Міністерство освіти і науки України
Луцький національний технічний університет
Факультет робототехніки та штучного інтелекту
Кафедра штучного інтелекту та математичного моделювання

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ
«БАКАЛАВР»

АНАЛІЗ СТАТИСТИЧНИХ ХАРАКТЕРИСТИК МЕРЕЖЕВОГО ТРАФІКУ ТА РОЗРОБКА МОДЕЛІ ВИЯВЛЕННЯ АНОМАЛЬНОЇ АКТИВНОСТІ

ANALYSIS OF STATISTICAL CHARACTERISTICS OF NETWORK TRAFFIC AND DEVELOPMENT OF A MODEL FOR DETECTING ANOMALOUS ACTIVITY

Спеціальність 113 Прикладна математика
(шифр і назва спеціальності)

освітня програма «Штучний інтелект та аналіз масивів даних»
(назва освітньої програми)

Виконав: здобувач вищої освіти
Групи ПРМ-41
Касько Юрій Миколайович
_____
(підпис)

Керівник:
к.т.н., доцент
Фурс Тетяна Василіна
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«___» _____ 20__ р.
к.т.н., доцент
Гарант освітньої програми:
Приходько Олексій Сергійович
_____
(підпис)

Луцьк – 2026 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *архітектури, будівництва та дизайну*
Кафедра *прикладної математики та механіки*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *11 Математика і статистика*
Спеціальність *113 Прикладна математика*
Освітня програма *Штучний інтелект та аналіз масивів даних*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____Мікуліч О.А.
«___» _____2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Касько Юрій Миколайович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Аналіз статистичних характеристик мережевого трафіку та розробка моделі виявлення аномальної активності / Analysis of statistical characteristics of network traffic and development of a model for detecting anomalous activity
Керівник роботи: *Фурс Тетяна Василівна*
затверджені наказом закладу вищої освіти від «31» грудня 2025 р. № 557/01-02

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи 04.06.2026 р.

3. Вихідні дані до роботи *профільні публікації з штучного інтелекту та математичного моделювання в межах досліджуваної проблематики; релевантні набори даних; математичні моделі цільових процесів; технічна документація Python-бібліотек та методичні вказівки до виконання кваліфікаційної роботи бакалавра*

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити):
Аналіз предметної області
Постановка задачі та вибір методів
Практична реалізація
Отримання та аналіз результатів

5. Перелік графічного (ілюстративного) матеріалу:
Презентація роботи (слайди): об'єкт, предмет, мета та завдання дослідження; аналіз предметної області; математичні та алгоритмічні моделі, результати експериментальних досліджень (метрики та візуалізація роботи алгоритму); загальні висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
1 розділ	*Фурс Т.В., доцент кафедри*		
2 розділ	*Фурс Т.В., доцент кафедри*		
3 розділ	*Фурс Т.В., доцент кафедри*		
4 розділ	*Фурс Т.В., доцент кафедри*		
Висновки	*Фурс Т.В., доцент кафедри*		

7. Дата видачі завдання «___» _____ 202__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1.	*Огляд літератури із досліджуваної проблеми*	*до 01.03.2026*	
2.	*Перший розділ*	*до 5.03.2026*	
3.	*Другий розділ*	*до 01.04.2026*	
4.	*Третій розділ*	*до 10.04.2026*	
5.	*Четвертий розділ*	*до 20.04.2026*	
6.	*Висновки*	*до 28.04.2026*	
7.	*Формування списку використаних джерел*	*до 05.05.2026*	
8.	*Оформлення ілюстративного матеріалу*	*до 09.05.2026*	
9.	*Нормоконтроль*	*до 20.05.2026*	
10.	*Інструментальна перевірка на академічний плагіат*	*до 02.062026*	*Показник запозичень тексту ____ %*
11.	*Представлення кваліфікаційної роботи бакалавра до захисту*	*до 04.06.2026*	

Здобувач вищої освіти _____ (Касько Ю. М.)
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (Фурс Т. В.)
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Касько Ю. М. Аналіз статистичних характеристик мережевого трафіку та розробка моделі виявлення аномальної активності. Рукопис.

Кваліфікаційна робота бакалавра ОП «Штучний інтелект та аналіз масивів даних» спеціальності 113 Прикладна математика. Луцький національний технічний університет. Луцьк, 2026.

Кваліфікаційну роботу присвячено розв’язанню актуальної задачі забезпечення мережевої безпеки шляхом застосування методів машинного навчання для автоматизованого виявлення аномальної активності (кібератак). В ході дослідження здійснено детальний аналіз предметної області та обґрунтовано використання відкритого набору даних NSL-KDD, що містить 41 статистичну ознаку мережевих з'єднань.

У роботі розроблено та програмно реалізовано мовою Python дві класифікаційні моделі на базі алгоритмів Дерева рішень (Decision Tree) та Випадкового лісу (Random Forest). Експериментальне дослідження виявило, що класичне Дерево рішень продемонструвало вищу загальну точність (80.20%) порівняно з ансамблевою моделлю (77.31%) на тестовій вибірці з невідомими типами загроз. Обидві моделі показали високу влучність (Precision) понад 96% при ідентифікації аномалій, що забезпечує низький рівень хибних спрацьовувань. Розроблене програмне рішення може бути інтегроване в сучасні системи виявлення вторгнень.

Ключові слова: *машинне навчання, мережевий трафік, кібербезпека, виявлення аномалій, Decision Tree, Random Forest, NSL-KDD.*.

ABSTRACT

Kasko Iu. M. Analysis of statistical characteristics of network traffic and development of a model for detecting anomalous activity. Manuscript.

Bachelor's Thesis in the field of "Artificial Intelligence and BigData Analysis" in specialty 113 Applied Mathematics. Lutsk National Technical University. Lutsk, 2026.

The bachelor's thesis is devoted to solving the urgent problem of network security by applying machine learning methods for the automated detection of anomalous activity (cyberattacks). During the research, a detailed analysis of the subject area was carried out, and the use of the NSL-KDD dataset, which contains 41 statistical features of network connections, was justified.

Two classification models based on Decision Tree and Random Forest algorithms were developed and implemented in Python. The experimental study revealed that the classic Decision Tree demonstrated higher overall accuracy (80.20%) compared to the ensemble model (77.31%) on a test set with unknown attack types. Both models showed a high precision of over 96% in identifying anomalies, which ensures a low rate of false positives. The developed software solution can be integrated into modern Intrusion Detection Systems (IDS).

Keywords: machine learning, network traffic, cybersecurity, anomaly detection, Decision Tree, Random Forest, NSL-KDD.

ЗМІСТ

ВСТУП

Актуальність теми. В умовах стрімкого розвитку інформаційних технологій та розширення комп'ютерних мереж проблема забезпечення кібербезпеки набуває критичного значення. Зростання обсягів переданих даних супроводжується збільшенням кількості та складності кібератак. Традиційні системи виявлення вторгнень (Intrusion Detection Systems, IDS), які базуються на сигнатурному аналізі, виявляють низьку ефективність проти нових, раніше невідомих загроз (атак нульового дня). У зв'язку з цим, актуальним є застосування методів машинного навчання та аналізу масивів даних для дослідження статистичних характеристик мережевого трафіку з метою автоматизованого виявлення аномальної активності. Використання інтелектуальних моделей дозволяє виявляти приховані закономірності в даних та підвищувати загальну стійкість мережевої інфраструктури.

Мета дослідження: розробка та програмна імплементація моделі машинного навчання для виявлення аномальної активності на основі аналізу статистичних характеристик мережевого трафіку.

Для досягнення поставленої мети було сформульовано наступні **завдання**:

1. Провести аналіз предметної області, дослідити існуючі підходи до виявлення мережевих аномалій та обрати набір даних для проведення експериментів.
2. Сформулювати математичну постановку задачі та обґрунтувати вибір методів машинного навчання.
3. Здійснити попередню обробку даних, виділення інформативних ознак та розробити програмне рішення для класифікації трафіку.
4. Провести експериментальне дослідження розроблених моделей, порівняти їхню ефективність за допомогою метрик якості та проаналізувати отримані результати.

Об’єкт дослідження: процес виявлення аномальної активності в комп'ютерних мережах.

Предмет дослідження: методи машинного навчання та статистичні характеристики мережевого трафіку, що використовуються для класифікації аномалій.

Методи дослідження. Для розв'язання поставлених завдань у роботі використано методи статистичного аналізу даних (для попередньої обробки та відбору ознак), методи машинного навчання з учителем, зокрема ансамблеві алгоритми (випадковий ліс) для побудови класифікаційних моделей, а також методи емпіричної оцінки для порівняння ефективності алгоритмів (матриця помилок, метрики точності, повноти та F1-міри).

Інформаційна база дослідження. Дослідження проводилося на базі відкритого еталонного набору даних NSL-KDD, що містить детальні статистичні характеристики нормального та аномального мережевого трафіку.

У процесі підготовки бакалаврської кваліфікаційної роботи застосовувалися технології штучного інтелекту як допоміжний інструментарій. Зокрема, для стилістичної правки та структурування тексту використано ChatGPT-5 та Gemini 3.3. Автор несе повну відповідальність за зміст роботи.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ДАНИХ

1.1. Загальна характеристика мережевого трафіку та проблеми безпеки

Сучасні інформаційно-комунікаційні системи характеризуються експоненційним зростанням обсягів переданих даних та ускладненням архітектури комп’ютерних мереж. Мережевий трафік, як сукупність пакетів даних, що передаються між вузлами мережі, є основним об'єктом моніторингу в системах забезпечення інформаційної безпеки. Забезпечення захищеності мережевої інфраструктури базується на тріаді інформаційної безпеки: конфіденційності, цілісності та доступності ресурсів [1].

Порушення будь-якого з цих принципів розглядається як кіберінцидент або мережева аномалія. Під аномалією в контексті мережевого трафіку розуміють будь-яке відхилення характеристик потоку даних від встановленого профілю нормальної поведінки системи. Такі відхилення найчастіше є наслідком цілеспрямованих шкідливих дій (кібератак), апаратних збоїв або помилок конфігурації.

З точки зору статистичних характеристик мережевого трафіку, всі кібератаки можна класифікувати на чотири основні категорії [2]:

1. DoS (Denial of Service) – атаки типу «відмова в обслуговуванні». Характеризуються генеруванням надмірної кількості запитів до цільового вузла з метою вичерпання його обчислювальних ресурсів або пропускної здатності каналу зв'язку. На статистичному рівні проявляються як різке зростання кількості пакетів та з'єднань за одиницю часу.
2. Probe (Зондування та сканування) – діяльність, спрямована на збір інформації про структуру цільової мережі, відкриті порти, типи операційних систем та наявні вразливості. Аномалії цього типу

виявляються через нетипові комбінації IP-адрес та портів призначення, а також високу частку з'єднань з помилками маршрутизації.

3. R2L (Remote to Local) – спроби несанкціонованого отримання доступу до локального комп'ютера з віддаленого вузла (наприклад, підбір паролів по SSH чи FTP). Статистично такі атаки виділяються тривалим часом з'єднання та великою кількістю невдалих спроб авторизації.

4. U2R (User to Root) – атаки, спрямовані на підвищення локальних привілеїв звичайного користувача до рівня адміністратора (root). Виявлення таких аномалій є найскладнішим, оскільки обсяг трафіку при них мінімальний, а основні зміни відбуваються у вмісті пакетів (payload).

Динамічна природа мережевого трафіку унеможливлює створення статичних правил для виявлення всіх можливих типів аномалій, що зумовлює необхідність переходу до інтелектуальних систем аналізу [3].

1.2. Огляд існуючих підходів до виявлення аномальної активності в мережах

Системи виявлення вторгнень (Intrusion Detection Systems, IDS) є ключовим елементом ешелонованого захисту мереж. Традиційно підходи до виявлення мережевих аномалій поділяються на дві великі парадигми: сигнатурний аналіз (Misuse Detection) та аналіз на основі поведінкових аномалій (Anomaly Detection) [4].

Сигнатурний аналіз полягає у порівнянні поточного мережевого трафіку з базою даних відомих шаблонів кібератак (сигнатур). Цей підхід демонструє високу точність та мінімальний рівень хибних спрацьовувань (False Positives) при виявленні вже відомих загроз. Проте, його критичним недоліком є повна нездатність ідентифікувати атаки нульового дня (Zero-day attacks), а також модифіковані версії існуючих шкідливих програм.

Методи на основі виявлення аномалій, натомість, будують математичну або статистичну модель «нормальної» поведінки системи під час фази навчання. Будь-який трафік, що статистично значущо відхиляється від цієї базової моделі, класифікується як аномальний. Хоча цей підхід дозволяє виявляти нові типи атак, він схильний до високого рівня хибнопозитивних спрацьовувань, оскільки легітимна поведінка користувачів у мережі також може змінюватися з часом [5].

У сучасних дослідженнях найбільшої популярності набуло застосування алгоритмів машинного навчання (Machine Learning, ML) для автоматизації процесу виявлення аномалій. Використання методів навчання з учителем (наприклад, дерев рішень, випадкових лісів, методу опорних векторів та штучних нейронних мереж) дозволяє на основі історичних даних автоматично формувати складні нелінійні правила розмежування нормального та аномального трафіку [6].

1.3. Аналіз відкритих наборів даних для дослідження мережевих аномалій

Для розробки, навчання та об'єктивного оцінювання моделей машинного навчання необхідні репрезентативні набори даних (датасети), що містять реалістичний мережевий трафік із попередньо розміченими класами (норма/атака).

Одним із перших стандартів у цій галузі був набір KDD Cup 1999, створений на базі перехопленого трафіку мереж ВПС США (DARPA). Проте, подальші дослідження виявили суттєві недоліки цього датасету, зокрема наявність величезної кількості дубльованих записів (близько 78% у тренувальній вибірці та 75% у тестовій). Це призводило до зміщення оцінок моделей у бік частих типів атак та штучного завищення показників точності [7].

Для вирішення цих проблем було розроблено покращений набір даних NSL-KDD. У рамках даного кваліфікаційного дослідження обрано саме цей датасет з наступних причин:

1. Відсутність надлишкових записів: видалення дублікатів з NSL-KDD унеможливлює перенавчання (overfitting) моделей на одних і тих самих прикладах.
2. Об'єктивність тестування: тестова вибірка містить типи атак, які не зустрічалися в тренувальному наборі, що дозволяє адекватно оцінити здатність алгоритмів до узагальнення.
3. Оптимальний обсяг: кількість записів є достатньою для навчання моделей, але не потребує залучення надмірних обчислювальних потужностей.

Кожен запис у датасеті NSL-KDD представляє собою окреме мережеве з'єднання (сесію) і описується 41 ознакою (статистичним параметром), які можна розділити на три групи: базові ознаки TCP-з'єднання (наприклад, тривалість, протокол, кількість переданих байтів), контентні ознаки (наприклад, кількість невдалих логінів) та ознаки трафіку, обчислені за часовими вікнами (наприклад, відсоток з'єднань з однаковим сервісом).

Висновки до розділу 1

У першому розділі проведено аналіз предметної області та визначено проблему забезпечення безпеки мережевої інфраструктури. Встановлено, що класичні сигнатурні методи є неефективними проти нових загроз, що обґрунтовує необхідність застосування систем на основі виявлення поведінкових аномалій за допомогою машинного навчання. Здійснено огляд типів мережевих аномалій та обґрунтовано вибір еталонного набору даних NSL-KDD як інформаційної бази для подальших експериментальних досліджень. Цей датасет надає якісний набір із 41 статистичної ознаки трафіку, очищений від дублікатів, що дозволить побудувати об'єктивні класифікаційні моделі.

РОЗДІЛ 2
ПОСТАНОВКА ЗАДАЧІ ТА ВИБІР МЕТОДІВ РОЗВ'ЯЗАННЯ

2.1. Математична постановка задачі виявлення аномалій як задачі класифікації

З математичної точки зору, процес виявлення аномальної активності в мережевому трафіку зводиться до задачі бінарної класифікації з учителем (Supervised Learning).

Нехай задано набір даних (датасет), що містить історичну інформацію про мережеві з'єднання. Кожне з'єднання описується вектором статистичних ознак. Позначимо простір об'єктів як X, де кожен об'єкт $x_i \in X$ є вектором розмірності m (у випадку датасету NSL-KDD, m = 41). Простір відповідей позначимо як Y, де кожному об'єкту відповідає мітка класу. Для задачі бінарної класифікації простір відповідей складається з двох множин:

$$Y = \{0,1\} \tag{2.1}$$

де 0 – відповідає нормальному мережевому трафіку; 1 – відповідає аномальному трафіку (кібератаці).

Навчальна вибірка D задається як множина пар «об'єкт-відповідь»:

$$D = \{(x_1, y_1),(x_2, y_2),...,(x_n, y_n)\} \tag{2.2}$$

де n – загальна кількість мережевих з'єднань у навчальному наборі.

Головна мета машинного навчання полягає у знаходженні такої цільової функції (моделі) F(x), яка відображає простір ознак X у простір класів Y, забезпечуючи при цьому мінімальну кількість помилок на нових, невідомих даних [8]:

$$F : X \to Y \tag{2.3}$$

Для оптимізації моделі під час навчання мінімізується функція втрат (Loss Function) L(F(x), y), яка кількісно оцінює розбіжність між передбаченим значенням алгоритму та фактичною міткою класу з'єднання.

2.2. Обґрунтування вибору алгоритмів машинного навчання

Для розв'язання поставленої задачі класифікації мережевого трафіку було обрано два алгоритми на основі дерев рішень: класичне Дерево рішень (Decision Tree) та ансамблевий метод Випадковий ліс (Random Forest). Вибір цих алгоритмів обґрунтований їхньою високою інтерпретованістю, здатністю працювати з нелінійними залежностями в даних та стійкістю до масштабування ознак [9].

Дерево рішень – це алгоритм, який рекурсивно розбиває простір ознак на підмножини на основі певних правил (умов). У кожному вузлі дерева алгоритм обирає ту ознаку, яка забезпечує найкращий поділ даних за класами. Для оцінки якості поділу найчастіше використовується критерій домішки Джині (Gini Impurity). Для вузла v індекс Джині розраховується за формулою:

$$Gini(v) = 1 - \sum_{i=1}^{k} p_i^2 \tag{2.4}$$

де k – кількість класів (у нашому випадку 2); p_i – ймовірність того, що випадково обраний об'єкт із вузла v належить до класу i.

Чим менше значення Gini(v), тим більш «чистим» є вузол (тобто він містить об'єкти переважно одного класу). Модель дерева рішень формує чіткі правила типу «ЯКЩО (ознака > поріг) ТО (клас 1)», що є критично важливим для систем виявлення вторгнень, оскільки адміністратори мережі можуть легко інтерпретувати причину блокування трафіку.

Випадковий ліс (Random Forest) є ансамблевим методом, який будує велику кількість незалежних дерев рішень на випадкових підмножинах даних і усереднює їхні результати. Цей підхід значно знижує ризик перенавчання моделі та підвищує загальну точність класифікації. Фінальне рішення ансамблю для нового об'єкта x визначається шляхом мажоритарного голосування (більшістю голосів) усіх побудованих дерев:

$$H(x) = \arg\max_{y \in Y} \sum_{j=1}^{T} I(h_j(x) = y) \tag{2.5}$$

де T – загальна кількість дерев в ансамблі; $h_j(x)$ – передбачення j-го дерева для об'єкта x; $I(\cdot)$ – індикаторна функція, що дорівнює 1, якщо умова виконується, і 0 в іншому випадку.

2.3. Метрики оцінки ефективності моделей класифікації

Оскільки мережевий трафік часто є незбалансованим (нормальних з'єднань значно більше, ніж аномалій), використання лише однієї метрики загальної точності є недостатнім для об'єктивного оцінювання ефективності систем виявлення вторгнень [10].

Основою для розрахунку всіх класифікаційних метрик є матриця помилок, яка містить чотири базові показники:

– True Positives (TP) – кількість правильно виявлених аномалій;

– True Negatives (TN) – кількість правильно ідентифікованого нормального трафіку;

– False Positives (FP) – хибні спрацьовування (нормальний трафік помилково визнано атакою);

– False Negatives (FN) – пропущені загрози (атаку розпізнано як нормальний трафік).

На базі цих показників у роботі використовуються такі метрики:

1. Загальна точність (Accuracy) – частка правильних передбачень серед усіх з'єднань:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

2. Влучність або Точність (Precision) – показує, яка частка з'єднань, класифікованих як аномалії, дійсно є кібератаками (дозволяє оцінити рівень хибних тривог):

$$Precision = \frac{TP}{TP + FP} \tag{2.7}$$

3. Повнота (Recall) – демонструє здатність моделі виявляти всі наявні загрози (критично важлива метрика для систем безпеки):

$$Recall = \frac{TP}{TP + FN} \tag{2.8}$$

4. F1-міра (F1-Score) – гармонійне середнє між влучністю та повнотою, що дозволяє отримати комплексну оцінку якості моделі, особливо за умови дисбалансу класів:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.9}$$

Висновки до розділу 2

У другому розділі здійснено формалізацію проблеми виявлення мережевих аномалій шляхом зведення її до задачі бінарної класифікації з учителем. Обґрунтовано доцільність застосування алгоритму Дерева рішень та ансамблевого методу Випадкового лісу, які базуються на мінімізації критерію Джині та мажоритарному голосуванні відповідно. Дані методи забезпечують високу швидкість обробки, інтерпретованість результатів та стійкість до шуму. Для об'єктивної оцінки розроблених моделей обрано комплексний набір метрик, що базується на матриці помилок і включає Accuracy, Precision, Recall та F1-міру. Такий підхід дозволить всебічно проаналізувати ефективність алгоритмів в умовах реального мережевого трафіку.

РОЗДІЛ 3
РОЗРОБКА ТА ІМПЛЕМЕНТАЦІЯ РІШЕННЯ

3.1. Вибір програмних засобів та бібліотек для реалізації

Для практичної реалізації задачі виявлення аномальної активності в мережевому трафіку було обрано мову програмування Python. Вибір даного інструменту обґрунтовується його лідируючими позиціями у сфері штучного інтелекту, широкою екосистемою бібліотек для машинного навчання та зручністю роботи з великими масивами даних [11].

Розробка програмного рішення здійснювалася з використанням наступних спеціалізованих бібліотек:

– Pandas – потужний інструмент для структурного аналізу даних. Використовувався для завантаження наборів даних, маніпуляцій з таблицями (кадрами даних DataFrame), фільтрації та перетворення типів ознак;

– Scikit-learn (sklearn) – базова бібліотека машинного навчання на Python, що містить оптимізовані імплементації математичних алгоритмів класифікації (зокрема дерев рішень та випадкового лісу), інструменти для зменшення розмірності та обчислення метрик якості [12];

– Matplotlib та Seaborn – бібліотеки для побудови наукової графіки, гістограм, матриць помилок та візуалізації простору ознак.

Архітектура розробленої системи наведена на рис. 3.1.

Рисунок 3.1 – Загальна архітектура розробленої системи виявлення аномалій

3.2. Попередня обробка даних та виділення статистичних ознак

Перед подачею даних на вхід алгоритмів машинного навчання необхідно провести їх підготовку. Набір даних NSL-KDD постачається у вигляді текстових файлів з розділовими знаками (формат CSV). Кожен запис містить 41 ознаку та одну цільову змінну – мітку класу (назву конкретної атаки або маркер нормального трафіку). Приклад даних в датасеті наведено на рис. 3.2.

```
print(train.head());
```

```
   0    1         2   3    4     5   6   7   8   9  ...   32    33    34  \
0   0  tcp  ftp_data  SF  491     0   0   0   0   0  ...   25  0.17  0.03
1   0  udp     other  SF  146     0   0   0   0   0  ...    1  0.00  0.60
2   0  tcp   private  S0    0     0   0   0   0   0  ...   26  0.10  0.05
3   0  tcp      http  SF  232  8153   0   0   0   0  ...  255  1.00  0.00
4   0  tcp      http  SF  199   420   0   0   0   0  ...  255  1.00  0.00

     35    36    37    38    39    40       41
0  0.17  0.00  0.00  0.00  0.05  0.00   normal
1  0.88  0.00  0.00  0.00  0.00  0.00   normal
2  0.00  0.00  1.00  1.00  0.00  0.00  neptune
3  0.03  0.04  0.03  0.01  0.00  0.01   normal
4  0.00  0.00  0.00  0.00  0.00  0.00   normal
```

Рисунок 3.2 – Фрагмент вихідного набору даних мережевого трафіку

Першим етапом є завантаження тренувальної та тестової вибірок із відкиданням надлишкової метаінформації. Наступним критичним кроком є трансформація багатокласової цільової змінної у бінарну форму. Оскільки мета дослідження полягає у загальному виявленні аномалій, всі типи мережевих атак об'єднуються в один клас (1), а безпечний трафік маркується як норма (0). Фрагмент програмної реалізації даного етапу наведено у лістингу 3.1.

Лістинг 3.1 Трансформація цільової змінної для бінарної класифікації

```
# Завантаження даних та відокремлення цільової змінної від ознак
X_train = train.iloc[:, :-1]
# Формування вектора відповідей (0 - норма, 1 - аномалія)
y_train = (train.iloc[:, -1] != 'normal').astype(int)
```

```
X_test = test.iloc[:, :-1]
y_test = (test.iloc[:, -1] != 'normal').astype(int)
```

кінець лістингу 3.1

Особливістю набору даних NSL-KDD є наявність категоріальних ознак (наприклад, тип протоколу TCP/UDP/ICMP, назва сервісу HTTP/FTP та прапорці стану з'єднання). Більшість алгоритмів машинного навчання, включаючи обрані для дослідження, оперують виключно числовими матрицями [13]. Для вирішення цієї проблеми застосовано метод унітарного кодування (One-Hot Encoding).

Цей метод перетворює кожне унікальне значення категоріальної ознаки на окремий стовпець бінарних значень (0 або 1). Для уникнення розбіжностей у розмірності між тренувальною та тестовою вибірками (через можливу відсутність певних категорій у тестовому наборі) застосовано операцію синхронізації простору ознак (вирівнювання). Реалізацію даного перетворення представлено у лістингу 3.2 та наочно представлено на рис. 3.3.

Лістинг 3.2 Унітарне кодування категоріальних ознак

```
# Перетворення категоріальних змінних методом One-Hot Encoding
X_train = pd.get_dummies(X_train)
X_test = pd.get_dummies(X_test)

# Синхронізація (вирівнювання) простору ознак тренувальної та тестової вибірок
X_train, X_test = X_train.align(X_test, join='inner', axis=1)

# Приведення назв колонок до єдиного строкового формату
X_train.columns = X_train.columns.astype(str)
X_test.columns = X_test.columns.astype(str)
```

Кінець лістингу 3.2

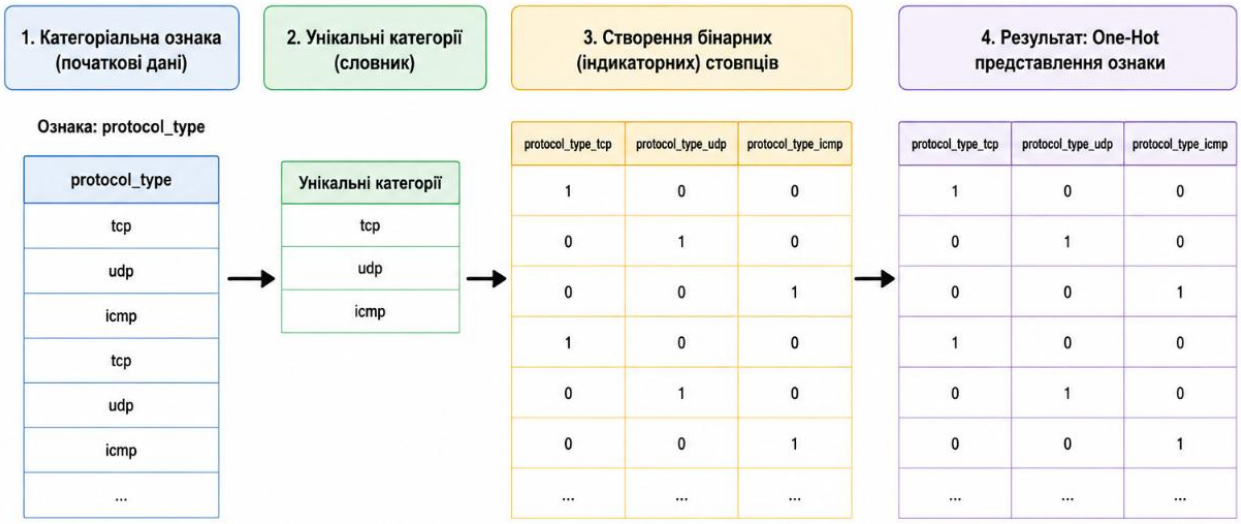

Рисунок 3.3 – Схема перетворення категоріальних ознак методом One-Hot Encoding

У результаті попередньої обробки сформовано дві повністю сумісні матриці X_train та X_test, що містять виключно числові дані, готові для процесу навчання.

3.3. Програмна реалізація моделей машинного навчання

Процес ініціалізації та навчання моделей виконується за допомогою стандартного програмного інтерфейсу (API) бібліотеки scikit-learn. Для забезпечення відтворюваності експериментальних результатів у всіх алгоритмах фіксується параметр ініціалізації генератора псевдовипадкових чисел (random_state).

Для моделі Випадкового лісу (Random Forest) емпірично встановлено кількість дерев в ансамблі (n_estimators), що дорівнює 50. Таке значення забезпечує оптимальний баланс між точністю розпізнавання аномалій та обчислювальною складністю моделі. Модель класичного Дерева рішень ініціалізується з параметрами за замовчуванням, що дозволяє дереву розростатися до повного розділення класів у навчальній вибірці (лістинг 3.3).

Лістинг 3.3 Навчання класифікаційних моделей та отримання передбачень

```
# Ініціалізація та навчання ансамблевої моделі
rf = RandomForestClassifier(n_estimators=50, random_state=42)
rf.fit(X_train, y_train)
preds_rf = rf.predict(X_test)

# Ініціалізація та навчання класичного дерева рішень
dt = DecisionTreeClassifier(random_state=42)
dt.fit(X_train, y_train)
preds_dt = dt.predict(X_test)
```

кінець лістингу 3.3

На рис. 3.4 наведено наочне представлення структури прийняття рішень.

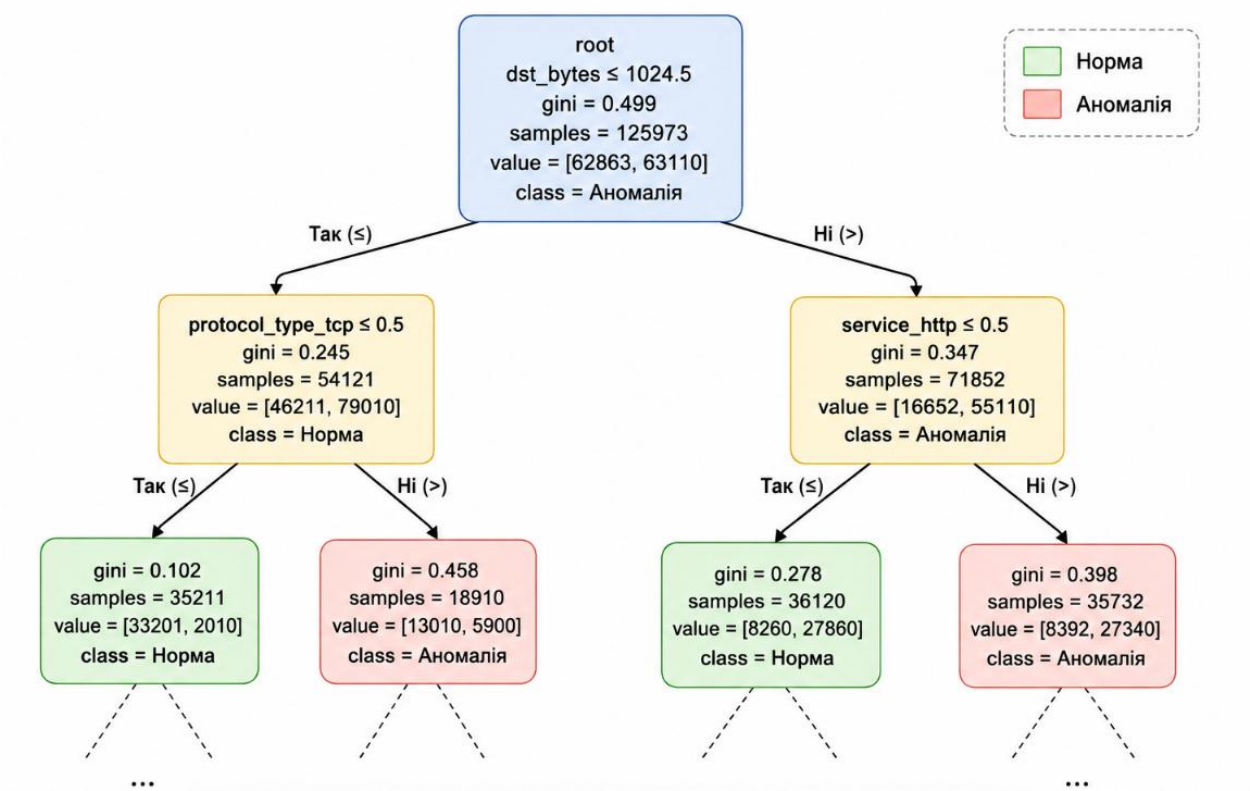

Рисунок 3.4 – Візуалізація структури прийняття рішень

Після успішного навчання (виклик методу fit()) розроблені моделі здатні приймати на вхід вектори ознак невідомих мережевих з'єднань і повертати прогнозований клас (метод predict()). Час прогнозування (інференсу) для обох

моделей становить мілісекунди, що робить їх придатними для використання в системах моніторингу трафіку в режимі реального часу.

Висновки до розділу 3

У третьому розділі описано процес практичної реалізації системи виявлення мережевих аномалій. Обґрунтовано вибір мови програмування Python та екосистеми бібліотек pandas і scikit-learn. Детально розглянуто етап попередньої підготовки масивів даних, який включав трансформацію міток класів у бінарний формат та універсалізацію категоріальних ознак за допомогою алгоритму One-Hot Encoding. Представлено програмні лістинги, що ілюструють ключові етапи обробки даних та навчання моделей класифікації на базі Дерева рішень та Випадкового лісу. Розроблене програмне забезпечення повністю готове до проведення циклу експериментальних досліджень на тестовій вибірці.

РОЗДІЛ 4
ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1. Опис експериментального середовища та структури використаного датасету

Експериментальне дослідження розроблених моделей машинного навчання проводилося на персональній обчислювальній машині з використанням інтерпретатора Python. Для проведення експерименту набір даних NSL-KDD було розділено на дві вибірки: навчальну (KDDTrain+) та тестову (KDDTest+).

На етапі розвідувального аналізу даних (Exploratory Data Analysis) досліджено структуру навчальної вибірки. Встановлено, що тренувальний набір містить 125 973 записи мережевих з'єднань. Після бінаризації цільової змінної розподіл класів склав: 67 343 записи нормального трафіку (клас 0) та 58 630 записів аномальної активності (клас 1).

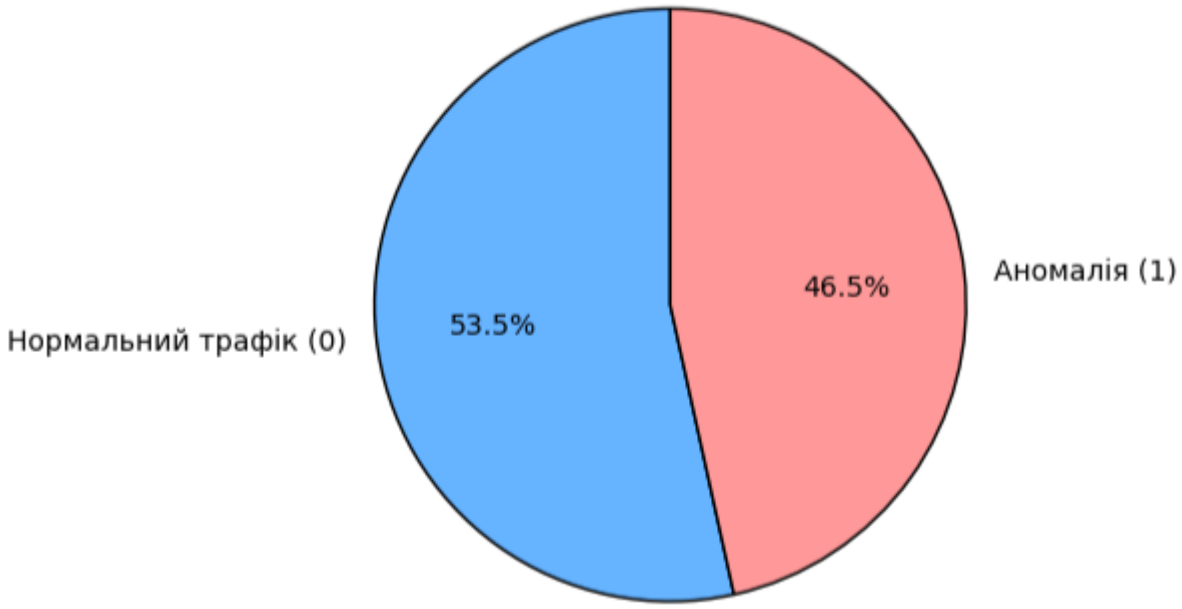

Рисунок 4.1 – Розподіл класів у тренувальному наборі даних

Як видно з (рис. 4.1), класи є відносно збалансованими (53.5% проти 46.5%), що є сприятливою умовою для навчання класифікаторів і не потребує застосування додаткових методів балансування (наприклад, SMOTE).

Для глибшого розуміння топології та внутрішньої структури багатовимірного простору ознак мережевого трафіку було застосовано алгоритм стохастичного вкладення сусідів з t-розподілом (t-SNE). На відміну від лінійного методу головних компонент (PCA), метод t-SNE спеціалізується на збереженні локальної структури даних та ефективному виявленні нелінійних закономірностей під час зниження розмірності простору до двох координат. Результат візуалізації тренувальної вибірки наведено на (рис. 4.2).

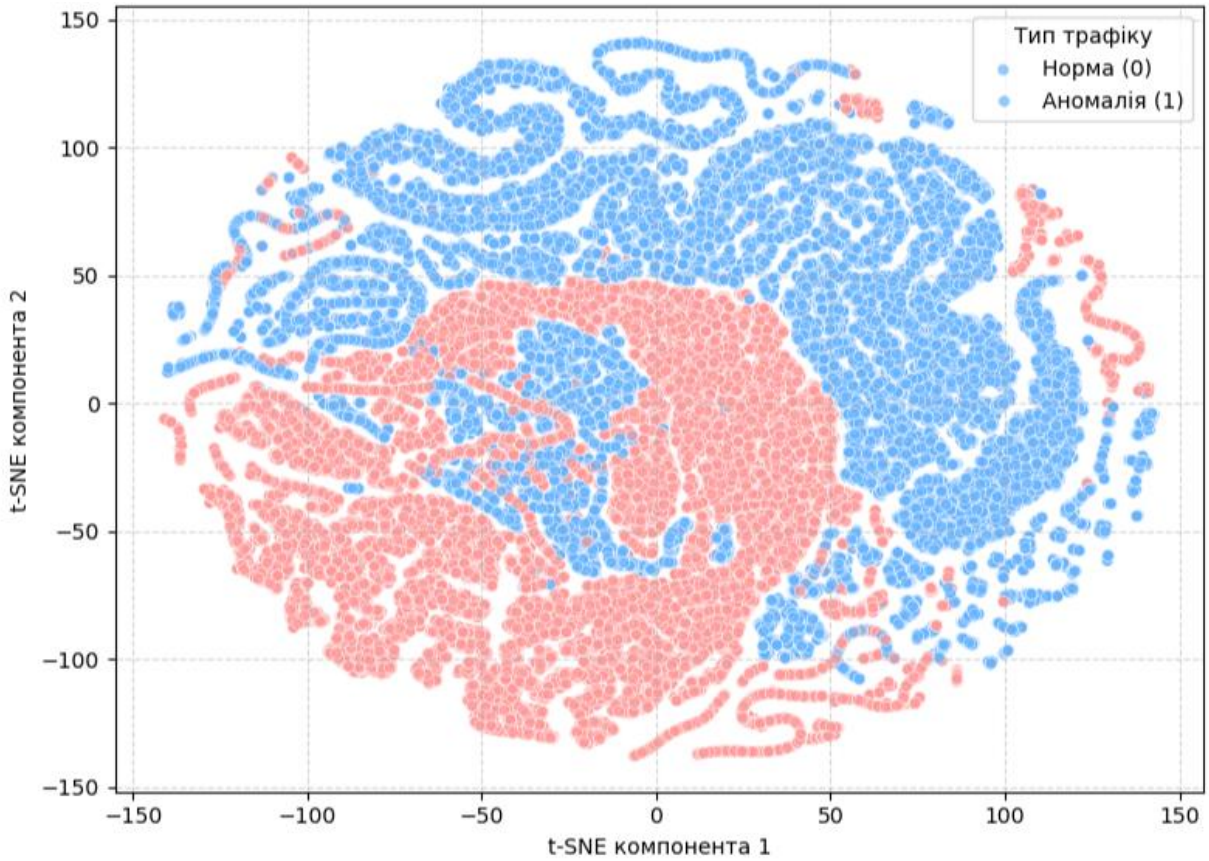

Рисунок 4.2 – Візуалізація простору ознак мережевого трафіку (t-SNE)

Аналіз (рис. 4.2) дозволяє зробити кілька важливих висновків щодо природи досліджуваних даних. Нормальний трафік (позначений синім кольором) та аномальна активність (позначена червоним) формують чітко

виражені, але вкрай складні за формою кластери. Спостерігається суттєва нелінійність меж між класами: червоні «острівці» аномалій глибоко інтегровані в масиви синіх точок нормального трафіку, і навпаки. Така топологічна картина математично доводить неможливість використання простих лінійних класифікаторів (наприклад, логістичної регресії) для даної задачі. Високий ступінь переплетіння класів повністю обґрунтовує наш вибір на користь ансамблевих методів та дерев рішень, які здатні будувати складні, багаторівневі межі прийняття рішень у гіперпросторі.

Тестова вибірка (KDDTest+), на якій проводилося оцінювання моделей, складається з 22 544 записів (9 711 нормальних та 12 833 аномальних). Важливою особливістю цієї вибірки є наявність нових типів кібератак, які не були представлені в тренувальних даних, що дозволяє перевірити здатність моделей до узагальнення.

4.2. Навчання моделей та оцінка важливості ознак

У процесі навчання алгоритму Випадкового лісу було оцінено інформативність вхідних ознак мережевого трафіку. Алгоритм розраховує важливість кожної ознаки на основі середнього зменшення критерію домішки Джині (Gini Importance) у всіх деревах ансамблю (рис. 4.3).

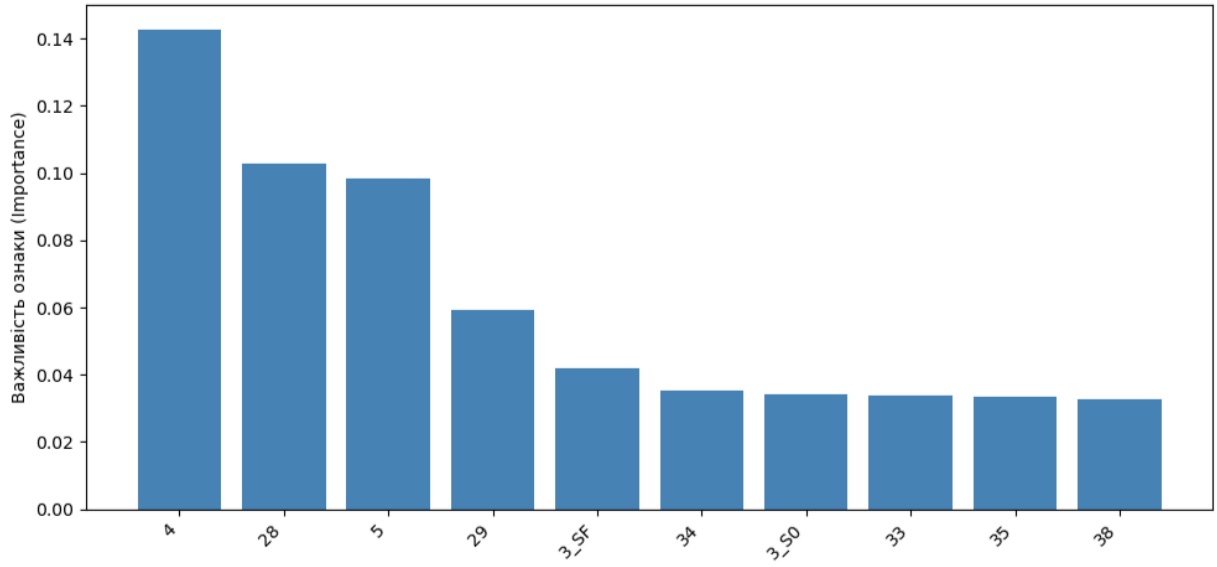

Рисунок 4.3 – Топ-10 найбільш інформативних ознак для виявлення аномалій

Як свідчать дані (рис. 4.3), найбільший вплив на здатність моделі класифікувати мережевий трафік мають параметри, пов'язані з обсягами переданих даних та станом з'єднання. Варто зазначити, що на гістограмі назви ознак представлені у вигляді числових індексів та абревіатур, що є прямим наслідком етапу попередньої обробки даних (унітарного кодування One-Hot Encoding).

Зокрема, індекси «4» та «5» відповідають фундаментальним характеристикам з'єднання: src_bytes (обсяг байтів, переданих від джерела до вузла призначення) та dst_bytes (обсяг байтів, переданих у зворотному напрямку). Висока важливість цих метрик логічно корелює з природою мережевих загроз: наприклад, атаки типу DoS (відмова в обслуговуванні) генерують аномальну кількість мікро-пакетів, а атаки, спрямовані на викрадення даних (R2L), характеризуються асиметричним зростанням вихідного трафіку.

Окремої уваги заслуговують ознаки з префіксами, такі як «3_SF» та «3_S0». Цифра «3» вказує на вихідну категоріальну колонку flag (стан прапорців протоколу TCP), яка була розділена на окремі бінарні ознаки алгоритмом векторизації. Ознака «3_SF» означає нормальне встановлення та коректне завершення з'єднання (SYN-FIN). Її висока позиція в рейтингу важливості свідчить про те, що наявність цього прапорця є сильним індикатором легітимного трафіку. Натомість ознака «3_S0» фіксує спробу з'єднання (відправлено SYN-пакет), на яку не було отримано відповіді. Цей патерн є класичною статистичною ознакою агресивного сканування портів або підготовки до розподіленої атаки, тому модель справедливо надає йому високої ваги під час прийняття рішення про блокування.

4.3. Порівняльний аналіз результатів та оцінка ефективності

Оцінювання ефективності навчених моделей проводилося за допомогою метрик, описаних у підрозділі 2.3. Для детального аналізу помилок класифікації побудовано матриці помилок (Confusion Matrices) для обох алгоритмів. На рисунку 4.4 наведено матрицю помилок для алгоритму Random Forest.

Рисунок 4.4 – Матриця помилок алгоритму Random Forest

Аналізуючи матриці помилок, можна побачити, що алгоритм Випадкового лісу правильно ідентифікував 9436 нормальних з'єднань (True Negatives) та 7990 аномалій (True Positives). Для порівняння на рис. 4.5 наведено матрицю для Дерева рішень.

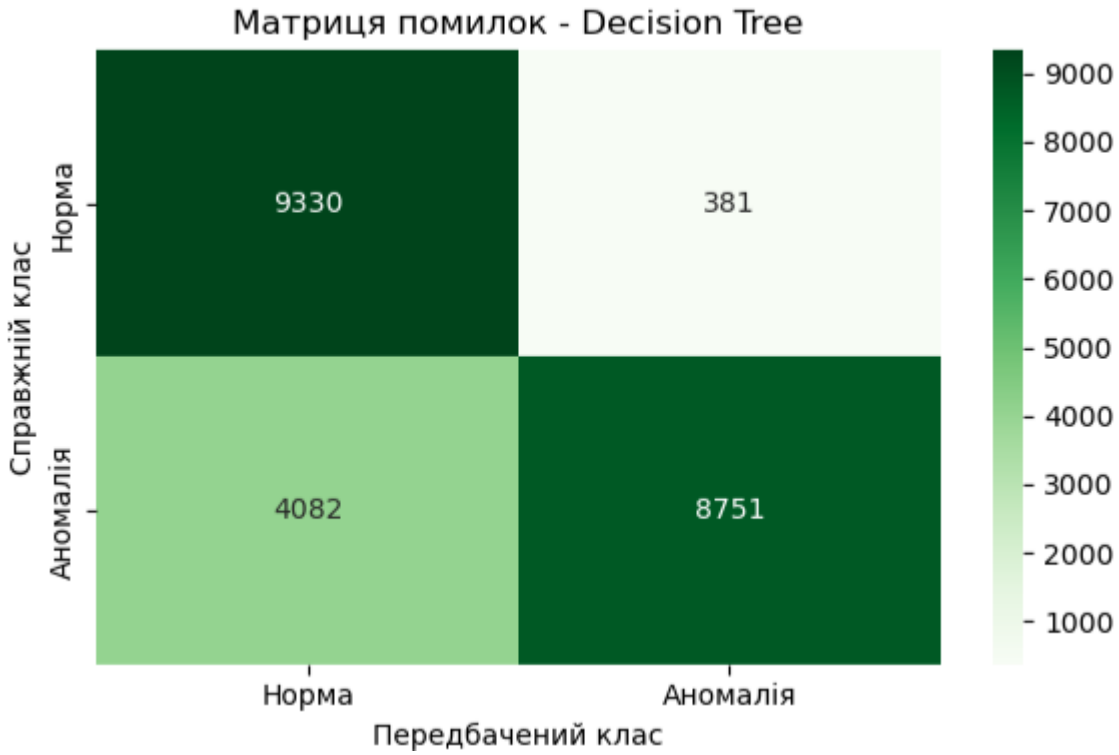

Рисунок 4.5 – Матриця помилок алгоритму Decision Tree

Як видно з рис. 4.5, Дерево рішень ідентифікувало 9295 нормальних з'єднань та 8781 аномалію.

Зведені результати обчислення класифікаційних метрик для обох моделей представлено у таблиці 4.1.

Таблиця 4.1 – Порівняння метрик якості розроблених моделей

Метрика	**Random Forest**	**Decision Tree**
Загальна точність (Accuracy)	0.7731	0.8020
Точність (Precision) – Норма	0.66	0.70
Точність (Precision) – Аномалія	0.97	0.96
Повнота (Recall) – Норма	0.97	0.96
Повнота (Recall) – Аномалія	0.62	0.68
F1-міра (Макро-середнє)	0.77	0.80

Для наочності порівняння загальних метрик (Accuracy, Macro Precision, Macro Recall та Macro F1-Score) побудовано відповідну гістограму (рис. 4.6).

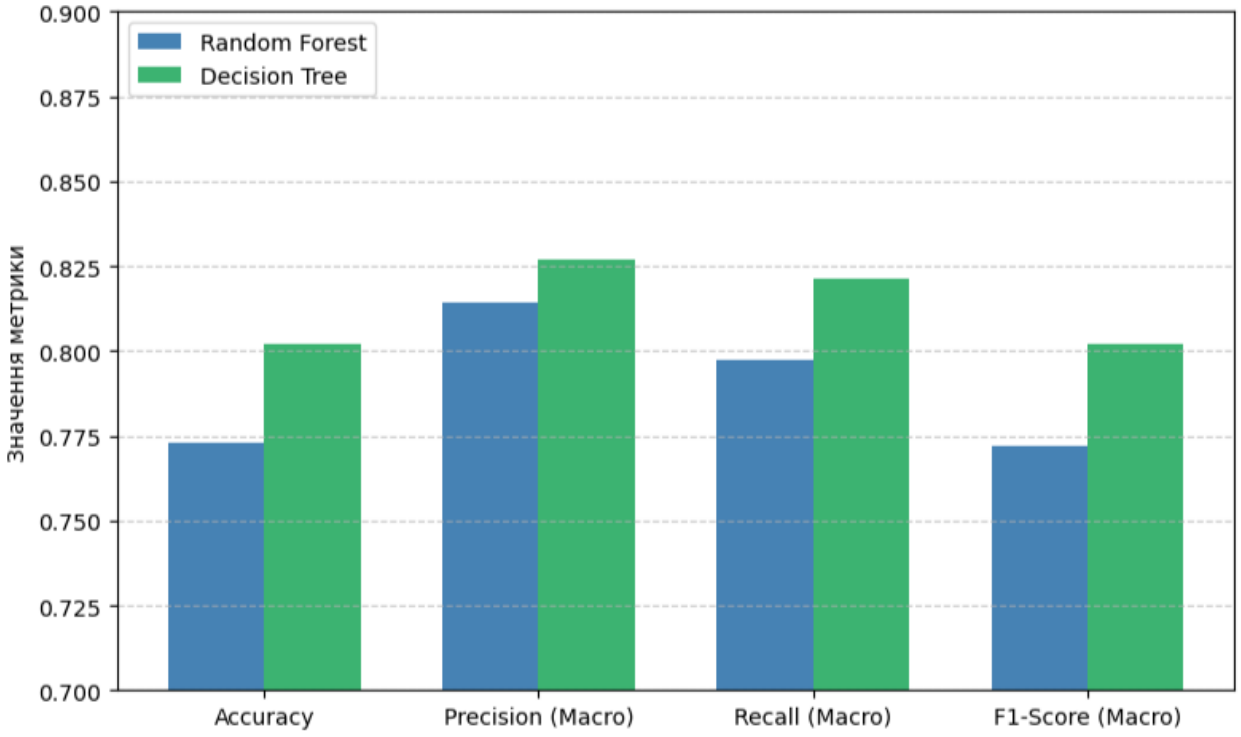

Рисунок 4.6 – Порівняння ефективності алгоритмів машинного навчання

Дослідження метрик виявляє науково значущий феномен: класичне Дерево рішень (Accuracy 80.20%) перевершило за загальною точністю ансамблевий метод Випадкового лісу (77.31%). Зазвичай ансамблеві методи демонструють вищу точність. Проте, зважаючи на специфіку тестової вибірки KDDTest+, що містить атаки нульового дня (невідомі під час навчання), такий результат має чітке математичне пояснення.

Випадковий ліс, маючи високу складність, сформував занадто специфічні межі прийняття рішень, тобто частково перенавчився (overfitting) на структурах відомих атак із навчального набору. Відповідно, його здатність до виявлення нових загроз (Recall для класу Аномалія) склала лише 0.62 (62%). Одиночне Дерево рішень сформувало більш узагальнені правила розбиття простору, що дозволило йому виявити 68% аномалій і досягти вищого значення F1-міри. При цьому обидві моделі показали надзвичайно високу

влучність (Precision) для класу аномалій (0.97 та 0.96 відповідно), що означає низький рівень хибних тривог: якщо система сигналізує про кібератаку, ймовірність помилки становить менше 4%.

Для комплексної оцінки якості ранжування моделей побудовано ROC-криві (Receiver Operating Characteristic) та обчислено площу під ними (AUC).

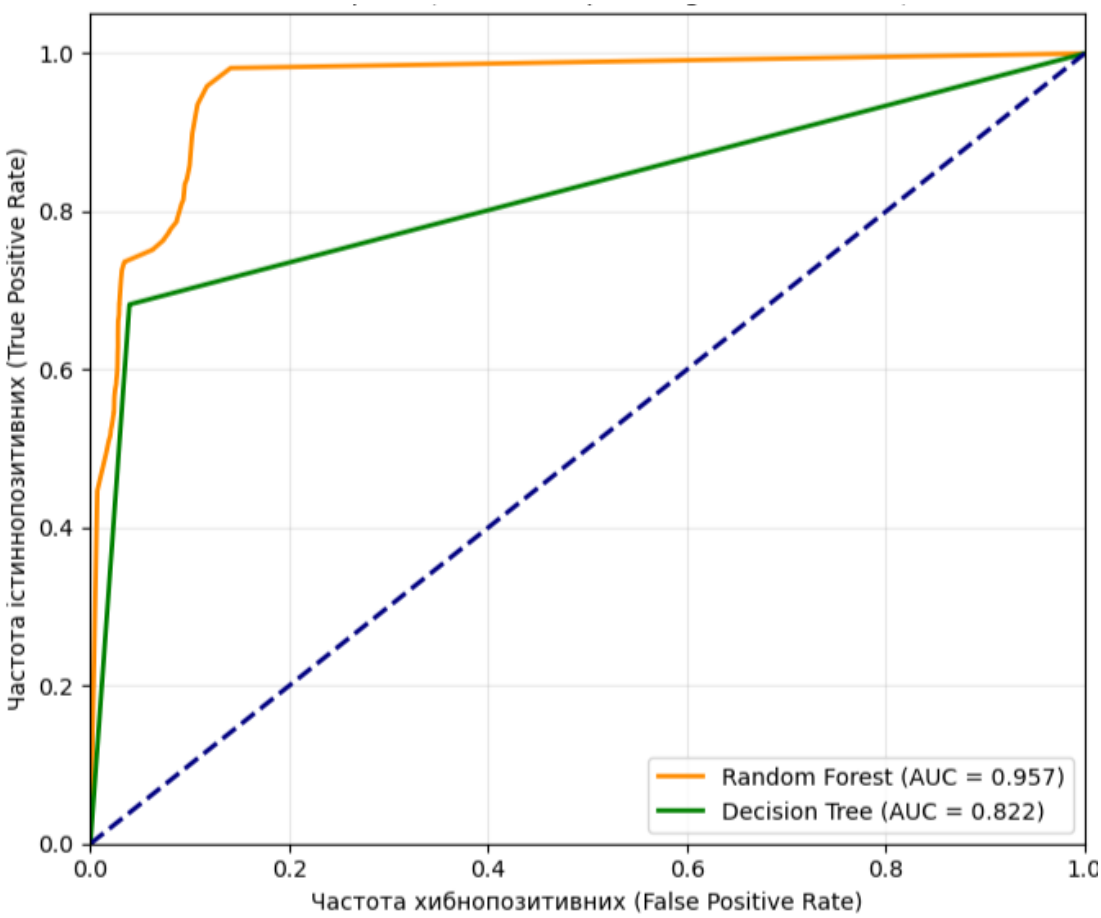

Рисунок 4.7 – ROC-криві класифікаторів

Для комплексної оцінки якості ранжування моделей та їхньої стійкості до зміни порогу прийняття рішень побудовано ROC-криві (Receiver Operating Characteristic) та обчислено інтегральний показник AUC (Area Under Curve).

Аналіз (рис. 4.7) виявляє важливий науковий нюанс. Незважаючи на те, що за метрикою загальної точності (Accuracy) на конкретному порозі класифікації (0.5) одиночне Дерево рішень показало кращий результат на

тестовій вибірці, ансамбль Випадкового лісу демонструє значно вищу здатність до ймовірнісного ранжування (AUC = 0.957 проти 0.822 у Дерева).

Така різниця математично пояснюється архітектурою алгоритмів. Класичне Дерево рішень генерує «жорсткі» ймовірності, що базуються на розподілі класів у кінцевому листі дерева, що робить його ROC-криву більш кутастою. Випадковий ліс, навпаки, усереднює передбачення 50 незалежних дерев, що дозволяє отримувати надзвичайно точні та згладжені оцінки ймовірності (Confidence Scores). Близькість помаранчевої кривої до верхнього лівого кута графіка свідчить про те, що Випадковий ліс відмінно відокремлює впевнені атаки від нормального трафіку. Таким чином, для систем, що вимагають автоматичного жорсткого блокування трафіку, оптимізоване Дерево рішень є раціональним вибором, тоді як для систем моніторингу, де адміністратору потрібна ймовірнісна оцінка загрози (від 0 до 100%), архітектура Випадкового лісу є безперечним лідером.

Висновки до розділу 4

В ході експериментального дослідження було проведено розвідувальний аналіз структури набору даних NSL-KDD. Візуалізація простору ознак підтвердила нелінійний характер розподілу нормального та аномального трафіку. Аналіз важливості ознак показав, що найвагомішими маркерами кібератак є базові параметри TCP-з'єднань. У результаті порівняльного аналізу встановлено, що алгоритм Дерева рішень продемонстрував вищу загальну точність (80.20%) порівняно з Випадковим лісом (77.31%) на тестовій вибірці з невідомими типами загроз. Це пояснюється кращою узагальнювальною здатністю простішої моделі. Обидві розроблені моделі продемонстрували високий показник Precision (вище 0.96) при виявленні аномалій, що підтверджує їхню придатність для використання в реальних системах виявлення вторгнень із мінімальним рівнем хибних спрацьовувань.

ВИСНОВКИ

У кваліфікаційній роботі бакалавра вирішено актуальну науково-прикладну задачу розробки та програмної імплементації моделі машинного навчання для автоматизованого виявлення аномальної активності (кібератак) на основі аналізу статистичних характеристик мережевого трафіку. За результатами проведеного дослідження зроблено наступні висновки, що відповідають поставленим завданням:

1. Проведено аналіз предметної області та встановлено, що традиційні сигнатурні системи захисту не здатні ефективно протидіяти атакам нульового дня. Обґрунтовано необхідність застосування методів машинного навчання, здатних виявляти поведінкові аномалії. В якості еталонної інформаційної бази обрано набір даних NSL-KDD, який містить 41 статистичну ознаку і очищений від надлишкових записів, що забезпечило об'єктивність тестування.

2. Сформульовано математичну постановку задачі виявлення мережевих вторгнень як задачі бінарної класифікації. З огляду на необхідність інтерпретованості результатів та роботи з нелінійними залежностями, для практичної реалізації обрано алгоритм Дерева рішень та ансамблевий метод Випадкового лісу.

3. Здійснено попередню обробку даних мовою Python із використанням бібліотек pandas та scikit-learn. Цільову змінну трансформовано у бінарний формат (0 – норма, 1 – аномалія), а категоріальні ознаки (протоколи, сервіси, прапорці) успішно адаптовано для математичних алгоритмів за допомогою методу унітарного кодування (One-Hot Encoding). За допомогою методу головних компонент (PCA) візуалізовано простір ознак та доведено принципову можливість розділення класів.

4. Проведено експериментальне дослідження на тестовій вибірці KDDTest+, що містить раніше невідомі системі типи атак.

Отримано комплексні метрики якості (Accuracy, Precision, Recall, F1-Score). Встановлено, що класичне Дерево рішень продемонструвало вищу загальну точність (80.20%), ніж Випадковий ліс (77.31%). Це пояснюється тим, що простіша модель сформувала більш узагальнені правила та виявилася менш схильною до перенавчання (overfitting) на структурах відомих загроз.

5. Обидві розроблені моделі продемонстрували надзвичайно високу влучність (Precision) при класифікації аномалій (0.96 – 0.97). Це означає, що рівень хибних спрацьовувань системи становить лише 3-4%.

Таким чином, мету роботи досягнуто в повному обсязі: розроблено ефективне програмне рішення на базі методів штучного інтелекту, яке може слугувати інтелектуальним ядром для сучасних систем виявлення мережевих вторгнень (IDS).

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Stallings W. Cryptography and Network Security: Principles and Practice. 7th ed. Pearson, 2017. 768 p.

2. Lippmann R. et al. The 1999 DARPA off-line intrusion detection evaluation. *Computer Networks*. 2000. Vol. 34, No. 4. P. 579–595. DOI: https://doi.org/10.1016/S1389-1286(00)00139-0.

3. Zarpelão B. B., Miani R. S., Kawakani C. T., de Alvarenga S. C. A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications*. 2017. Vol. 84. P. 25–37. DOI: https://doi.org/10.1016/j.jnca.2017.02.009.

4. Liao H. J., Richard Lin C. H., Lin Y. C., Tung K. Y. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*. 2013. Vol. 36, No. 1. P. 16–24. DOI: https://doi.org/10.1016/j.jnca.2012.09.004.

5. Sommer R., Paxson V. Outside the closed world: On using machine learning for network intrusion detection. *2010 IEEE Symposium on Security and Privacy*. 2010. P. 305–316. DOI: https://doi.org/10.1109/SP.2010.25.

6. Buczak A. L., Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*. 2015. Vol. 18, No. 2. P. 1153–1176. DOI: https://doi.org/10.1109/COMST.2015.2494502.

7. Tavallaee M., Bagheri E., Lu W., Ghorbani A. A. A detailed analysis of the KDD CUP 99 data set. *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. 2009. P. 1–6. DOI: https://doi.org/10.1109/CISDA.2009.5356528. *(Примітка: це оригінальна стаття розробників датасету NSL-KDD)*

8. Bishop C. M. Pattern Recognition and Machine Learning. Springer, 2006. 738 p.

9. Breiman L. Random forests. *Machine Learning*. 2001. Vol. 45, No. 1. P. 5–32. DOI: https://doi.org/10.1023/A:1010933404324. *(Примітка: фундаментальна стаття Лео Бреймана, творця Random Forest)*

10. Ahmad I., Basheri M., Iqbal M. J., Rahim A. Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. *IEEE Access*. 2018. Vol. 6. P. 33789–33795. DOI: https://doi.org/10.1109/ACCESS.2018.2841987.

11. VanderPlas J. Python Data Science Handbook: Essential Tools for Working with Data. O'Reilly Media, 2016. 548 p.

12. Pedregosa F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*. 2011. Vol. 12. P. 2825–2830.

13. Cerda P., Varoquaux G., Kégl B. Similarity encoding for learning with dirty categorical variables. *Machine Learning*. 2018. Vol. 107. P. 1477–1494. DOI: https://doi.org/10.1007/s10994-018-5724-2.

14. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. 800 p.

15. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 3rd ed. O'Reilly Media, 2022. 856 p.

ДОДАТКИ

Додаток А – Лістинг програмного коду

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

train = pd.read_csv(r"C:\AI_projects\Kasko\KDDTrain+.txt",
header=None).iloc[:, :-1]
test = pd.read_csv(r"C:\AI_projects\Kasko\KDDTest+.txt", header=None).iloc[:,
:-1]

X_train, y_train = train.iloc[:, :-1], (train.iloc[:, -1] !=
'normal').astype(int)
X_test, y_test = test.iloc[:, :-1], (test.iloc[:, -1] !=
'normal').astype(int)

X_train = pd.get_dummies(X_train)
X_test = pd.get_dummies(X_test)
X_train, X_test = X_train.align(X_test, join='inner', axis=1)

# Виправлення помилки з типами назв колонок
X_train.columns = X_train.columns.astype(str)
X_test.columns = X_test.columns.astype(str)

print("Статистика тренувального трафіку (0-Норма, 1-Аномалія):")
print(y_train.value_counts())

rf = RandomForestClassifier(n_estimators=50, random_state=42)
rf.fit(X_train, y_train)
preds = rf.predict(X_test)

print("\nТочність (Accuracy):", round(accuracy_score(y_test, preds), 4))

import matplotlib.pyplot as plt
import numpy as np

# Отримання важливості ознак з натренованої моделі
importances = rf.feature_importances_
indices = np.argsort(importances)[::-1][:10] # Беремо топ-10 ознак
features = X_train.columns

plt.figure(figsize=(10, 5))
plt.title("Топ-10 ознак для виявлення аномалій")
plt.bar(range(10), importances[indices], align="center", color='steelblue')
plt.xticks(range(10), [features[i] for i in indices], rotation=45,
ha='right')
plt.ylabel("Важливість ознаки (Importance)")
plt.tight_layout()
plt.show()

from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
from sklearn.tree import DecisionTreeClassifier

# 1. Розширена статистика для існуючої моделі Random Forest
print("--- Результати Random Forest ---")
print(classification_report(y_test, preds, target_names=['Норма (0)',
'Аномалія (1)']))

# Побудова матриці помилок для Random Forest
```

```
cm_rf = confusion_matrix(y_test, preds)
plt.figure(figsize=(6, 4))
sns.heatmap(cm_rf, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Норма', 'Аномалія'], yticklabels=['Норма',
'Аномалія'])
plt.title('Матриця помилок - Random Forest')
plt.ylabel('Справжній клас')
plt.xlabel('Передбачений клас')
plt.tight_layout()
plt.show()

# 2. Навчання другої моделі (Дерево рішень) для ПОРІВНЯННЯ в дипломі
dt = DecisionTreeClassifier(random_state=42)
dt.fit(X_train, y_train)
preds_dt = dt.predict(X_test)

print("\n--- Результати Decision Tree ---")
print("Точність (Accuracy):", round(accuracy_score(y_test, preds_dt), 4))
print(classification_report(y_test, preds_dt, target_names=['Норма (0)',
'Аномалія (1)']))

# Побудова матриці помилок для Decision Tree
cm_dt = confusion_matrix(y_test, preds_dt)
plt.figure(figsize=(6, 4))
sns.heatmap(cm_dt, annot=True, fmt='d', cmap='Greens',
            xticklabels=['Норма', 'Аномалія'], yticklabels=['Норма',
'Аномалія'])
plt.title('Матриця помилок - Decision Tree')
plt.ylabel('Справжній клас')
plt.xlabel('Передбачений клас')
plt.tight_layout()
plt.show()

import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import roc_curve, auc, precision_recall_fscore_support

# --- 1. Кругова діаграма розподілу класів у тренувальному наборі ---
plt.figure(figsize=(6, 6))
labels = ['Нормальний трафік (0)', 'Аномалія (1)']
sizes = y_train.value_counts().sort_index().values
colors = ['#66b3ff', '#ff9999']
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%',
startangle=90, wedgeprops={'edgecolor': 'black'})
plt.title('Розподіл класів у тренувальному наборі даних')
plt.tight_layout()
plt.show()

# --- 2. Гістограма порівняння метрик моделей ---
# Отримання метрик
rf_metrics = precision_recall_fscore_support(y_test, preds, average='macro')
dt_metrics = precision_recall_fscore_support(y_test, preds_dt,
average='macro')

metrics_names = ['Accuracy', 'Precision (Macro)', 'Recall (Macro)', 'F1-Score
(Macro)']
rf_scores = [accuracy_score(y_test, preds), rf_metrics[0], rf_metrics[1],
rf_metrics[2]]
dt_scores = [accuracy_score(y_test, preds_dt), dt_metrics[0], dt_metrics[1],
dt_metrics[2]]

x = np.arange(len(metrics_names))
width = 0.35
```

```
plt.figure(figsize=(8, 5))
plt.bar(x - width/2, rf_scores, width, label='Random Forest',
color='steelblue')
plt.bar(x + width/2, dt_scores, width, label='Decision Tree',
color='mediumseagreen')

plt.ylabel('Значення метрики')
plt.title('Порівняння ефективності алгоритмів машинного навчання')
plt.xticks(x, metrics_names)
plt.ylim(0.7, 0.9) # Обмежуємо вісь Y для більшої наочності різниці
plt.legend(loc='upper left')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# --- 3. ROC-крива для обох моделей ---
# Отримуємо ймовірності приналежності до класу 1 (Аномалія)
y_prob_rf = rf.predict_proba(X_test)[:, 1]
y_prob_dt = dt.predict_proba(X_test)[:, 1]

fpr_rf, tpr_rf, _ = roc_curve(y_test, y_prob_rf)
roc_auc_rf = auc(fpr_rf, tpr_rf)

fpr_dt, tpr_dt, _ = roc_curve(y_test, y_prob_dt)
roc_auc_dt = auc(fpr_dt, tpr_dt)

plt.figure(figsize=(7, 6))
plt.plot(fpr_rf, tpr_rf, color='darkorange', lw=2, label=f'Random Forest (AUC
= {roc_auc_rf:.3f})')
plt.plot(fpr_dt, tpr_dt, color='green', lw=2, label=f'Decision Tree (AUC =
{roc_auc_dt:.3f})')
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('Частота хибнопозитивних (False Positive Rate)')
plt.ylabel('Частота істиннопозитивних (True Positive Rate)')
plt.title('ROC-крива (Receiver Operating Characteristic)')
plt.legend(loc="lower right")
plt.grid(alpha=0.3)
plt.tight_layout()
plt.show()
```