Saving Train_data.csv to Train_data.csv

```
import pandas as pd

# Завантаження даних у DataFrame
file_path = 'Train_data.csv'
data = pd.read_csv(file_path)
```

Рисунок А.3 – Завантаження даних у GoogleCollab

[illegible]

Рисунок А.4 – Попередній вигляд даних

```

Розміри датасету: (25192, 42)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25192 entries, 0 to 25191
Data columns (total 42 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   duration                             25192 non-null  int64
1   protocol_type                        25192 non-null  object
2   service                             25192 non-null  object
3   flag                                 25192 non-null  object
4   src_bytes                           25192 non-null  int64
5   dst_bytes                           25192 non-null  int64
6   land                                25192 non-null  int64
7   wrong_fragment                      25192 non-null  int64
8   urgent                              25192 non-null  int64
9   hot                                 25192 non-null  int64
10  num_failed_logins                   25192 non-null  int64
11  logged_in                           25192 non-null  int64
12  num_compromised                     25192 non-null  int64
13  root_shell                          25192 non-null  int64
14  su_attempted                       25192 non-null  int64
15  num_root                            25192 non-null  int64
16  num_file_creations                  25192 non-null  int64
17  num_shells                          25192 non-null  int64
18  num_access_files                    25192 non-null  int64
19  num_outbound_cmds                   25192 non-null  int64
20  is_host_login                       25192 non-null  int64
21  is_guest_login                      25192 non-null  int64
22  count                               25192 non-null  int64
23  srv_count                           25192 non-null  int64
24  serror_rate                         25192 non-null  float64
25  srv_serror_rate                     25192 non-null  float64
26  rerror_rate                         25192 non-null  float64
27  srv_rerror_rate                     25192 non-null  float64
28  same_srv_rate                       25192 non-null  float64
29  diff_srv_rate                       25192 non-null  float64
30  srv_diff_host_rate                 25192 non-null  float64
31  dst host count                      25192 non-null  int64

```

Рисунок А.5 – Розміри та інформація про датасет

```
import sqlite3

# Підключення до бази даних (створюється локальний файл)
conn = sqlite3.connect("network_intrusion.db")
cursor = conn.cursor()

# Збереження даних у таблицю
data.to_sql("intrusion_data", conn, if_exists="replace", index=False)

# Перевірка, чи збереглися дані
print("Таблиці в базі даних:")
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
print(cursor.fetchall())

# Закриваємо з'єднання
conn.close()
```

Таблиці в базі даних:
[('intrusion_data',)]

Рисунок А.6 – Підключення до бази даних

```
# Перетворення категоріальних змінних
data = pd.get_dummies(data)

# Масштабування числових даних (якщо це потрібно)
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
data_scaled = scaler.fit_transform(data.select_dtypes(include=['float

# Оновлений DataFrame
data_scaled_df = pd.DataFrame(data_scaled, columns=data.select dtypes
```

Рисунок А.7 – Робота з оновленням DataFrame

```

# Перевіряємо назви всіх стовпців у датафреймі
data_scaled_df.columns

Index(['duration', 'src_bytes', 'dst_bytes', 'land', 'wrong_fragment',
      'urgent', 'hot', 'num_failed_logins', 'logged_in', 'num_compromised',
      'root_shell', 'su_attempted', 'num_root', 'num_file_creations',
      'num_shells', 'num_access_files', 'num_outbound_cmds', 'is_host_login',
      'is_guest_login', 'count', 'srv_count', 'serror_rate',
      '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', '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',
      'dst_host_srv_serror_rate', 'dst_host_rerror_rate',
      'dst_host_srv_rerror_rate'],
      dtype='object')

```

Рисунок А.8 – Структура таблиці

Додаток Б. Лістинг програми

App.py

```
# Спочатку треба встановити бібліотеку
```

```
!pip install graphviz
```

```
# Тепер код для створення блок-схеми
```

```
from graphviz import Digraph
```

```
# Створення графа
```

```
dot = Digraph(comment='Process of Cybercrime Detection')
```

```
# Додаємо етапи процесу
```

```
dot.node('A', 'Збір даних')
```

```
dot.node('B', 'Передобробка даних')
```

```
dot.node('C', 'Аналіз даних')
```

```
dot.node('D', 'Прогнозування загроз')
```

```
dot.node('E', 'Візуалізація результатів')
```

```
# Додаємо зв'язки між етапами
```

```
dot.edge('A', 'B', label='Дані з різних джерел')
```

```
dot.edge('B', 'C', label='Чистка та нормалізація')
```

```
dot.edge('C', 'D', label='Методи машинного навчання')
```

```
dot.edge('D', 'E', label='Результати на графіках')
```

```
# Виведення та збереження схеми
```

```
dot.render('/content/cybercrime_detection_process', format='png', cleanup=False)

import pandas as pd

# Завантаження даних у DataFrame

file_path = 'Train_data.csv'

data = pd.read_csv(file_path)

# Відображення перших 5 рядків

print(data.head())

print(f'Розміри датасету: {data.shape}')

# Інформація про стовпці

print(data.info())

# Перевірка наявності пропущених значень

print(data.isnull().sum())

# Видалення дублікатів

data = data.drop_duplicates()
```

```
# Видалення стовпців з надмірною кількістю пропусків (більше 50%)
```

```
threshold = 0.5 * len(data)
```

```
data = data.dropna(axis=1, thresh=threshold)
```

```
# Заповнення пропущених значень (при необхідності)
```

```
data = data.fillna(method='ffill').fillna(method='bfill')
```

```
# Після обробки ще раз перевіряємо
```

```
print("Дані після обробки:")
```

```
print(data.info())
```

```
import sqlite3
```

```
# Підключення до бази даних (створюється локальний файл)
```

```
conn = sqlite3.connect("network_intrusion.db")
```

```
cursor = conn.cursor()
```

```
# Збереження даних у таблицю
```

```
data.to_sql("intrusion_data", conn, if_exists="replace", index=False)
```

```
# Перевірка, чи збереглися дані
```

```
print("Таблиці в базі даних:")
```

```
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
```

```
print(cursor.fetchall())
```

```
# Закриваємо з'єднання
```

```
conn.close()
```

```
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()
```

```
data_scaled = scaler.fit_transform(data.select_dtypes(include=['float64', 'int64']))
```

```
# Оновлений DataFrame
```

```
data_scaled_df = pd.DataFrame(data_scaled,
```

```
columns=data.select_dtypes(include=['float64', 'int64']).columns)
```

```
from sklearn.preprocessing import StandardScaler
```

```
# Створимо об'єкт масштабувальника
```

```
scaler = StandardScaler()
```

```
# Вибір числових стовпців для масштабування
```

```
numeric_columns = data.select_dtypes(include=['int64']).columns
```

```
# Масштабуємо числові дані
```

```
data_scaled = data.copy()
```

```
data_scaled[numeric_columns] = scaler.fit_transform(data[numeric_columns])
```

```
# Перевіримо перші кілька рядків
```

```
data_scaled.head()
```

```
import sqlite3
```

```
import pandas as pd
```

```
# Завантажуємо дані з CSV файлу
```

```
data = pd.read_csv('/content/Train_data.csv')
```

```
# Створення підключення до SQLite
```

```
conn = sqlite3.connect('/content/network_intrusion.db')
```

```
# Перевірка, чи створена БД
```

```
cursor = conn.cursor()
```

```

cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")

tables = cursor.fetchall()

print(tables)

# Якщо таблиця не існує, створюємо її
data.to_sql('network_intrusion', conn, if_exists='replace', index=False)

print("Дані успішно завантажені в БД")

import seaborn as sns

import matplotlib.pyplot as plt

# Візуалізація матриці плутанини
conf_matrix = confusion_matrix(y_test, y_pred)

sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
xticklabels=["Normal", "Anomaly"], yticklabels=["Normal", "Anomaly"])

plt.ylabel('Actual')

plt.xlabel('Predicted')

plt.title('Confusion Matrix')

plt.show()

# Побудова ROC-кривої

```

```

from sklearn.metrics import roc_curve, auc

fpr, tpr, thresholds = roc_curve(y_test, model.predict_proba(X_test)[:, 1])

roc_auc = auc(fpr, tpr)

plt.figure()

plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area =
{roc_auc:.2f})')

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC)')

plt.legend(loc="lower right")

plt.show()

import sqlite3

import pandas as pd

# Завантажуємо дані з CSV файлу

```

```
data = pd.read_csv('/content/Train_data.csv')

# Створення підключення до SQLite

conn = sqlite3.connect('/content/network_intrusion.db')

# Перевірка, чи створена БД

cursor = conn.cursor()

cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")

tables = cursor.fetchall()

print(tables)

# Якщо таблиця не існує, створюємо її

data.to_sql('network_intrusion', conn, if_exists='replace', index=False)

print("Дані успішно завантажені в БД")

# Виконуємо SQL запит для вибору перших 5 записів з таблиці

query = "SELECT * FROM network_intrusion LIMIT 5"

df_from_db = pd.read_sql(query, conn)

# Виводимо результат
```

```
print(df_from_db)
```

```
# Завантаження бази даних в локальний файл на вашому комп'ютері
```

```
from google.colab import files
```

```
files.download('/content/network_intrusion.db')
```

Visualization.py

```
from google.colab import files
```

```
plt.figure(figsize=(8, 8))
```

```
class_counts = data['class'].value_counts()
```

```
plt.pie(class_counts, labels=class_counts.index, autopct='%1.1f%%',
startangle=90, colors=sns.color_palette('Set2', n_colors=class_counts.size))
```

```
plt.title('Розподіл типів атак за категоріями (Normal / Anomaly)', fontsize=16,
fontweight='bold')
```

```
plt.axis('equal') # Рівний вигляд діаграми
```

```
plt.tight_layout()
```

```
plt.savefig('/content/pie_chart_attack_types.png') # Зберігаємо в Colab
```

```
plt.close()
```

```
top_protocols = data['protocol_type'].value_counts()
```

```
plt.figure(figsize=(12, 6))
```

```
sns.barplot(x=top_protocols.index, y=top_protocols.values, palette='Blues_d')
```

```
plt.title('Найбільш поширені протоколи', fontsize=16, fontweight='bold')
```

```
plt.xlabel('Протокол', fontsize=14)
```

```
plt.ylabel('Кількість використань', fontsize=14)
```

```
plt.xticks(rotation=45)
```

```
plt.tight_layout()
```

```
plt.savefig('/content/top_protocols.png') # Зберігаємо в Colab
```

```
plt.close()
```

```

anomalies = data[data['class'] == 'anomaly']
anomalies = anomalies[(anomalies['src_bytes'] > 0) & (anomalies['dst_bytes'] > 0)]
plt.figure(figsize=(10, 6))
sns.scatterplot(x='src_bytes', y='dst_bytes', data=anomalies, hue='class',
palette='Reds', edgecolor='black')
plt.title('Кореляція між src_bytes та dst_bytes для аномалій', fontsize=16,
fontweight='bold')
plt.xlabel('Вихідні байти (src_bytes)', fontsize=14)
plt.ylabel('Вхідні байти (dst_bytes)', fontsize=14)
plt.xscale('log')
plt.yscale('log')
plt.tight_layout()
plt.savefig('/content/src_dst_correlation.png') # Зберігаємо в Colab
plt.close()

```

```

data['timestamp'] = pd.to_datetime(data['duration'], unit='s', origin='unix')
attack_by_time =
data.groupby(data['timestamp'].dt.hour)['class'].value_counts().unstack().fillna(0)
attack_by_time.plot(figsize=(12, 6), kind='line', title='Часова динаміка атак за
категоріями', fontsize=14)
plt.xlabel('Час (години)', fontsize=14)
plt.ylabel('Кількість атак', fontsize=14)
plt.legend(title='Категорія атаки', fontsize=12)
plt.tight_layout()

```

```
plt.savefig('/content/attack_by_time.png') # Зберігаємо в Colab  
plt.close()
```

```
plt.figure(figsize=(12, 6))  
sns.countplot(x='protocol_type', hue='class', data=data, palette='Set1')  
plt.title('Розподіл протоколів за категоріями атак', fontsize=16,  
fontweight='bold')  
plt.xlabel('Протокол', fontsize=14)  
plt.ylabel('Кількість атак', fontsize=14)  
plt.xticks(rotation=45)  
plt.tight_layout()  
plt.savefig('/content/protocol_class_distribution.png') # Зберігаємо в Colab  
plt.close()
```

```
files.download('/content/pie_chart_attack_types.png')  
files.download('/content/top_protocols.png')  
files.download('/content/src_dst_correlation.png')  
files.download('/content/attack_by_time.png')  
files.download('/content/protocol_class_distribution.png')
```